

Lociranje i mape

Mobilne aplikacije

Stevan Gostojić

Fakultet tehničkih nauka, Novi Sad

3. decembar 2024.

Pregled sadržaja

- 1 Lociranje
- 2 Servisi za lociranje
- 3 Mape
- 4 Google Maps API
- 5 OpenStreetMap API

Lokacija

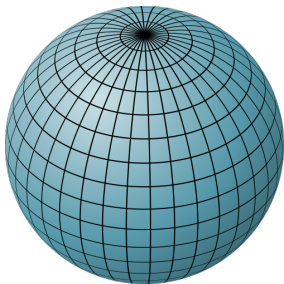


Figure 1: Geografski koordinatni sistem.

- Lokaciju na Zemljinoj površini (ili u njenoj blizini) možemo opisati u geografskom koordinatnom sistemu sa:
 - geografskom širinom
 - geografskom dužinom
 - nadmorskom visinom

Globalni navigacioni satelitski sistemi

- GPS (Global Positioning System)
- GLONASS
- COMPASS
- Galileo

GPS

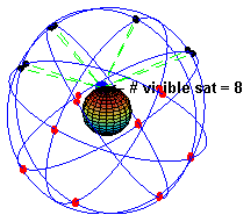


Figure 2: GPS sazvežđe.

- GPS je globalni navigacioni satelitski sistem koji omogućava određivanje lokacije i tačnog vremena
- Sastoji se iz svemirskog, kontrolnog i korisničkog segmenta
- Svemirski segment čine minimalno 24 satelita koji šalju signal na osnovu koga može da se odredi udaljenost od satelita i tačno vreme
- Korisnički segment čine GPS prijemnici koji primaju taj signal i metodom trilateracije izračunavaju svoju lokaciju i tačno vreme

GSM

- GSM mreža takođe može da se koristi za određivanje lokacije
- Svaka bazna stanica emituje CID (Cell ID)
- Postoje baze podataka koje sadrže pozicije baznih stanica
- Mobilni uređaji mogu da odrede svoju lokaciju na osnovu lokacije bazne stanice (i eventualno jačine i kašnjenja signala)

Wi-Fi

- Pristupne tačke Wi-Fi mreža emituju MAC (media access control) adresu i SSID (service set identification)
- Mobilni uređaju mogu da odrede svoju lokaciju na sličan način na koji određuju lokaciju pomoću GSM mreže

Pregled sadržaja

- 1 Lociranje
- 2 Servisi za lociranje
- 3 Mape
- 4 Google Maps API
- 5 OpenStreetMap API

Servisi za lociranje

- Android Location API nudi GPS_PROVIDER, NETWORK_PROVIDER i PASSIVE_PROVIDER (i deo je Android SDK)
- Google Location API nudi FUSED_PROVIDER (i deo je Google Play Services SDK)

Servisi za lociranje

Table 1: Android Location API

Klasa/Interfejs	Opis
LocationManager	Omogućava pristup sistemskom servisu za lociranje.
LocationProvider	Predstavljaju različite metode lociranja (Deprecated in API_LEVEL ≥ 31).
LocationListener	Koristi se za primanje obaveštenja od LocationManager-a.
Location	Predstavlja geografsku lokaciju.
Address	Predstavlja adresu.

LocationManager

- Glavna klasa Android Location API-a koja omogućava komponentama aplikacije da:
 - periodično primaju informacije o lokaciji uređaja i
 - prime obaveštenje kada se uređaj nađe u blizini zadate lokacije

Načini određivanja lokacije

- Lokacija uređaja može da se odredi na četiri načina, koja su definisana konstantama iz klase `LocationManager`:
 - `GPS_PROVIDER` (GPS)
 - `NETWORK_PROVIDER` (GSM, Wi-Fi)
 - `PASSIVE_PROVIDER` (informacije o lokaciji koje je dobavila druga aplikacija)
 - `FUSED_PROVIDER` (since `API_LEVEL 31`)

Servisi za lociranje

- 1 Zatražiti prava pristupa
- 2 Definirati obrađivač događaja koji prima obaveštenja o lokaciji
- 3 Pribaviti LocationManager
- 4 Registrovati obrađivač događaja
- 5 Odregistrovati obrađivač događaja

AndroidManifest.xml

```
1 <manifest ... >
2   <!-- ... -->
3   <uses-permission android:name="android.permission.INTERNET" />
4   <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
5   <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
6   <!-- ... -->
7 </manifest>
8
```

ExampleActivity.java

```
1  LocationListener locationListener = new LocationListener() {  
2  
3      // Called when the location has changed  
4      public void onLocationChanged(Location location) {  
5          // ...  
6      }  
7  
8      // Called when the provider is enabled by the user  
9      public void onProviderEnabled(String provider) {  
10         // ...  
11     }  
12  
13     // Called when the provider is disabled by the user  
14     public void onProviderDisabled(String provider) {  
15         // ...  
16     }  
17  
18 };  
19
```

ExampleActivity.java

```
1
2  @Override
3  public void onCreate(Bundle savedInstanceState) {
4      // ...
5      locationManager =
6          (LocationManager) getSystemService(Context.LOCATION_SERVICE);
7      Location location =
8          locationManager.getLastKnownLocation(LocationManager.FUSED_PROVIDER);
9  }
10
11  @Override
12  protected void onResume() {
13      // ...
14      locationManager.requestLocationUpdates(
15          LocationManager.FUSED_PROVIDER, 0, 0, locationListener);
16  }
17
18  @Override
19  protected void onPause() {
20      // ...
21      locationManager.removeUpdates(locationListener);
22  }
23
```


Servisi za lociranje

Table 2: Parametri metode `requestLocationUpdates`.

Parametar	Opis
<code>provider</code>	GPS_PROVIDER, NETWORK_PROVIDER, PASSIVE_PROVIDER ili FUSED_PROVIDER
<code>time</code>	minimalni vremenski interval između osvežavanja lokacije
<code>distance</code>	minimalna udaljenost između osvežavanja lokacije
<code>listener</code>	obrađivač događaja koji prima obaveštenja o lokaciji

ExampleActivity.java

```
1 locationManager.addProximityAlert(  
2     latitude, longitude, radius, expiration, PendingIntent  
3     intent) ;
```

Servisi za lociranje

Table 3: Parametri metode addProximityAlert.

Parametar	Opis
latitude	geogragska širina centralne tačke regiona
longitude	geografska dužina centralne tačke regiona
radius	poluprečnik od centralne tačke regiona (u metrima)
expiration	rok trajanja (u milisekundama) ili -1 za neograničeno trajanje
intent	PendingIntent koji će se generisati kada se detektuje ulaz u region

ExampleActivity.java

```
1 boolean enabled =
2     locationManager.isProviderEnabled (LocationManager.GPS_PROVIDER);
3
4 // check if enabled and if not send user to the GSP settings
5 // Better solution would be to display a dialog and suggesting to
6 // go to the settings
7 if (!enabled) {
8     Intent intent =
9         new Intent (Settings.ACTION_LOCATION_SOURCE_SETTINGS);
10    startActivity (intent);
11 }
12
```

Servisi za lociranje

Table 4: Svojstva klase Location.

Svojstvo	Opis
<code>double getLatitude()</code>	geografska širina (u stepenima)
<code>double getLongitude()</code>	geografska dužina (u stepenima)
<code>double getAltitude()</code>	nadmorska visina (u metrima)
<code>long getTime()</code>	vreme (u milisekundama od 1. januara 1970)
<code>float getSpeed()</code>	brzina (u metrima u sekundi)
<code>float getBearing()</code>	azimut (u stepenima)

Geokodiranje i inverzno geokodiranje

- Geokodiranje je proces transformisanja adrese ili drugog opisa lokacije u geografske koordinate (geografsku širinu i geografsku dužinu)
- Inverzno geokodiranje je proces transformisanja geografskih koordinata u (delimičnu) adresu

ExampleActivity.java

```

1 // Gets the Geocoder instance
2 Geocoder geocoder = new Geocoder(context);
3
4 // Gets address from location
5 geocoder.getFromLocation(location.getLatitude(),
6                           location.getLongitude(), 1,
7                           new Geocoder.GeocodeListener() {
8   @Override
9   public void onGeocode(@NonNull List<Address> list) {
10    //...
11  }
12 });
13
14 // Gets address from latitude and longitude
15 geocoder.getFromLocationName(locationName, 1,
16                               new Geocoder.GeocodeListener() {
17   @Override
18   public void onGeocode(@NonNull List<Address> list) {
19    //...
20  }
21 });
22

```

Geokodiranje i inverzno geokodiranje

Table 5: Parametri metode `getFromLocation`.

Metoda	Opis
<code>latitude</code>	geografska širina
<code>longitude</code>	geografska dužina
<code>maxResults</code>	maksimalni broj vraćenih adresa

Geokodiranje i inverzno geokodiranje

Table 6: Svojstva klase Address.

Metoda	Opis
<code>String getAddressLine(int index)</code>	jedan red adrese
<code>String getLocality()</code>	mesto (npr. Novi Sad)
<code>String getPostalCode()</code>	poštanski broj (npr. 21000)
<code>String getCountryName()</code>	naziv zemlje (npr. Serbia)
<code>public String getCountryCode()</code>	kod zemlje (npr. RS)
<code>double getLatitude()</code>	geografska širina (u stepenima)
<code>double getLongitude()</code>	geografska dužina (u stepenima)

Servisi za lociranje

Postoji više metoda na osnovu kojih može da se odredi lokacija uređaja (GPS, GSM i Wi-Fi). Potrebno je napraviti kompromis između različitih zahteva

Metod	Prednosti	Mane
GPS	velika preciznost, globalna pokrivenost, nije potreban internet	optička vidljivost sa satelitima, spora inicijalizacija, velika potrošnja
GSM, Wi-Fi	radi u zatvorenom prostoru, brza inicijalizacija, mala potrošnja	mala preciznost, lokalna pokrivenost, potreban internet

Servisi za lociranje

- Lokacija uređaja može da se promeni (treba je periodično osvežavati)
- Preciznost i tačnost lokacije varira
- Moguće je da je staro merenje preciznije i tačnije od novog merenja
- Takođe treba napraviti kompromis između preciznosti lokacije i perioda njenog osvežavanja (GPS) i životnog veka baterije (GSM i Wi-Fi)

Servisi za lociranje

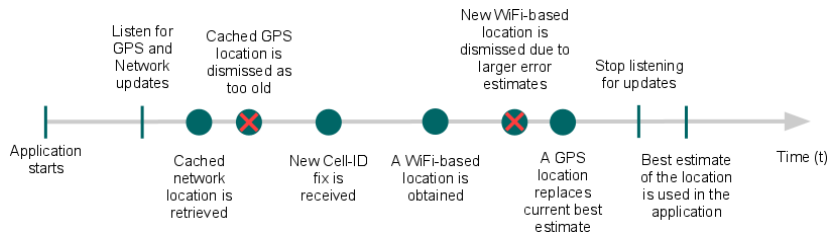


Figure 3: Tipičan rad sa servisom za lociranje.

Servisi za lociranje

- Početi sa prikupljanjem informacija o lokaciji od određenog izvora
- Pamtiti trenutno najbolju procenu lokacije filtriranjem novih ali manje preciznih merenja
- Zaustaviti prikupljanje informacija o lokaciji
- Iskoristiti poslednju najbolju procenu

Simuliranje lokacije

- ručno slanje geografskih koordinata uređaju ili korišćenje datoteka da bi se simulirala ruta
 - fiksna geografska širina i dužina
 - GPX (GPS Exchange Format)
 - KML (Keyhole Markup Language)
- geo command
 - fiksna geografska širina i dužina
 - NMEA 0183

Pregled sadržaja

- 1 Lociranje
- 2 Servisi za lociranje
- 3 Mape**
- 4 Google Maps API
- 5 OpenStreetMap API

Mapa

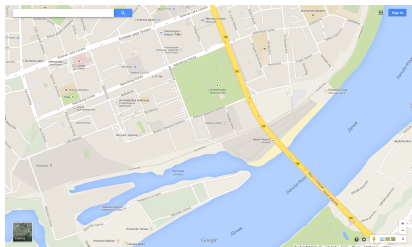


Figure 4: Mapa.

- Mapa je umanjena slika Zemljine površine na kojoj su prikazani raspored i uzajamna povezanost prirodnih, veštačkih i društvenih pojava

Elementi mape

Osnovni elementi mape su

- razmera
- pravac (severa)
- legenda
- naslov
- projekcija
- datum proizvodnje

Projekcije

- kartografska projekcija je preslikavanje geografske širine i dužine na površini geoida na lokaciju na ravni
- svaka kartografska projekcija izobličuje zemljinu površinu

Pregled sadržaja

- 1 Lociranje
- 2 Servisi za lociranje
- 3 Mape
- 4 Google Maps API
- 5 OpenStreetMap API

Google Maps API

Google Maps API omogućava korišćenje Google Maps servisa na Android platformi

- prikaz mape
- rukovanje kamerom (pomeranje, zumiranje mape, itd.)
- postavljanje oznaka, linija, poligona, itd.

Google Maps API

- preuzeti Google Play Services SDK prema uputstvu:
<https://developers.google.com/maps/documentation/android-sdk/config>
- preuzeti ključ za korišćenje Google Maps API-a
- uneti ključ u `AndroidManifest.xml`
- zatražiti odgovarajuća prava pristupa u `AndroidManifest.xml`
- deklarirati mapu kao pogled u odgovarajućem rasporedu
- definisati aktivnost koja prikazuje raspored

AndroidManifest.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest ... >
3
4   <meta-data
5       android:name="com.google.android.geo.API_KEY"
6       android:value="@string/google_maps_key" />
7
8   <uses-permission android:name="android.permission.INTERNET" />
9   <uses-permission android:name="android.permission.
10      WRITE_EXTERNAL_STORAGE" />
11   <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE
12      " />
13   <uses-permission android:name="android.permission.
14      ACCESS_COARSE_LOCATION" />
15   <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
16      />
17
18   <!-- ... -->
19
20 </manifest>
```

activity_maps.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <fragment xmlns:android="http://schemas.android.com/apk/res/android"
3     android:name="com.google.android.gms.maps.SupportMapFragment"
4     android:id="@+id/map"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"/>
7
```

ExampleActivity.java

```

1 public class ExampleActivity
2     extends FragmentActivity implements OnMapReadyCallback {
3
4     @Override
5     protected void onCreate(Bundle savedInstanceState) {
6         // ...
7         setContentView(R.layout.activity_maps);
8         SupportMapFragment mapFragment =
9             (SupportMapFragment) getSupportFragmentManager()
10                .findFragmentById(R.id.map);
11         mapFragment.getMapAsync(this);
12     }
13
14     @Override
15     public void onMapReady(GoogleMap googleMap) {
16         map = googleMap;
17
18         // Add a marker in Sydney, Australia, and move the camera.
19         LatLng sydney = new LatLng(-34, 151);
20         map.addMarker(new MarkerOptions().position(sydney).title("Marker
21         in Sydney"));
22         map.moveCamera(CameraUpdateFactory.newLatLng(sydney));

```


ExampleActivity.java

```
1 // Polylines are useful for marking paths and routes on the map.
2 map.addPolyline(new PolylineOptions().geodesic(true)
3     .add(new LatLng(-33.866, 151.195)) // Sydney
4     .add(new LatLng(-18.142, 178.431)) // Fiji
5     .add(new LatLng(21.291, -157.821)) // Hawaii
6     .add(new LatLng(37.423, -122.091)) // Mountain View
7 );
8 map.moveCamera(CameraUpdateFactory.newLatLngZoom(
9     new LatLng(-18.142, 178.431), 2));
10
11 // MAP_TYPE_TERRAIN, MAP_TYPE_HYBRID and MAP_TYPE_NONE
12 map.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
13 }
14 }
15
```

Google Maps API

Google Maps API omogućava podešavanje početnog stanja mape deklarativno (u XML dokumentu) ili proceduralno (u Java klasi)

- pozicija kamere (lokacija, zum, azimut i nagib)
- vrsta mape
- vidljivost kontrola (zum, kompas)
- omogućeni gestovi (za rukovanje kamerom)

main.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <fragment
3     map:cameraBearing="112.5 "
4     map:cameraTargetLat="-33.796923 "
5     map:cameraTargetLng="150.922433 "
6     map:cameraTilt="30 "
7     map:cameraZoom="13 "
8     map:mapType="normal "
9     map:uiCompass="false "
10    map:uiZoomControls="false "
11    map:uiRotateGestures="true "
12    map:uiScrollGestures="false "
13    map:uiTiltGestures="true "
14    map:uiZoomGestures="true "/>
15
```

ExampleActivity.java

```
1  @Override
2  public void onMapReady(GoogleMap googleMap) {
3
4      googleMap.moveCamera(CameraUpdateFactory.newLatLngZoom(
5          new LatLng(-18.142, 178.431), 2));
6
7      // MAP_TYPE_TERRAIN, MAP_TYPE_HYBRID and MAP_TYPE_NONE
8      googleMap.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
9  }
10
```

Tipovi mapa

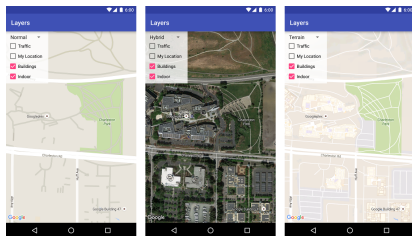
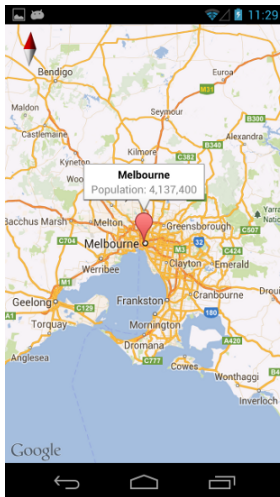


Figure 5: Tipovi mapa.

Konstanta	Opis
Normal	uobičajena autokarta
Hybrid	satelitski snimak na kome su vidljivi i putevi
Satellite	satelitski snimak
Terrain	topografska karta
None	bez pločica

Oznake



ExampleActivity.java

```
1  @Override
2  public void onMapReady(GoogleMap googleMap) {
3      map = googleMap;
4
5      // Add a marker in Sydney, Australia, and move the camera.
6      LatLng sydney = new LatLng(-34, 151);
7      map.addMarker(new MarkerOptions().position(sydney)
8          .title("Marker in Sydney"));
9      map.moveCamera(CameraUpdateFactory.newLatLng(sydney));
10 }
11
```

Oznake

Table 7: Svojstva oznake.

Svojstvo	Opis
Position	geografske koordinate oznake na karti
Alpha	prozirnost oznake
Title	tekst koji se prikazuje kada korisnik dodirne oznaku
Icon	ikona koja se prikazuje umesto podrazumevane
draggable	da li je moguće pomeriti oznaku
Visible	vidljivost oznake
Rotation	orijentacija oznake

Oblici

Na mapu je moguće dodati oblike

- linije
- poligone
- kružnice

Pregled sadržaja

- 1 Lociranje
- 2 Servisi za lociranje
- 3 Mape
- 4 Google Maps API
- 5 OpenStreetMap API

OpenStreetMap

- OpenStreetMap (OSM) online servis za prikaz mapa
- Omogućava prikaz mapa za sve delove sveta, uključujući puteve, prirodne karakteristike i znamenitosti
- Servis je besplatan, podaci koje koristi su otvoreni i na projektu su angažovani volonteri
- To je otvorena alternativa za Google Maps i slične komercijalne servise

OSMdroid API

- OSMdroid je jedna od implementacija GUI komponente za rad sa OSM mapama na Android platformi
- To je zamena za MapView klasu

OSMdroid API

- Dodati zavisnost u `build.gradle`
- Zatražiti odgovarajuća prava pristupa u `AndroidManifest.xml`
- Deklarisati mapu kao pogled u odgovarajućem rasporedu
- Definirati aktivnost koja prikazuje raspored

build.gradle

```
1 dependencies {  
2     implementation 'org.osmdroid:osmdroid-android:<VERSION>'  
3 }  
4
```

AndroidManifest.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest ... >
3
4   <!-- ... -->
5
6   <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
7     />
8   <uses-permission android:name="android.permission.INTERNET" />
9   <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"
10     />
11   <uses-permission android:name="android.permission.
12     WRITE_EXTERNAL_STORAGE" />
13
14   <!-- ... -->
15
16 </manifest>
```

src/main/res/layouts/main.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/
  android"
3   android:orientation="vertical"
4   android:layout_width="fill_parent"
5   android:layout_height="fill_parent">
6   <org.osmdroid.views.MapView android:id="@+id/map"
7     android:layout_width="fill_parent"
8     android:layout_height="fill_parent" />
9 </LinearLayout>
10
```


MainActivity.java

```

1 public class MainActivity
2     extends AppCompatActivity {
3
4     import org.osmdroid.config.Configuration;
5     import org.osmdroid.tileprovider.tilesource.TileSourceFactory;
6     import org.osmdroid.views.MapView
7
8     @Override
9     protected void onCreate(Bundle savedInstanceState) {
10         // ...
11         // Load/initialize the osmdroid configuration
12         Context ctx = getApplicationContext();
13         Configuration.getInstance().load(ctx, PreferenceManager.
14             getDefaultSharedPreferences(ctx));
15
16         map = (MapView) findViewById(R.id.map);
17         map.setTileSource(TileSourceFactory.MAPNIK);
18
19         requestPermissionsIfNecessary(arrayOf(
20             // if you need to show the current location, uncomment the line
21             // below
22             Manifest.permission.ACCESS_FINE_LOCATION,
23             // WRITE_EXTERNAL_STORAGE is required in order to show the map
24             Manifest.permission.WRITE_EXTERNAL_STORAGE
25         ));
26     }
27 }

```

MainActivity.java

```
1  @Override
2  public void onResume() {
3      super.onResume();
4      //this will refresh the osmdroid configuration on resuming.
5      //if you make changes to the configuration, use
6      SharedPreferences prefs = PreferenceManager.
7      getDefaultSharedPreferences(this);
8      Configuration.getInstance().load(this, PreferenceManager.
9      getDefaultSharedPreferences(this));
10     map.onResume(); //needed for compass, my location overlays, v6
11     .0.0 and up
12 }
13
14 @Override
15 public void onPause() {
16     super.onPause();
17     //this will refresh the osmdroid configuration on resuming.
18     //if you make changes to the configuration, use
19     SharedPreferences prefs = PreferenceManager.
20     getDefaultSharedPreferences(this);
21     Configuration.getInstance().save(this, prefs);
22     map.onPause(); //needed for compass, my location overlays, v6
23     .0.0 and up
24 }
```

MainActivity.java

```

1  @Override
2  public void onRequestPermissionsResult(int requestCode, String[]
   permissions, int[] grantResults) {
3      ArrayList<String> permissionsToRequest = new ArrayList<>();
4      for (int i = 0; i < grantResults.length; i++) {
5          permissionsToRequest.add(permissions[i]);
6      }
7      if (permissionsToRequest.size > 0) {
8          ActivityCompat.requestPermissions(
9              this,
10             permissionsToRequest.toArray(new String[0]),
11             REQUEST_PERMISSIONS_REQUEST_CODE);
12     }
13 }
14
15 private void requestPermissionsIfNecessary(String[] permissions) {
16     ArrayList<String> permissionsToRequest = new ArrayList<>();
17     for (String permission : permissions) {
18         if (ContextCompat.checkSelfPermission(this, permission) !=
19             PackageManager.PERMISSION_GRANTED) {
20             // Permission is not granted
21             permissionsToRequest.add(permission);
22         }
23     }
24     if (permissionsToRequest.size > 0) {
25         ActivityCompat.requestPermissions(
26             this,
27             permissionsToRequest.toArray(new String[0]),
28             REQUEST_PERMISSIONS_REQUEST_CODE);
29     }
30 }

```

MainActivity.java

```
1 // Add default zoom buttons, and ability to zoom with 2 fingers (
    multi-touch)
2 map.setBuiltInZoomControls(true);
3 map.setMultiTouchControls(true);
4
5 // Move the map on a default view point
6 IMapController mapController = map.getController();
7 mapController.setZoom(9.5);
8 GeoPoint startPoint = new GeoPoint(48.8583, 2.2944);
9 mapController.setCenter(startPoint);
10
```