Lociranje i mape

Mobilne aplikacije

Stevan Gostojić

Fakultet tehničkih nauka, Novi Sad

6. decembar 2022.

Pregled sadržaja

Lokacija

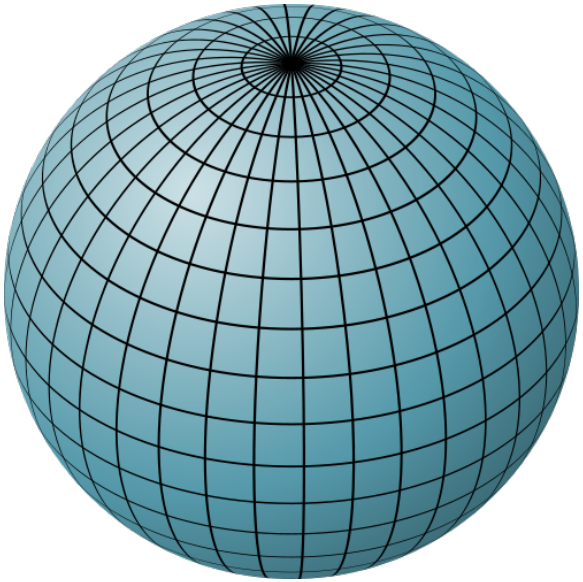


Figure 1: Geografski koordinatni sistem.

- 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

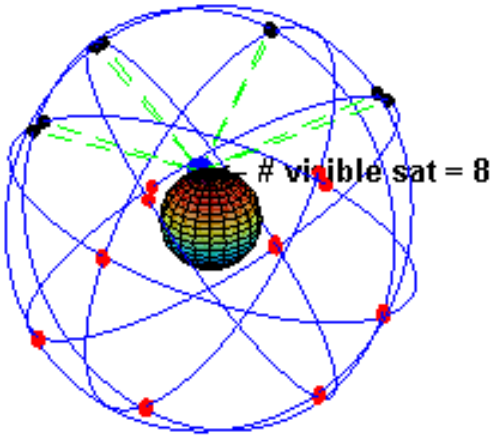


Figure 2: GPS sazvežđe.

- 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

- GMS 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

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`	Naslednice ove klase predstavljaju različite metode lociranja.
`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 i azimutu uređaja i
  - prime obaveštenje kada se uređaj nađe u blizini zadate lokacije

LocationProvider

- Lokacija uređaja može da se odredi na tri načina:
  - `GPS_PROVIDER` (GPS)
  - `NETWORK_PROVIDER` (GSM, Wi-Fi)
  - `PASSIVE_PROVIDER` (informacije o lokaciji koje je dobavila druga aplikacija)

Servisi za lociranje

1. Zatražiti prava pristupa
2. Definisati 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 status changes
9    public void onStatusChanged(String provider, int status, Bundle
       bundle) {
10     // OUT_OF_SERVICE
11     // TEMPORARILY_UNAVAILABLE
12     // AVAILABLE
13   }
14
15   // Called when the provider is enabled by the user
16   public void onProviderEnabled(String provider) {
17     // ...
18   }
19
20   // Called when the provider is disabled by the user
21   public void onProviderDisabled(String provider) {
22     // ...
23   }
24 };
25
```

ExampleActivity.java

```
1
2  @Override
3  public void onCreate(Bundle savedInstanceState) {
4    // ...
5    locationManager = (LocationManager) getSystemService(Context.
       LOCATION_SERVICE);
6    Criteria criteria = new Criteria();
7    // ...
8    LocationProvider locationProvider =
9      locationManager.getBestProvider(criteria, true);
10   location = locationManager.getLastKnownLocation(locationProvider);
11 }
12
13 @Override
14 protected void onResume() {
15   // ...
16   locationManager.requestLocationUpdates(
17     LocationManager.FUSED_PROVIDER, 0, 0, locationListener);
18 }
19
20 @Override
21 protected void onPause() {
22   // ...
23   locationManager.removeUpdates(locationListener);
24 }
25
```

Servisi za lociranje

Table 2: Parametri metode `requestLocationUpdates`.

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

ExampleActivity.java

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

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)
`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 = new Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS
       );
9    startActivity(intent);
10 }
11
```

Servisi za lociranje

Table 4: Svojstva klase Location.

Svojstvo	Opis
`double getLatitude()`	geografska širina (u stepenima)
`double getLongitude()`	geografska dužina (u stepenima)
`double getAltitude()`	nadmorska visina (u metrima)
`long getTime()`	vreme (u milisekundama od 1. januara 1970)
`float getSpeed()`	brzina (u metrima u sekundi)
`float getBearing()`	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  List<Address> addresses = geocoder.getFromLocation(
6    location.getLatitude(), location.getLongitude(), 1);
7
8  // Gets latitude and longitude from location name
9  List<Address> addresses = geocoder.getFromLocationName(locationName,
       1)
10
```

Geokodiranje i inverzno geokodiranje

Table 5: Parametri metode getFromLocation.

Metoda	Opis
`latitude`	geografska širina
`longitude`	geografska dužina
`maxResults`	maksimalni broj vraćenih adresa

Geokodiranje i inverzno geokodiranje

Table 6: Svojstva klase Address.

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

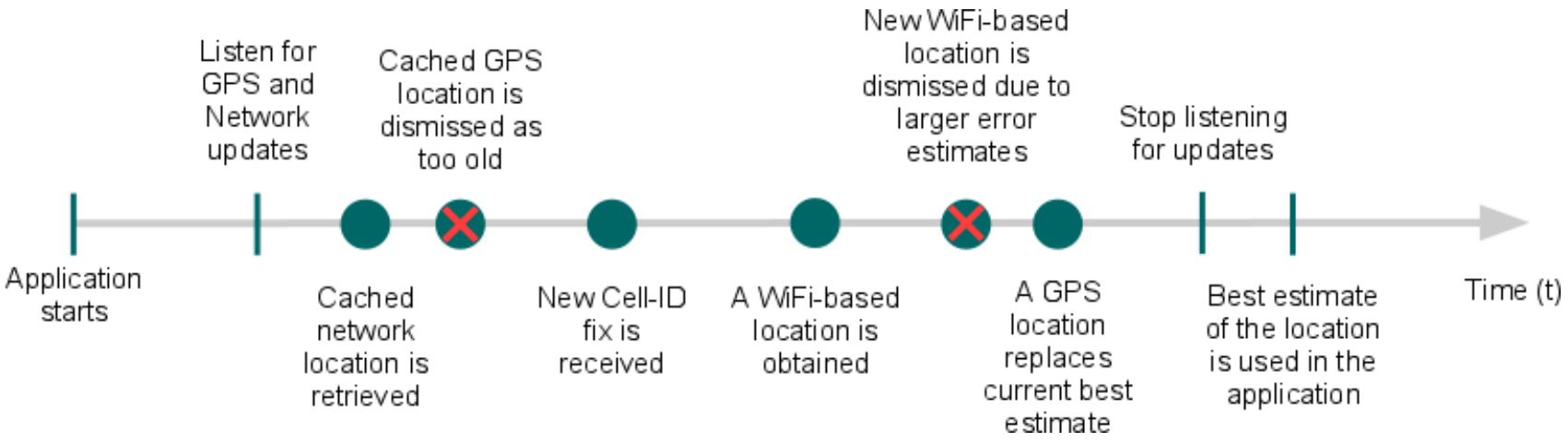


Figure 3: Tipičan rad sa servisom za lociranje.

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

Lažiranje lokacije

- DDMS (ručno slanje geografskih koordinata uređaju ili korišćenje ili datoteka da bi se lažirala 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

Mapa

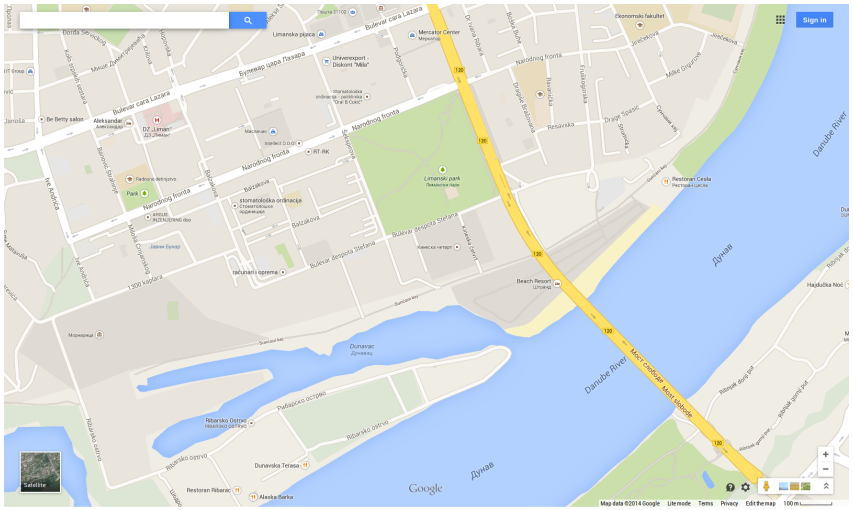


Figure 4: Mapa.

- Mapa je umanjena slika Zemljine porvšine 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

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
- preuzeti ključ za korišćenje Google Maps API-a
- uneti ključ u `AndroidManifest.xml`
- zatražiti odgovarajuća prava pristupa u `AndroidManifest.xml`
- deklarisati 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.
       WRITE_EXTERNAL_STORAGE" />
10   <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE
       " />
11   <uses-permission android:name="android.permission.
       ACCESS_COARSE_LOCATION" />
12   <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
        />
13
14   <!-- ... -->
15
16 </manifest>
17
```

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.MapFragment"
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
       in Sydney"));
21     map.moveCamera(CameraUpdateFactory.newLatLng(sydney));
22
```

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 klasu)

- pozicija kamere (lokaciju, 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

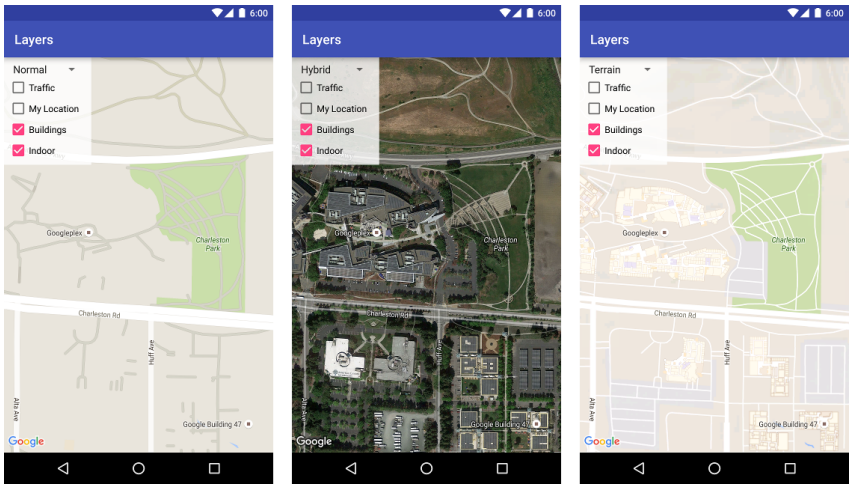
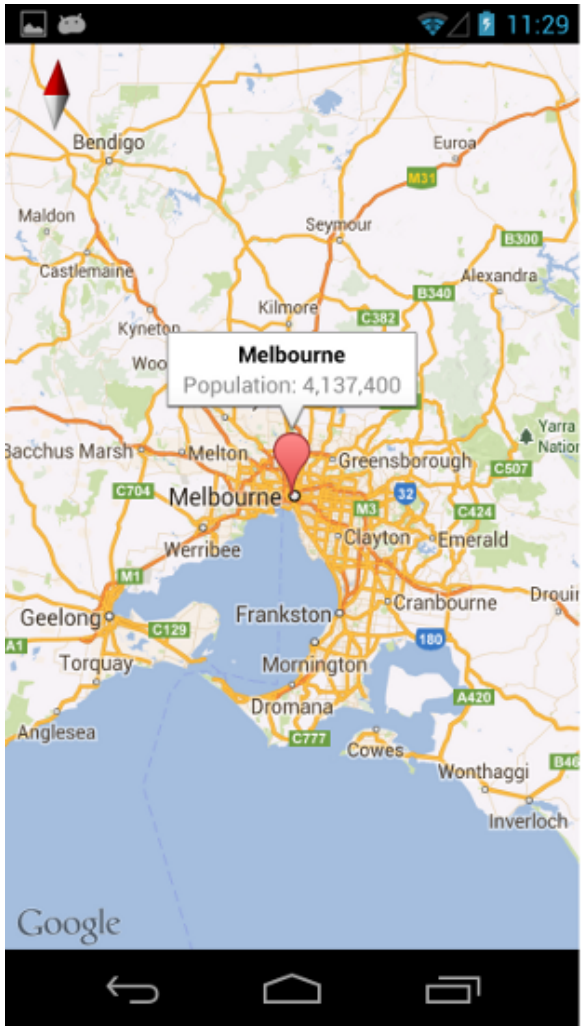


Figure 5: Tipovi mapa.

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

Oznake


11:29
Bendigo
Euroa
Maldon
Seymour
Castlemaine
Alexandra
Kilmore
Kyneton
Melbourne
Population: 4,137,400
Melton
Greensborough
Melbourne
Clayton
Emerald
Werribee
Geelong
Frankston
Cranbourne
Torquay
Mornington
Dromana
Anglesea
Cowes
Wonthaggi
Inverloch
Google


11:30
Bendigo
Euroa
Maldon
Seymour
Castlemaine
Alexandra
Melbourne
Population: 4,137,400
Melton
Greensborough
Melbourne
Clayton
Emerald
Werribee
Geelong
Frankston
Cranbourne
Torquay
Mornington
Dromana
Anglesea
Cowes
Wonthaggi
Inverloch
Google

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