Location APIs

Dominic Duggan
Stevens Institute of Technology

LOCATION API

Location Providers

```
LocationManager locationManager =
  (LocationManager)
    getSystemService(Context.LOCATION_SERVICE);
```

- Common Location Providers:
  - `LocationManager.GPS_PROVIDER`
  - `LocationManager.NETWORK_PROVIDER`

```
Location location =
  locationManager.getLastKnownLocation
    (LocationManager.GPS_PROVIDER);

boolean enabledOnly = true;
List<String> providers =
   locationManager.getProviders(enabledOnly);
```

3

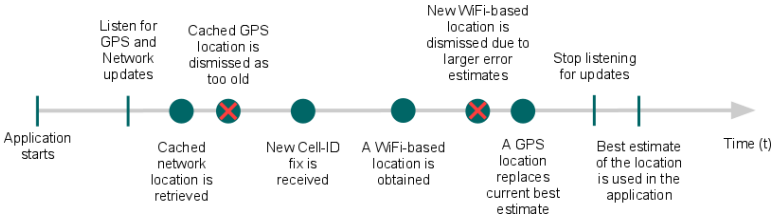
Criteria

```
Criteria criteria = new Criteria();
criteria.setAccuracy(10);
criteria.setAltitudeRequired(false);
criteria.setBearingRequired(false);
criteria.setCostAllowed(true);
criteria.setPowerRequirement(Criteria.NO_REQUIREMENT);
criteria.setSpeedRequired(false);

String bestProvider =
       locationManager.getBestProvider(criteria, true);

List<String> matchingProviders =
       locationManager.getProviders(criteria, false);
```

4

Criteria

- Return matching location provider with greatest accuracy
- If none match, relax criteria in this order:
  - Power use
  - Accuracy
  - Ability to return bearing, speed, altitude



5

Maintaining Current Best Estimate

```
long timeDelta =
   newLocation.getTime() - currentBestLocation.getTime();

boolean isSignificantlyNewer = timeDelta > TWO_MINUTES;
boolean isSignificantlyOlder = timeDelta < -TWO_MINUTES;
boolean isNewer = timeDelta > 0;

int accuracyDelta = (int)
   (newLocation.getAccuracy() –
         currentBestLocation.getAccuracy());

boolean isLessAccurate = accuracyDelta > 0;
boolean isMoreAccurate = accuracyDelta < 0;
boolean isSignificantlyLessAccurate =
   accuracyDelta > 200;
```

6

Maintaining Current Best Estimate

```
if (isSignificantlyNewer)
    currentBestLocation = newLocation;

else if (isSignificantlyOlder) {
    /* Do nothing */ ;

else if (isMoreAccurate)
    currentBestLocation = newLocation;

else if (isNewer && !isLessAccurate) {
    currentBestLocation = newLocation;

else if (isNewer &&
         !isSignificantlyLessAccurate &&
         isFromSameProvider) {
    currentBestLocation = newLocation;
```

7

Getting Location

```
locationManager = (LocationManager)
       getSystemService(Context.LOCATION_SERVICE);

Criteria criteria = ...

String provider =
       locationManager.getBestProvider(criteria, true);

Location location =
       locationManager.getLastKnownLocation(provider);

... location.getLatitude() ... location.getLongitude() ...
```

8

Getting Location

- Permissions tag

```
<uses-permission android:name=
    “android.permission.ACCESS_FINE_LOCATION”/>
<uses-permission android:name=
   “android.permission.ACCESS_COARSE_LOCATION”/>
```

9

Notification of Location Changes

```
LocationListener myLocationListener = new LocationListener() {

   public void onLocationChanged(Location location) {
        … location.getLatitude() … location.getLongitude() …
   }

   public void onProviderDisabled(String provider){
      ...
   }

   public void onProviderEnabled(String provider){
      ...
   }

   public void onStatusChanged(String provider, int status,
                               Bundle extras){
      ...
   };
```

10

Notification of Location Changes

```
locationManager
  .requestLocationUpdates(
      provider,
      2000,
      10,
      locationListener);

locationManager
   .removeUpdates(myLocationListener);
```

Time interval in milliseconds

Range of movement in meters

11

Proximity Alert (1)

```
private void setProximityAlert() {
   LocationManager locationManager = (LocationManager)
      getSystemService(Context.LOCATION_SERVICE);

   double lat = 73.147536;
   double lng = 0.510638;
   float radius = 100f; //meters
   long expiration = -1; //do not expire

   Intent intent = new Intent (TREASURE_PROXIMITY_ALERT);
   PendingIntent proximityIntent =
      PendingIntent.getBroadcast
       (getApplicationContext(), -1, intent, 0);
   locationManager.addProximityAlert
       (lat, lng, radius, expiration, proximityIntent);

   IntentFilter filter =
        new IntentFilter(TREASURE_PROXIMITY_ALERT);
   registerReceiver(new ProximityIntentReceiver(), filter);
 }
```

12

Proximity Alert (2)

```
PendingIntent proximityIntent =
   PendingIntent
     .getBroadcast (getApplicationContext(), -1, intent, 0);

locationManager.addProximityAlert (lat, lng, radius, expiration,
                                   proximityIntent);

public class ProximityIntentReceiver extends BroadcastReceiver {
     public void onReceive (Context context, Intent intent) {
       String key = LocationManager.KEY_PROXIMITY_ENTERING;

       Boolean entering = intent.getBooleanExtra(key, false);
       Toast.makeText(MyActivity.this, "Treasure: " + entering,
                      Toast.LENGTH_LONG).show();
       // ... perform proximity alert actions ...
     }
}
```


Reverse Geocoding

```
location =
   locationManager.getLastKnownLocation
         LocationManager.GPS_PROVIDER);
double latitude = location.getLatitude();
double longitude = location.getLongitude();

List<Address> addresses = null;

Geocoder geocoder =
   new Geocoder(getApplicationContext(), Locale.getDefault());

try {
   addresses = geocoder.getFromLocation(latitude, longitude, 10);
} catch (IOException e) {}

Toast.makeText(this, addresses.get(0).getCountryName(),
               Toast.LENGTH_LONG).show();
```

Forward Geocoding

```
String streetAddress =
   "160 Riverside Drive, New York, New York";

List<Address> locations = null;

Geocoder fwdGeocoder = new Geocoder(this, Locale.US);
try {
   locations =
     fwdGeocoder.getFromLocationName(streetAddress, 10);
} catch (IOException e) {}

String latLngString =
   addresses.get(0).getLatitude() + ", " +
   addresses.get(0).getLongitude();

Toast.makeText(this, latLngString , Toast.LENGTH_LONG)
   .show();
```

15

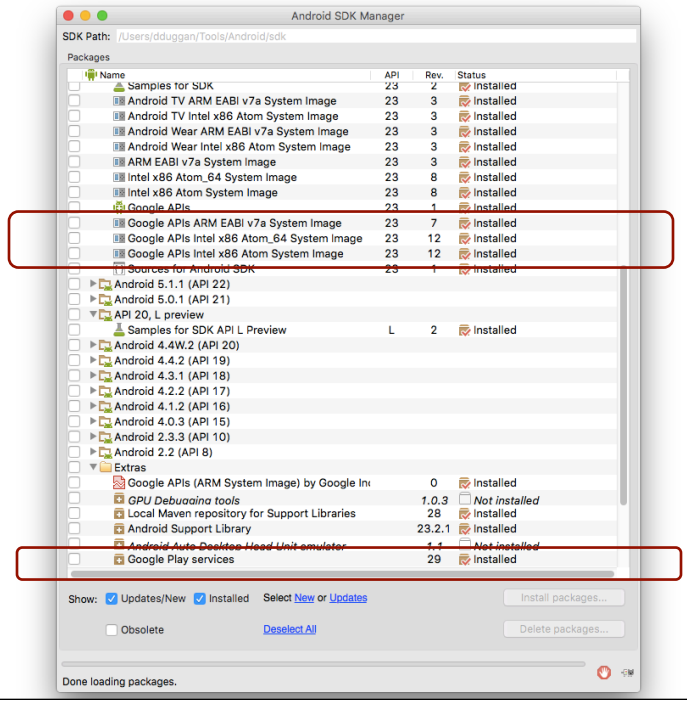
GOOGLE PLAY SERVICES

16

Installing Google Play Services

- From Android SDK:
  - Download Google APIs for Android version
  - Create AVD for Google APIs
- From Android Studio:
  - Add dependencies for Google Play Services

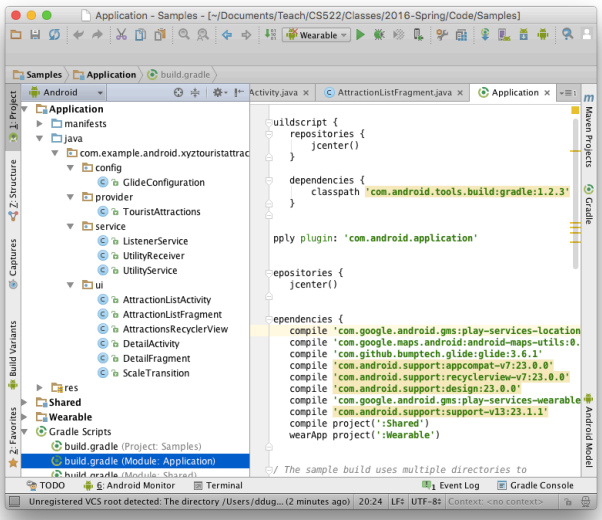
17



18

Installing Google Play Services

- Reference Play Services library from app:
  - build.gradle for application



19

Installing Google Play Services

- Android Studio: Add to application build.gradle:

```
apply plugin: 'com.android.application'
  ...
  dependencies {
   compile 'com.google.android.gms:play-services:8.4.0'
  }
```

- Selective dependencies:

```
apply plugin: 'com.android.application'
  ...
  dependencies {
   compile 'com.google.android.gms:play-services-maps:8.4.0'
   compile 'com.google.android.gms:play-services-location:8.4.0'
  }
```

Installing Google Play Services

- Add version information to app manifest:

```
<meta-data
   android:name=
      "com.google.android.gms.version"
   android:value=
     "@integer/google_play_services_version" />
```

21

Installing Google Play Services

- Check Play Services APK version

```
public static boolean checkPlayServices(Activity context) {
    int resultCode = GooglePlayServicesUtil
                     .isGooglePlayServicesAvailable(context);
    if (resultCode != ConnectionResult.SUCCESS) {
        if (GooglePlayServicesUtil
             .isUserRecoverableError(resultCode)) {
            GooglePlayServicesUtil.getErrorDialog(resultCode, context,
                                        PLAY_SERVICES_RESOLUTION_REQUEST)
                                  .show();
        } else {
            Log.e(TAG, "This device is not supported.");
            context.finish();
        }
        return false;
    }
    return true;
}
```

22

Installing Google Play Services

- Use Google API Client (Preferred)

```
if (client == null) {
    client = new GoogleApiClient.Builder(this)
        .addConnectionCallbacks(this)
        .addOnConnectionFailedListener(this)
        .addApi(LocationServices.API)
        .build();
}

protected void onStart() {
    client.connect();
    super.onStart();
}

protected void onStop() {
    client.disconnect();
    super.onStop();
}
```

23

LOCATION SERVICE API

24

Permissions

- ACCESS_COARSE_LOCATION
- ACCESS_FINE_LOCATION

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

25

Location Client

```
if (client == null) {
    client = new GoogleApiClient.Builder(this)
        .addConnectionCallbacks(this)
        .addOnConnectionFailedListener(this)
        .addApi(LocationServices.API)
        .build();
}

protected void onStart() {
    client.connect();
    super.onStart();
}

protected void onStop() {
    client.disconnect();
    super.onStop();
}
```

26

Location Client

```
public class MainActivity extends Activity implements
        GooglePlayServicesClient.ConnectionCallbacks,
        GooglePlayServicesClient.OnConnectionFailedListener {
    public void onConnected(Bundle dataBundle) { }
    ...
    public void onDisconnected() {  /* Due to error */ }
    ...
    public void onConnectionFailed(
                ConnectionResult connectionResult) {
        /* Google Play Service fails */
        if (connectionResult.hasResolution()) {
           // Start an Activity that tries to resolve the error
           connectionResult.startResolutionForResult(this,
                       CONNECTION_FAILURE_RESOLUTION_REQUEST);
        }
    }
}
```


Location Client

```
public class MainActivity extends Activity implements
        GooglePlayServicesClient.ConnectionCallbacks,
        GooglePlayServicesClient.OnConnectionFailedListener {
    ...
    /* Connected to service */
    public void onConnected(Bundle connectionHint) {
        mLastLocation = LocationServices
                        .FusedLocationApi
                        .getLastLocation(client);
        if (mLastLocation != null) {
            mLatitudeText.setText(
                String.valueOf(mLastLocation.getLatitude()));
            mLongitudeText.setText(
                String.valueOf(mLastLocation.getLongitude()));
        }
    }
}
}
```

Google Services API Client

- Create in onCreate()

```
GoogleApiClient client;
client =
   new new GoogleApiClient.Builder(this)...build();
```

- Connect in onStart()

```
client.connect();
```

- Get current location:

```
Location current =
    LocationServices.FusedLocationApi
                    .getLastLocation(client)
```

- Disconnect in onStop()

```
client.disconnect();
```

29

Location Updates

- Define callback:

```
public class MainActivity extends Activity implements
        GooglePlayServicesClient.ConnectionCallbacks,
        GooglePlayServicesClient.OnConnectionFailedListener,
        LocationListener {
    ...
    @Override
    public void onLocationChanged(Location location) {
        ... Double.toString(location.getLatitude() ...
        ... Double.toString(location.getLongitude());
    }
    ...
}
```

30

Location Updates

- Priority
  - PRIORITY_BALANCED_POWER_ACCURACY
    - City block (WIFI, cell tower)
  - PRIORITY_HIGH_ACCURACY
    - GPS
  - PRIORITY_LOW_POWER
    - City
  - PRIORITY_NO_POWER
    - Location updates for other apps

31

Location Updates

- Tune frequency & accuracy
- Update interval

```
LocationRequest.setInterval();
```

- Fastest update interval
  - Avoid data overflow
  - Save power

```
LocationRequest.setFastestInterval();
```

32

Location Updates

- Tune frequency & accuracy
- Update interval

  `LocationRequest.setInterval();`
- Fastest update interval
  - Avoid data overflow
  - Save power
  - Adjust down if long-running (e.g. network) work

  `LocationRequest.setFastestInterval();`

33

Location Updates

```
public static final int UPDATE_INTERVAL_IN_SECONDS = 10;
private static final long UPDATE_INTERVAL =
          UPDATE_INTERVAL_IN_SECONDS * 1000;
private static final int FASTEST_INTERVAL_IN_SECONDS = 5;
private static final long FASTEST_INTERVAL =
          FASTEST_INTERVAL_IN_SECONDS * 1000;

LocationRequest locationRequest = LocationRequest.create();
locationRequest
    .setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);
locationRequest.setInterval(UPDATE_INTERVAL);
locationRequest.setFastestInterval(FASTEST_INTERVAL);
```

34

Requesting Location Updates

- Create Google Play Services client on onCreate()
- Connect to Services in onStart()
- Request updates in onConnected()

```
GoogleApiClient client;
LocationRequest request;
boolean userAllowsUpdates;

public void onConnected(Bundle dataBundle) {
  if (userAllowsUpdates) {
    LocationServices.FusedLocationApi
     .requestLocationUpdates(client, request, this);
  }
}
```

35

Requesting Location Updates

- Create Google Play Services client on onCreate()
- Connect to location service in onStart()
- Disable updates in onStop()

```
GoogleApiClient client;
boolean userAllowsUpdates;

public void onStop() {
   LocationServices.FusedLocationApi
      .removeLocationUpdates(client, this);
   }
}
```

36

Requesting Location Updates

- Create Google Play Services client on onCreate()
- Connect to location service in onStart()
- Resume updates in onResume()

```
GoogleApiClient client;
LocationRequest request;
boolean userAllowsUpdates;

public void onResume() {
   if (client.isConnected())
     LocationServices.FusedLocationApi
       .requestLocationUpdates(client, request, this);
   }
}
```

37

Address Lookup

```
Private class GetAddressTask
      extends AsyncTask<Location, Void, String> {
   protected String doInBackground(Location… Params) {
      Geocoder geocoder =
         new Geocoder(context, Locale.getDefault);
      Location loc = params[0];
      List<Address> addresses = geocoder
         .getFromLocation(loc.getLatitude,
                          loc.getLongitude, 1);
      Address address = addresses.get(0);
      String addressText = ...
         ... address.getAddressLine(i)
         ... address.getLocality()
         ... address.getCountryName()
      return addressText;
   }
}
```

38

GEOFENCING

39

Geofencing

40

Geofencing

- Combine:
  - Current location
  - Proximity to locations of interest
- Mark locations of interest with latitude, longtitude
- Geofence defined by radius from location
  - Also duration
- Request enter and exit events for geofence

41

Creating a Geofence

- Geofence.builder:
  - Latitude, longitude, radius
  - Expiration time
  - Transition type (“entry” and “exit”)
  - Geofence ID
- Store geofence flattened
  - Database
  - Storage provider

42

Geofence Builder

```
mGeofenceList.add(new Geofence.Builder()
    // Set the request ID of the geofence.
   .setRequestId(entry.getKey())

   .setCircularRegion(
            entry.getValue().latitude,
            entry.getValue().longitude,
            Constants.GEOFENCE_RADIUS_IN_METERS
    )

   .setExpirationDuration(
           Constants.GEOFENCE_EXPIRATION_IN_MILLISECONDS)

   .setTransitionTypes(Geofence.GEOFENCE_TRANSITION_ENTER |
           Geofence.GEOFENCE_TRANSITION_EXIT)

    .build());
```

43

Geofence Request Builder

```
GeofencingRequest.Builder builder =
   new GeofencingRequest.Builder();

builder
   .setInitialTrigger(
     GeofencingRequest
          .INITIAL_TRIGGER_ENTER);

builder.addGeofences(mGeofenceList);

return builder.build();
```

44

Getting Geofence Transitions

- Pending intent e.g. for IntentService:

```
private PendingIntent getGeofencePendingIntent() {
    // Create an explicit Intent
    Intent intent = new Intent(this,
            GeofenceTransitionsIntentService.class);
    /*
     * Return the PendingIntent
     */
    return PendingIntent.getService(
            this,
            0,
            intent,
            PendingIntent.FLAG_UPDATE_CURRENT);
}
```

45

Sending Monitoring Requests

```
// Get the PendingIntent for the request
pendingIntent = getGeofencePendingIntent();

// Send a request to add the current geofences
LocationServices.GeofencingApi
            .addGeofences(client,
                          geofenceRequest,
                          pendingIntent)
            .setResultCallback(this);
```

46

Handling Geofence Transitions

```
public class GeofenceTransitionsIntentService extends IntentService
{
   protected void onHandleIntent(Intent intent) {
   GeofencingEvent gfEvent = GeofencingEvent.fromIntent(intent);

   if (!gfEvent.hasError(intent)) {
      int transitionType =
          geofencingEvent.getGeofenceTransition(intent);

      if ( (transitionType == Geofence.GEOFENCE_TRANSITION_ENTER)
        || (transitionType == Geofence.GEOFENCE_TRANSITION_EXIT)) {

          List <Geofence> triggerList =
             gfEvent.getTriggeringGeofences(intent);
          ...

        }
    }
```

47

Handling Geofence Transitions

```
public class GeofenceTransitionsIntentService extends IntentService
{
   protected void onHandleIntent(Intent intent) {
   GeofencingEvent gfEvent = GeofencingEvent.fromIntent(intent);

          ...

          List <Geofence> triggerList =
             gfEvent.getTriggeringGeofences(intent);

          String[] triggerIds = new String[triggerList.size()];
          // Store the Id of each geofence
          for (int i = 0; i < triggerIds.length; i++) {
              triggerIds[i] = triggerList.get(i).getRequestId();
          // Do something with the list of geofence
        }
    }
```

48

Stopping Geofence Monitoring

- Remove all geofences:

```
LocationServices.GeofencingApi
  .removeGeofences(client, pendingIntent)
  .setResultCallback(this)
```

- Remove a collection of geofences:

```
LocationServices.GeofencingApi
  .removeGeofences(client, listOfIds)
  .setResultCallback(this)
```

49

Best Practices

- Reduce power consumption
  - Higher responsiveness
  - Higher radius
- Choose optimal radius for geofence
  - Minimum 100-150m
  - WIFI 20-50m
  - Indoor 5m

50

Best Practices

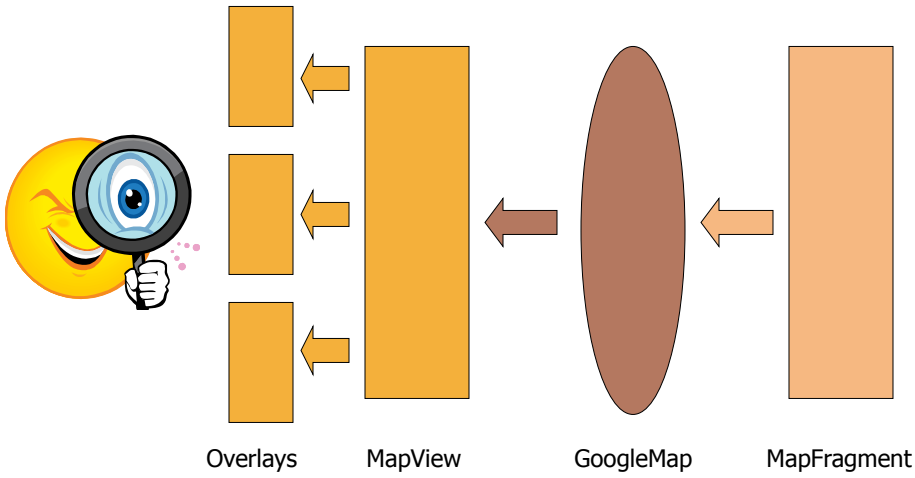
- Use dwell transition type to reduce alert spam
  - GEOFENCE_TRANSITION_DWELL instead of GEOFENCE_TRANSITION_ENTER
- Re-register geofences only when required
  - App installed/re-installed
  - App data clear
  - Google Play Services data cleared
  - GEOFENCE_NOT_AVAILABLE alert received

51

MAPS

52

Map-Based Activities



53

Map Fragment

- Create map fragment

```
mapFragment = MapFragment.newInstance();
```

- Add into activity

```
FragmentTransaction fragmentTransaction =
     getFragmentManager().beginTransaction();
fragmentTransaction
    .add(R.id.map_container, mMapFragment);
fragmentTransaction.commit();
```

- Extract map object

```
GoogleMap map = mapFragment.getMap();
```

54

Map Fragment

- Alternatively declare map fragment in layout

```
<fragment
  android:id="@+id/map"
  android:name=
      "com.google.android.gms.maps.MapFragment"
  android:layout_width="match_parent"
  android:layout_height="match_parent" />
```

- Get reference to map fragment resource by resid

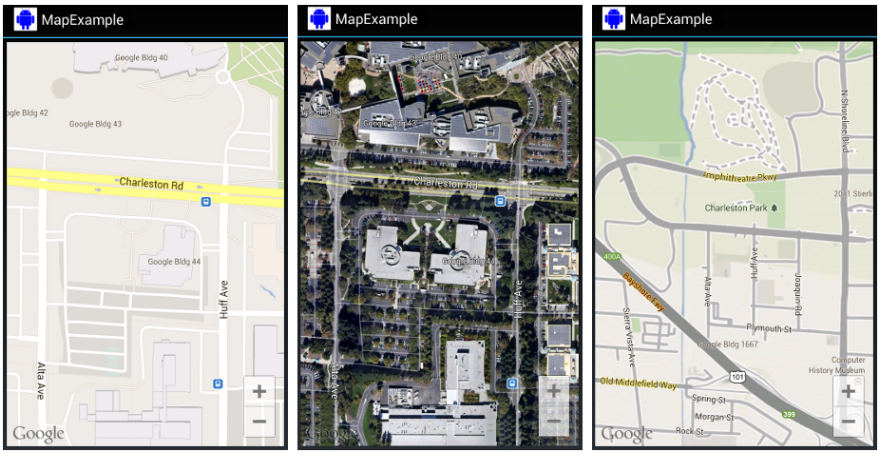
```
mapFragment = (MapFragment)
     getFragmentManager().findFragmentById(R.id.map);
```

- Extract map object

```
GoogleMap map = mapFragment.getMap();
```

55

Types of Maps



56

Types of Maps

- Normal
- Satellite
- Hybrid
- Topographic

```
GoogleMap map;
map.setMapType(GoogleMap.MAP_TYPE_HYBRID);
```

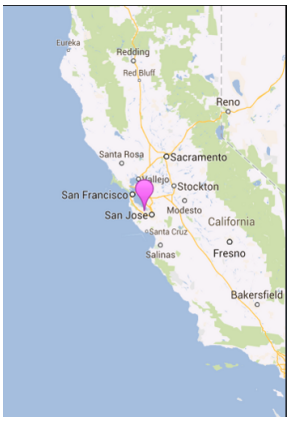
57

Map Attributes

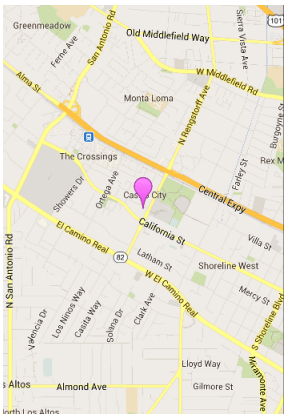
- Map type
- Camera position
  - Location: `cameraTargetLat, cameraTargetLng`
  - Zoom: `cameraZoom`
  - Bearing: `cameraBearing`
  - Tilt: `cameraTilt`
- Enable/disable zoom buttons & compass
- Gestures

58

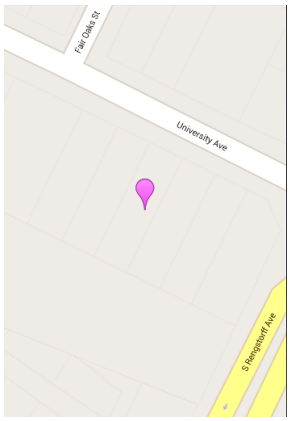
Camera Zoom



Zoom level 6



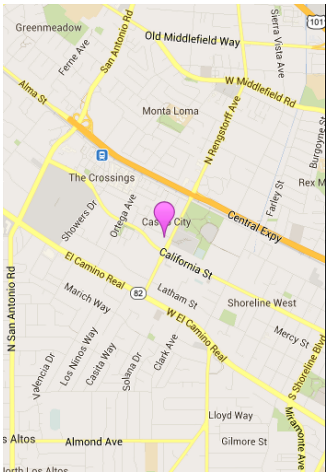
Zoom level 14

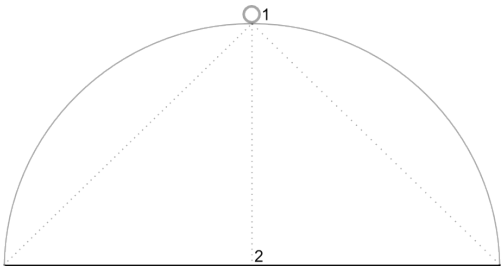


Zoom level 19

59

Camera Tilt

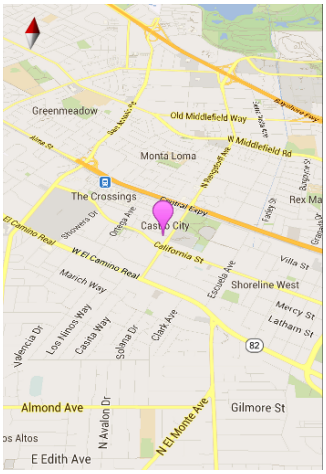


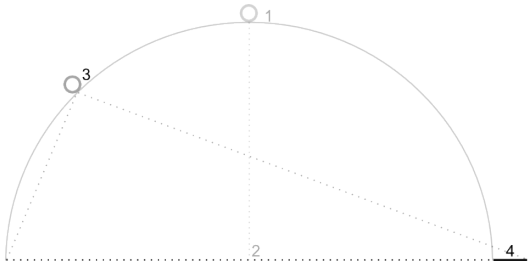


Default (0 degrees viewing angle)

60

Camera Tilt





45 degree (viewing angle)

61

```
<fragment
  xmlns:android=
       "http://schemas.android.com/apk/res/android"
  xmlns:map="http://schemas.android.com/apk/res-auto"
  android:id="@+id/map"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  class="com.google.android.gms.maps.MapFragment"
  map:cameraBearing="112.5"
  map:cameraTargetLat="-33.796923"
  map:cameraTargetLng="150.922433"
  map:cameraTilt="30"
  map:cameraZoom="13"
  map:mapType="normal"
  map:uiCompass="false"
  map:uiRotateGestures="true"
  map:uiScrollGestures="false"
  map:uiTiltGestures="true"
  map:uiZoomControls="false"
  map:uiZoomGestures="true"/>
```

62

Setting Options Programmatically

```
GoogleMapOptions options =
   new GoogleMapOptions();

Options
   .mapType(GoogleMap.MAP_TYPE_SATELLITE)
   .compassEnabled(false)
   .rotateGesturesEnabled(false)
   .tiltGesturesEnabled(false);

MapFragment.newInstance(options)

new MapView(context, options)
```

63

Camera Update

- Changing zoom Level

```
CameraUpdateFactory.zoomIn()
CameraUpdateFactory.zoomOut()

CameraUpdateFactory.zoomTo(float)

CameraUpdateFactory.zoomBy(float)

CameraUpdateFactory.zoomBy(float, Point)
```

64

Camera Update

- Changing camera position & zoom Level

```
CameraUpdateFactory.newLatLng(LatLng)

CameraUpdateFactory
   .newLatLngZoom(LatLng, float)

CameraUpdateFactory
   .newCameraPosition(CameraPosition)
new CameraPosition.Builder()
```

65

Camera Update

- Set boundaries (greatest possible zoom level)

```
CameraUpdateFactory
   .newLatLngBounds(LatLngBounds bounds,
                    int padding)

CameraUpdateFactory
   .newLatLngBounds(LatLngBounds bounds,
                    int width, int height,
                    int padding)
```

66

Camera Update

- Panning
  - +x: move camera right
  - +y: move camera down (!)
  - Direction based on current bearing

```
CameraUpdateFactory.scrollBy(float, float)
```

67

Applying Camera Update

- Animating

```
GoogleMap.moveCamera(cameraUpdate)

GoogleMap.animateCamera(cameraUpdate)

GoogleMap
  .animateCamera(cameraUpdate,
                 callback,
                 duration)
```

68

Example

```
LatLng HOBOKEN = new LatLng(47.4, -74.0);

CameraPosition cameraPosition =
   new CameraPosition.Builder()
     .target(HOBOKEN)
     .zoom(17)
     .bearing(90) // Face east
     .tilt(30)
     .build();

map.animateCamera
   (CameraUpdateFactory
      .newCameraPosition(cameraPosition));
```

69

ACCESSING MAPS API

70

Step 1: Device Certificate

- Application certificate used to sign apps
  - Debug certificate: `debug.keystore`
    - Linux, MacOS: `~/.android/debug.keystore`
    - Windows: `C:\Users\<user>\.android\debug.keystore`
  - Release certificate:

```
keytool -genkey -v -keystore my-keys.keystore
   -alias alias-name -keyalg RSA -keysize 2048
   -validity 10000
```

  - Further information:

```
http://developer.android.com/tools/publishing/app-
signing.html
```

71

Step 1: Device Certificate

- Maps API key based on SHA-1 fingerprint
  - Debug:

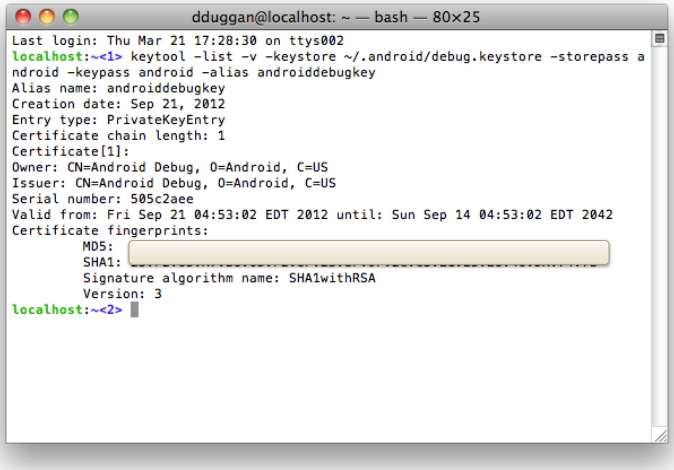
```
keytool -list -v -keystore ~/.android/debug.keystore
-storepass android -keypass android
-alias androiddebugkey
```

  - Release:

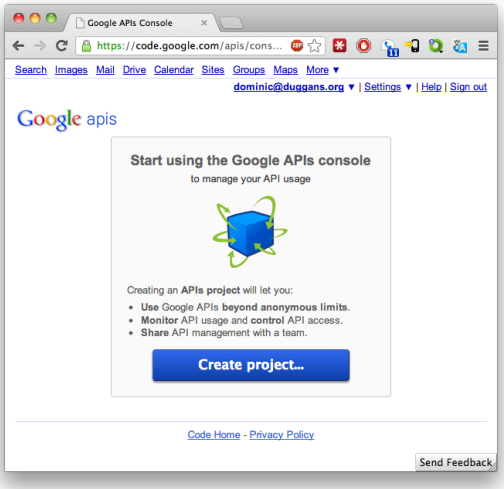
```
keytool -list -v -keystore your_keystore_name
–storepass keystore-password –keypass key-password
-alias your_alias_name
```

72

Step 1: Device Certificate

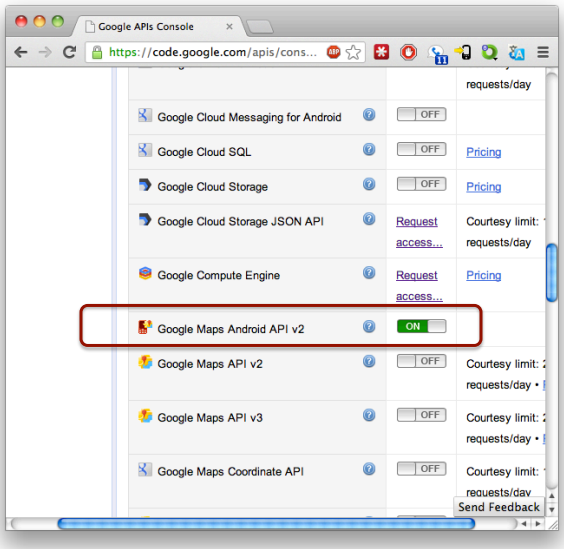


Step 2: Create API Project



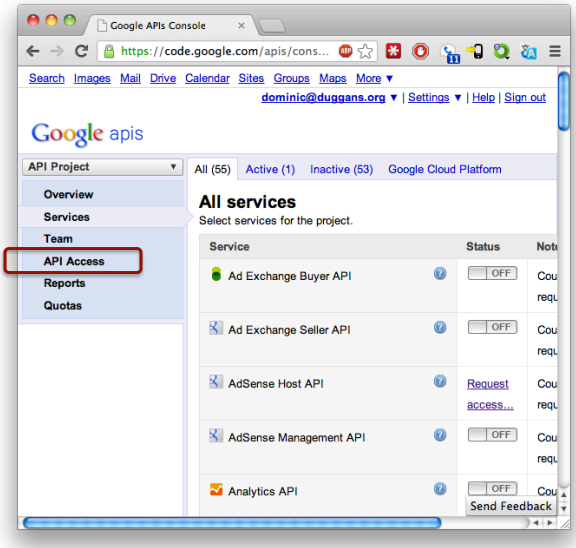
https://code.google.com/apis/console/

Step 2: Create API Project



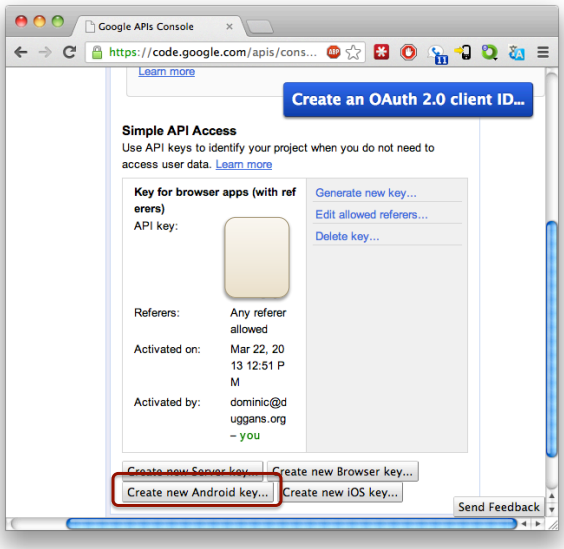
75

Step 3: Obtaining an API Key



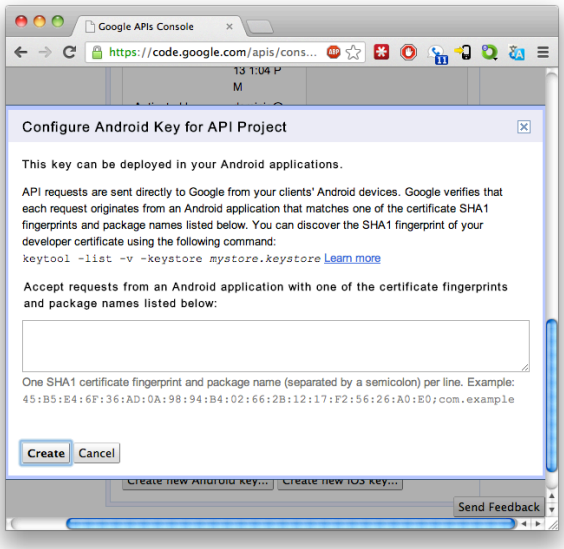
76

Step 3: Obtaining an API Key



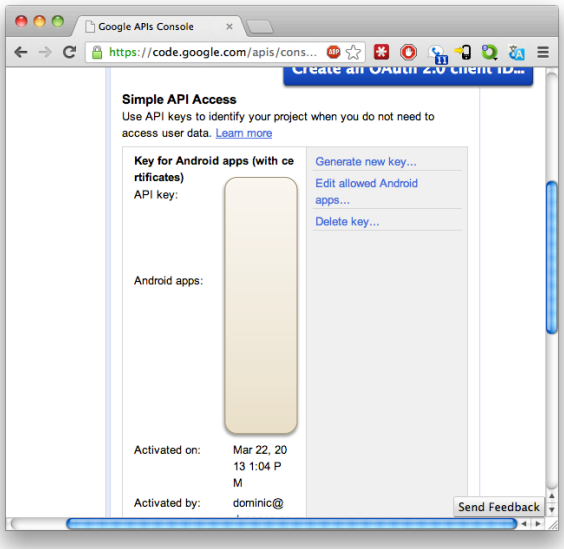
77

Step 3: Obtaining an API Key



78

Step 3: Obtaining an API Key



79

Step 4: Application Manifest

- API Key:

```
<application>
   ...
   <meta-data
      android:name=
         "com.google.android.maps.v2.API_KEY"
      android:value="api_key"/>
</application>
```

- Require OpenGL ES Version 2

```
<uses-feature
   android:glEsVersion="0x00020000"
   android:required="true"/>
```

80

Step 4: Application Manifest

- App-specific permission:

```
<permission
  android:name=
    "com.example.mapdemo.permission.MAPS_RECEIVE"
  android:protectionLevel="signature"/>

<uses-permission
  android:name=
    "com.example.mapdemo.permission.MAPS_RECEIVE"/>
```


Step 4: Application Manifest

```
<permission
  android:name="com.example.mapdemo.permission.MAPS_RECEIVE"
  android:protectionLevel="signature"/>
<uses-permission android:name="com.example.mapdemo.permission.MAPS_RECEIVE"/>
<!-- Copied from Google Maps Library/AndroidManifest.xml. -->
<uses-sdk
  android:minSdkVersion="8"
  android:targetSdkVersion="16"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"/>
<!-- External storage for caching. -->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<!-- My Location -->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<!-- Maps API needs OpenGL ES 2.0. -->
<uses-feature
  android:glEsVersion="0x00020000"
  android:required="true"/>
```

Step 4: Permissions

```
android.permission.INTERNET

com.google.android.providers
   .gsf.permission.READ_GSERVICES

android.permission.ACCESS_NETWORK_STATE

android.permission.WRITE_EXTERNAL_STORAGE

<uses-feature
        android:glEsVersion="0x00020000"
        android:required="true"/>
```

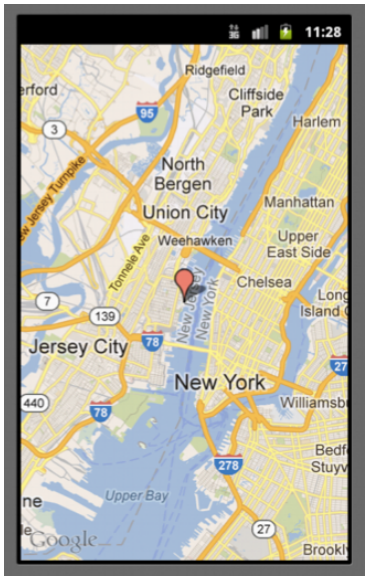
83

INTERACTING WITH MAPS

84

Markers

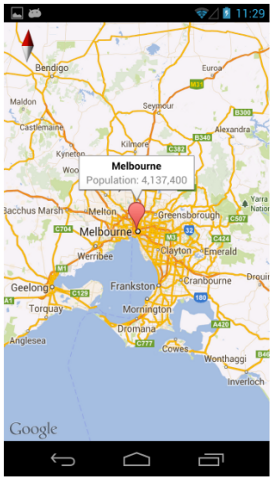
```
GoogleMap map;

map.addMarker(

   new MarkerOptions()

      .position(
       new LatLng(...))

      .title(
       "My location")

      .icon(...));
```



Marker Customization

- Position
- Title
- Snippet
- Draggable
- Visible
- Anchor
- Icon

Changing Icon




```
map.addMarker(

   new MarkerOptions()

     .position(new LatLng(-37.81319,
                          144.96298))

     .title("Melbourne")

     .snippet("Population: 4,137,400")

     .icon(BitmapDescriptorFactory
           .defaultMarker
              (BitmapDescriptorFactory.HUE_AZURE)));
```

87

Changing Icon

```
BitmapDescriptorFactory
   .fromAsset(String assetName)
   .fromBitmap (Bitmap image)
   .fromFile (String path)
   .fromResource (int resourceId)
```

```
map.addMarker(

   new MarkerOptions()

     .position(new LatLng(-37.81319,
                          144.96298))

     .title("Melbourne")

     .snippet("Population: 4,137,400")

     .icon(BitmapDescriptorFactory
           .fromResource(R.drawable.arrow)));
```

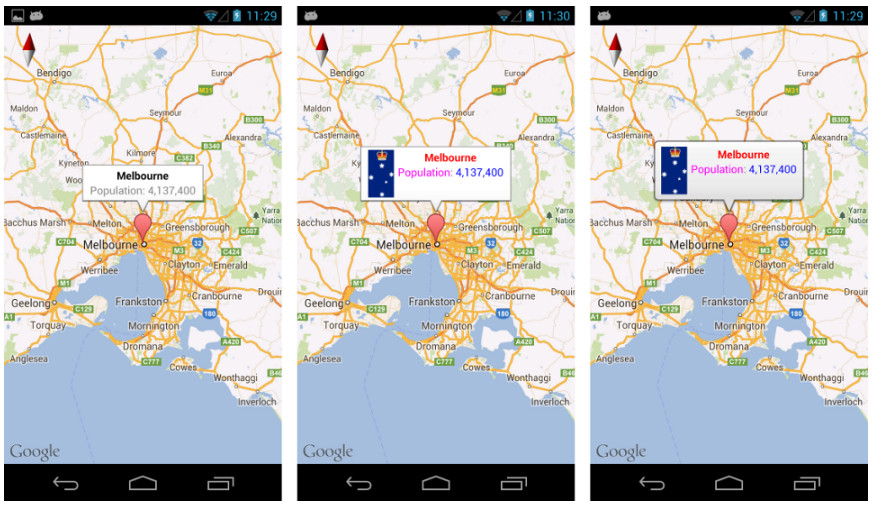
88

Changing Info Window

```
InfoWindowAdapter adapter =
   new InfoWindowAdapter() {
      public InfoWIndow getInfoWindow() {
         /* Default info window */
         /* factory method */
      }
      public InfoWIndow getInfoContents() {
         /* Called if other method returns null */
      }
   };

GoogleMap map;
map.setInfoWindowAdapter(adapter);
```

89

Changing Info Window



90

Events

- Marker click events
  `onMarkerClick(Marker)`
- Marker drag events
  `onMarkerDragListener`
  `onMarkerDragStart(Marker)`
  `onMarkerDrag(Marker)`
  `onMarkerDragEnd(Marker)`
  `Marker.getPosition()`
- Info window click events
  `onInfoWindowClick(Marker)`

91

Shapes

- Polyline
- Polygon
- Circle

```
// Add a rectangle
PolygonOptions rectOptions = new PolygonOptions()
              .add(new LatLng(37.35, -122.0),
                   new LatLng(37.45, -122.0),
                   new LatLng(37.45, -122.2),
                   new LatLng(37.35, -122.2),
                   new LatLng(37.35, -122.0));

Polygon polygon = map.addPolygon(rectOptions);
```

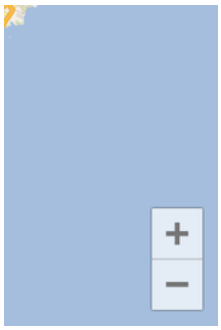
92

Donuts

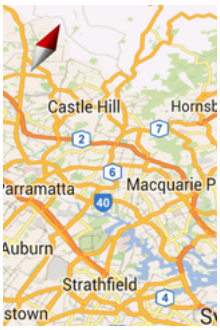
```
Polygon(
   new PolygonOptions()
        .add(new LatLng(0, 0),
             new LatLng(0, 5),
             new LatLng(3, 5),
             new LatLng(3, 0),
             new LatLng(0, 0))
        .addHole(new LatLng(1, 1),
                 new LatLng(1, 2),
                 new LatLng(2, 2),
                 new LatLng(2, 1),
                 new LatLng(1, 1))
        .fillColor(Color.BLUE));
```

93

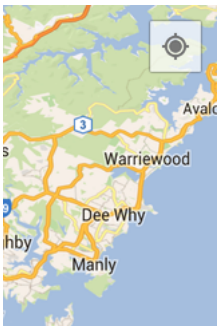
Map Controls



Zoom



Compass



Location

94

Map Gestures

- Zoom
- Scroll (pan)
- Tilt
- Rotate

95

Map Events

- Click
- Long click
- Camera change

96