



Programming with Android: The Google Maps Library

Luca Bedogni

Marco Di Felice

Dipartimento di Scienze dell'Informazione

Università di Bologna



Outline

Google Maps: History and General Information

Google Maps Library: Installation and Registration

Google Maps Library: *MapView* and *MapActivity*

Google Maps Library: *MapController*

Google Maps Library: *Overlay* Definition

Google Maps Library: *GPS* Localization

Google Maps Library: *Geocoding*



Android: Gmaps Important Dates ...

- **2004** → Google Inc bought the Australian company *Where 2 Technologies*, that developed a prototype WebMap system.
- **2005** (February) → Google Maps was announced
- **2006** → Google Maps updated to use the same satellite image database as Google Earth
- **2007** → Google Street View launched
- **2010** → On Christmas and New Year's day, mobile usage of Google Maps surpassed desktop usage for the first time
- **NOW**: Google Maps, Google Sky, Google Moon, Google Mars, Google Transit, Google Aerial View, etc



Android: Gmaps Stats and Information

- Maps are based on a variant of *Mercator* projections.
- Frequency of updates for satellite images ~ 3 years

SERVICE COVERAGE

Map Tiles: 209 countries over 218 → ~96%

Street View: 23 countries over 218 → ~10%

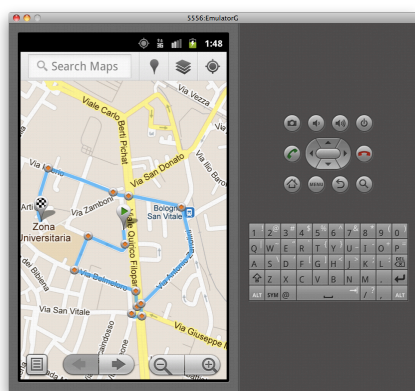
Traffic View: 22 countries over 218 → ~10%

Business info: 37 countries over 218 → ~17%



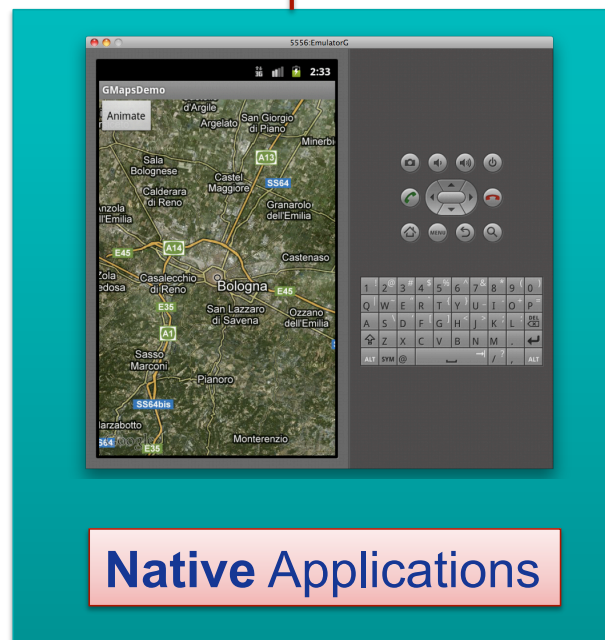
Android: Deploying Map-based Apps

Deploying Map-based Applications in Android



Hybrid Applications

WebView +
Google Maps +
Web technologies

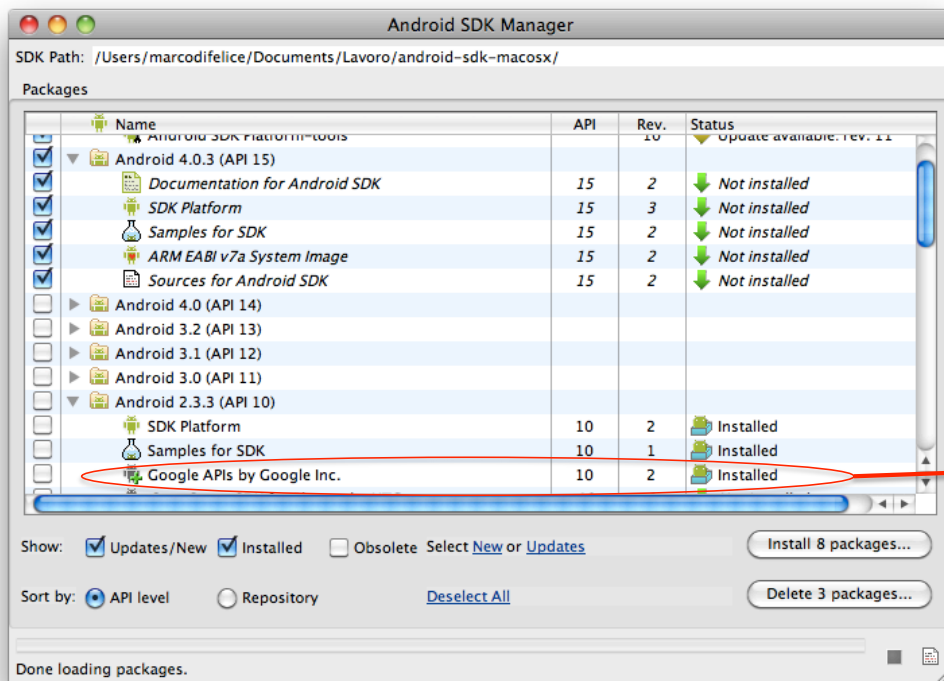


Native Applications



Android: Installing Google APIs

STEP -1: Install Google APIs to use the Maps in a native application.



Window → Android SDK Manager
→ Installed packages

Check Google APIs is installed,
or install it otherwise



Android: Getting a Google Maps API Key

STEP 0: <http://code.google.com/intl/it-IT/android/maps-api-signup.html>

Sign Up for the Android Maps API - Android Maps API - Google Code - Android - Google Developers

Home

- Google APIs Add-On
 - Overview
 - Installing
- Cloud to Device Messaging
- Android Backup Service
- Nexus Binaries
- Nexus Factory Images
- Android Developer Challenge
- ADC Sub-Saharan Africa
- www.android.com
- developer.android.com

Sign Up for the Android Maps API - Android Maps API - Google Code

The Android Maps API lets you embed [Google Maps](#) in your own Android applications. A single Maps API key is valid for all applications signed by a single certificate. See this [documentation page](#) for more information about application signing. To get a Maps API key for your certificate, you will need to provide its the certificate's fingerprint. This can be obtained using Keytool. For example, on Linux or Mac OSX, you would examine your debug keystore like this:

```
$ keytool -list -keystore ~/.android/debug.keystore
...
Certificate fingerprint (MD5): 94:1E:43:49:87:73:BB:E6:A6:88:D7:20:F1:8E:B5:98
```

If you use different keys for signing development builds and release builds, you will need to obtain a separate Maps API key for each certificate. Each key will only work in applications signed by the corresponding certificate.

You also need a [Google Account](#) to get a Maps API key, and your API key will be connected to your Google Account.

1.3. If you use the Maps APIs in conjunction with any other Google products or services, including any other Google API, (collectively, the "Services"), your agreement with Google will also include the terms applicable to those Services. All of these are referred to as the "Additional Terms." If Additional Terms apply, they will be accessible to you either within or through your use of that Service. If there is any contradiction between what any Additional Terms say and what the Maps APIs Terms say, then the Maps APIs Terms will take precedence only as it relates to the Maps APIs, and not to any other Services.

1.4. Google reserves the right to make changes to the Terms from time to time. When these changes are made, Google will make a new copy of the Terms available at <http://code.google.com/android/maps-api-tos.pdf>. You understand and agree that if you use the Service after the date on which the Terms have changed, Google will treat your use as acceptance of the updated Terms. If a modification is unacceptable to you, you may terminate the agreement by ceasing use of the Maps APIs as well distribution of any applications that use the Maps APIs.

1.5. Definitions

I have read and agree with the [terms and conditions \(printable version\)](#)

My certificate's MD5 fingerprint:

Paste here your fingerprint MD5 code

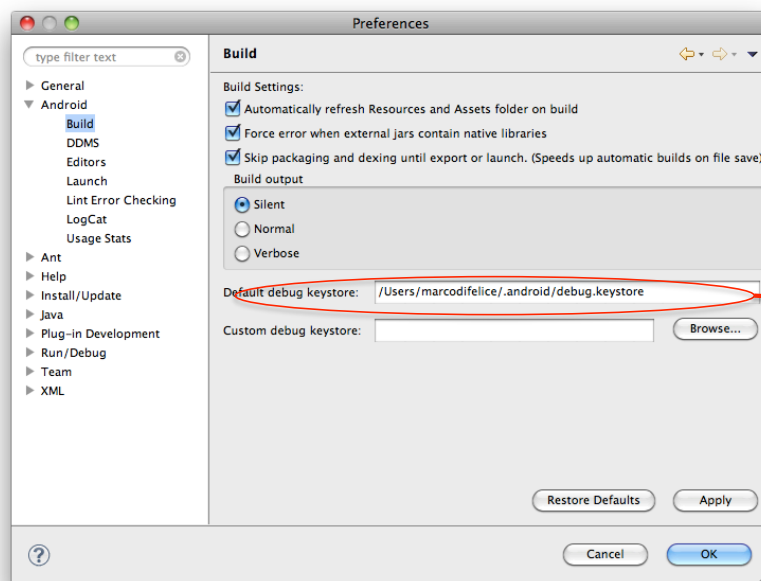
... and get the API Key



Android: Getting a Google Maps API Key

STEP 0: Get a valid **Maps API Key** to utilize the Google Maps library.

0.1: Retrieve the fingerprint MD5 of the certificate used to sign the apps.



Window → Preferences → Android
→ Build

Get the debug keystore path



Android: Getting a Google Maps API Key

STEP 0: Get a valid **Maps API Key** to utilize the Google Maps library.

0.1: Retrieve the fingerprint MD5 of the certificate used to sign the apps.

```
mylaptop:~ marco$ keytool -list -keystore /Users/  
marcodifelice/.android/debug.keystore -storepass  
android -keypass android  
...  
androiddebugkey, Feb 1, 2011, PrivateKeyEntry,  
Certificate fingerprint (MD5): A2:34:B1:A3:A5:BB:  
11:21:21:B3:20:56:92:12:AB:DB
```



Android: **Google MAPs library overview**

What can I do with **Google MAPs library** in Android?

- 1. Integrate** a Google Map into an Android application
- 2. Control** the Map visualization **options**
- 3. Customize** the Map
- 4. Integrate** the Map with GPS data



Android: Google MAPs library overview

Instantiate these objects to integrate a Google Map

1. **MapView** (com.google.android.maps.MapView)

- A **View** that displays a Map

2. **MapActivity** (com.google.android.maps.MapActivity)

- Extension of the **Activity** class
- Base class with code to manage the necessities of any Activity that displays a MapView ...



Android: Google Maps library overview

Define a **MapView** in the layout file (main.xml)

```
<LinearLayout>
...
< com.google.android.maps.MapView
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:id="@+id/map"
    android:apiKey="*****"
/>
...
</LinearLayout>
```

Paste your **Google API Key** here



Android: Google Maps library overview

Define a **MapActivity** in the Java code ..

```
public class MapDemoActivity extends MapActivity {  
    .....  
}
```

Implement the method **isRouteDisplayed()**:

```
protected boolean isRouteDisplayed() {  
    return false;  
}
```

Is my app giving routing information?



Android: Google Maps library overview

Set the permissions in the `AndroidManifest.xml`

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

Set the libraries in the `AndroidManifest.xml`

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



Android: **Google Maps library overview**

Some **methods** of a **MapView** ...

MapView options:

- public void **setSatellite**(boolean enable)
- public void **setTraffic**(boolean enable)
- public void **setStreetView**(boolean enable)

MapView interaction modes:

- public void **setClickable**(boolean enable)
- public void **setBuiltInZoomControls** (boolean enable)



Android: Google Maps library overview

➤ How to **control** the Google Map **visualization**?

... Through the **MapController** object!

- **Center** the Map at a given location
- **Zoom** in/out operations
- **Enable** animations on the map

How to get a MapController from a MapView?

```
public MapController getController()
```




Android: Google Maps library overview

Some methods of a MapController ...

Center the map at a given location:

➤ public void **setCenter(Geopoint p)**

A **GeoPoint** defines a location on the Map ...

```
GeoPoint BOLOGNA=new GeoPoint(44494290,11346526);
```

*<latitude, longitude> in microgrades, i.e. grade*10⁶*



Android: Google Maps library overview

Some methods of a MapController ...

Control the **Zoom IN/OUT** operations

- public void **zoomIn()**
- public void **zoomOut()**

Enable animations on the map

- public void **animateTo(GeoPoint gp)**
- public void **animateTo(Geopoint gp, Message msg)**
- public void **animateTo(Geopoint gp, Runnable runnable)**



Android: Google Maps library overview

Overlays → Map customizations, markers with icon, title, snippet and associated events (e.g. touch, tap, etc).

- **Overlay** (Base class representing an Overlay on the map)
- **ItemizedOverlay** (Extension of Overlay, List of **OverlayItems**)
- **MyLocationOverlay** (Extension of Overlay for drawing user's **current location** on the map, and/or a **compass-rose** inset)

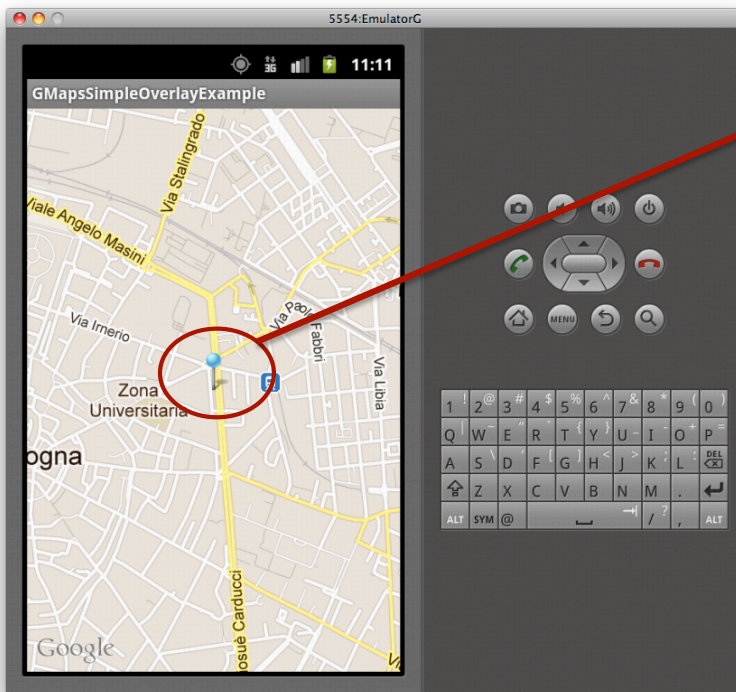
ADDING an OVERLAY to a MAPVIEW

```
mapView.getOverlays().add(newOverlay);
```



Android: Google Maps library overview

Overlay → Basic class to add a Marker to the Map ...



➤ **Extend** the Overlay class

➤ **Override** the method:

`draw(Canvas c, MapView m, boolean b)`

➤ **Add** the Overlay to the Map:

```
mapView.getOverlay.add  
(mySimpleOverlay)
```



Android: Google Maps library overview

```
class SimpleOverlay extends Overlay {
    private GeoPoint gp;
    private Bitmap marker;

    public SimpleOverlay(GeoPoint p, Bitmap d) {
        gp=p;
        marker=d;
    }

    public void draw(Canvas canvas, MapView mapView, boolean
shadow) {
        super.draw(canvas, mapView, shadow);
        Point point=new Point();
        mapView.getProjection().toPixels(gp, point);
        canvas.drawBitmap(marker, point.x-24, point.y-48, null);
    }
}
```



Android: Google Maps library overview

ItemizedOverlay → Overlay Extension, Collection of OverlayItem

OVERLAYITEM Constructor

```
OverlayItem(Geopoint gp, String title, String snippet)
```

ITEMIZEDITEM Constructor

```
ItemizedOverlay(Drawable defaultMarker)
```

Extend the ItemizedOverlay and **Override** the following methods:

- public int **size**()
- protected OverlayItem **createItem**(int i)



Android: Google Maps library overview

ItemizedOverlay → Overlay Extension, Collection of OverlayItem

OVERLAYITEM Constructor

```
OverlayItem(Geopoint gp, String title, String snippet)
```

ITEMIZEDITEM Constructor

```
ItemizedOverlay(Drawable defaultMarker)
```

Other methods:

- protected void **populate()**
- protected boolean **onTap(int index)**



Android: Google Maps library overview

MyLocationOverlay → Overlay Extension, Draw user's position

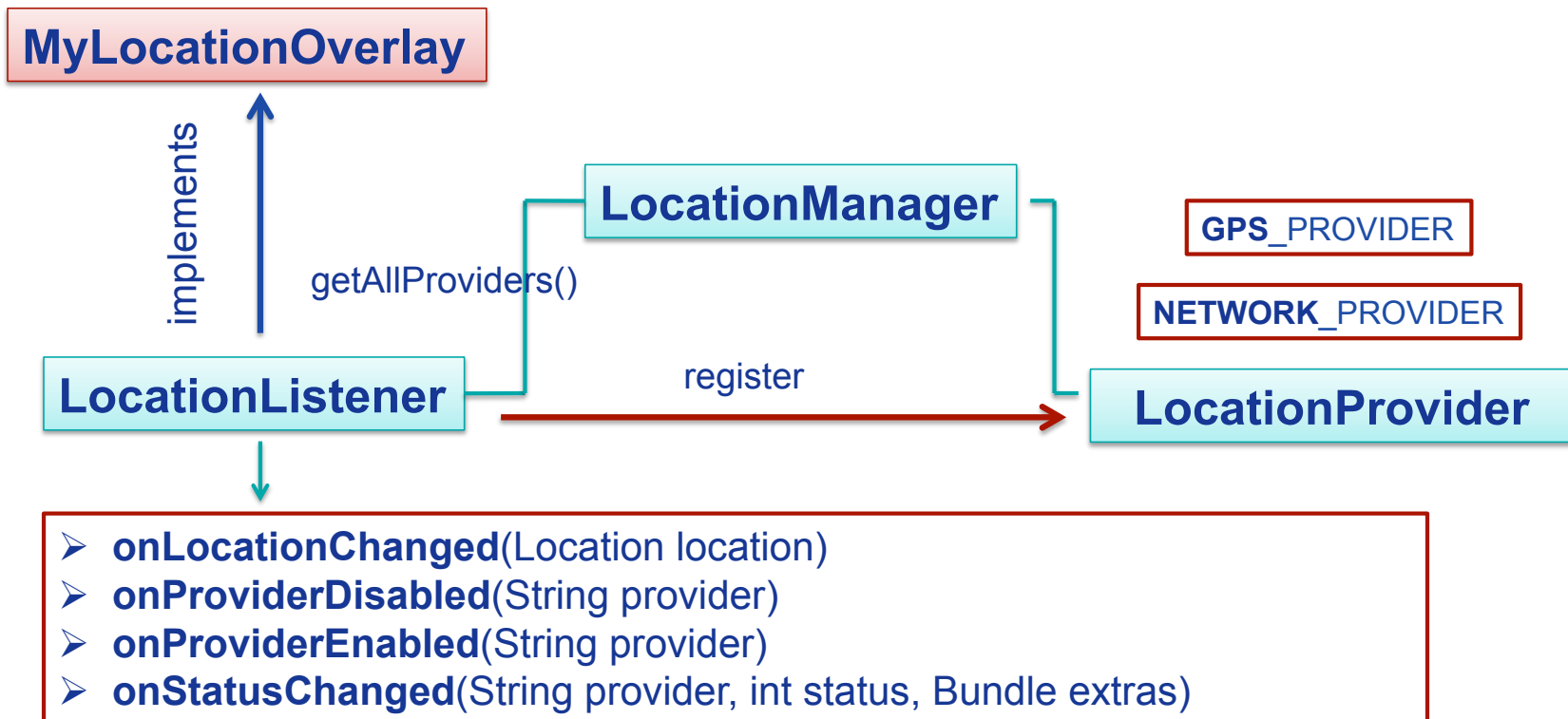
MYLOCATIONOVERLAY Constructor

```
MyLocationOverlay(Context context, MapView mapView)
```

- public boolean **enableMyLocation()**
- public boolean **disableMyLocation()**
- public GeoPoint **getMyLocation()**
- public Location **getLastFix()**
- public boolean **runOnFirstFix(Runnable)**



Android: Google Maps library overview





Android: Google Maps library overview

GeoCoding → Technique to convert an Address into a GeoPoint, or viceversa ...

Implemented by the Geocoder class

```
public Geocoder(Context contex)
```

Main methods:

- `public List<Address> getFromLocation(double latitude, double longitude, int maxResults)`
- `public List<Address> getFromLocationName(String locationName, int maxResults)`



Android: Google Maps library overview

WebView → A **View** that displays web pages, including simple browser methods (history, zoom in/out/ search, etc).

Implemented by the WebView class

```
public WebView(Context context)
```

Main methods:

- public void **loadUrl**(String url) → load the HTML page at url
- public void **loadData**(String data, String mimeType, String encoding) → load the HTML page contained in data