GETTING STARTED WITH THE GEOMOBY ANDROID SDK

FEBRUARY 2014

"GEOMOBY BRINGS LOCATION AND CONTEXT AWARENESS TO MOBILE APPS WITH A BATTERY SAFE, REAL-TIME SOLUTION FOR INDOOR/OUTDOOR GEOFENCING AND TARGETED MESSAGING"

SUPPORT:
support@geomoby.com

TABLE OF CONTENTS

PREREQUISITES

STEP ONE: Registration

Browse over to https://www.geomoby.com/signup and create an account with us. We offer a free trial which enables you up to 1000 notifications.

STEP TWO: Login

Login via the URL: https://www.geomoby.com/loginYou need to make note of ***Your API Key*** which is visible from the first page (account page) from login in. This is under your ***Business Details***, next to your ***Business Name***.

STEP THREE: Download the latest SDK

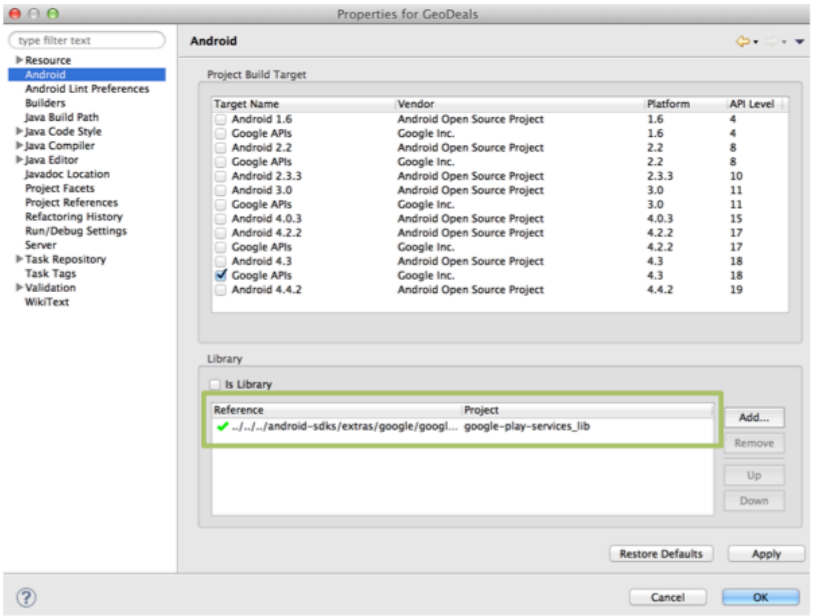
The latest SDKs can always be downloaded at https://www.geomoby.com/admin/files/

NOTE:

- Our library is compatible from **API Level 10** e.g Android 2.3.3 (GINGERBREAD)
- For iBeacon support (SDK 2.0X), you would need **API Level 18** e.g Android 4.3 (JELLY BEAN MR2)

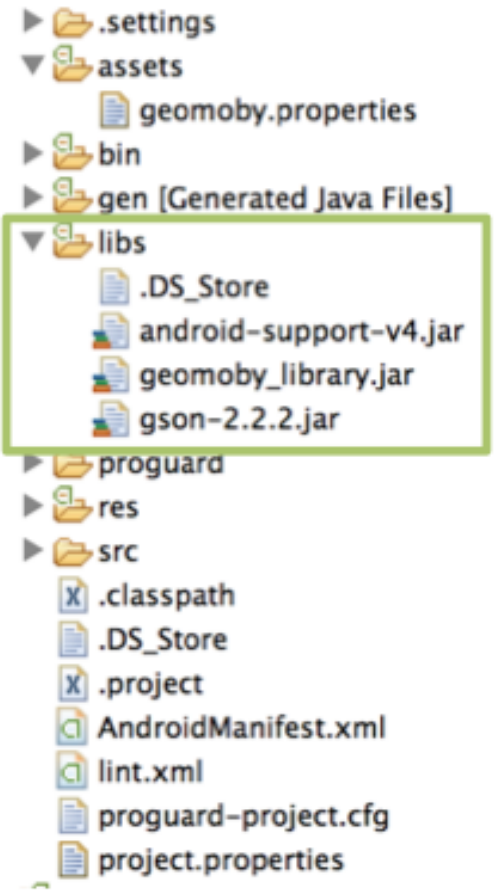
STEP FOUR: Add Google Play Services as a library

You will find all instructions here: http://developer.android.com/google/play-services/setup.html



STEP FIVE: Add GeoMoby SDK to your Project (Eclipse)

The GeoMoby Android SDK is distributed as a JAR library and comes with a few dependencies. Simply add the .jars into your project's libs/ directory and voila!



For developers with access to the GeoMoby Android SDK source code, you can also reference the SDK as an Android Library Project.

After pulling down the source you can easily reference an Android Library project from Eclipse or from the command line.

If you're having trouble, try checking your application's ***project.properties*** file.

USAGE

We often update the usage documentation on our website, check it out:
https://www.geomoby.com/admin/doc/sdk_android

STEP ONE: Permissions in your Manifest file

Before you begin, you'll need to modify your application's ***AndroidManifest.xml*** file to include the following permissions:

```
<!-- ****** START: GeoMoby Permissions ******* -->
<uses-sdk android:minSdkVersion="10" android:targetSdkVersion="19" />
<uses-permission android:name="android.permission.INTERNET" >
</uses-permission>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" >
</uses-permission>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" >
</uses-permission>
<uses-permission android:name="android.permission.VIBRATE" >
</uses-permission>
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<!-- Keeps the processor from sleeping when a message is received. -->
<uses-permission android:name="com.google.android.gms.permission.ACTIVITY_RECOGNITION"
/>
<!-- Creates a custom permission so only this app can receive its messages. -->
<!-- Note: The next 2 custom permission names should begin with your application's
package name! -->
<permission android:name="YOUR.PACKAGE.NAME.permission.C2D_MESSAGE"
android:protectionLevel="signature" />
<!-- These permissions are required to enable the C2DM features of the SDK. -->
<uses-permission android:name="YOUR.PACKAGE.NAME.permission.C2D_MESSAGE" />
<uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
<!-- ****** END: GeoMoby Permissions ******* -->
```

STEP TWO: Declare services and activities

To enable the GeoMoby Service you'll need to declare the following services and activities:

```
<!-- ****** START: GeoMoby Activities ******* -->
<service android:name="com.geomoby.logic.GeomobyStartService" /> <service
android:name="com.geomoby.logic.GeomobyStopService" /> <service
android:name="com.geomoby.service.GeomobyNotifyService" /> <service
android:name="com.geomoby.service.ActivityRecognitionIntentService" />

<receiver android:name="com.geomoby.service.GeomobyNotifyServiceReceiver" > </receiver>
<!-- By default in our SDK but you can also build your own receiver -->
<receiver android:name="com.geomoby.logic.GeomobyReceiver"
android:permission="com.google.android.c2dm.permission.SEND" >
<intent-filter>
<action android:name="com.google.android.c2dm.intent.RECEIVE" />
<action android:name="com.google.android.c2dm.intent.REGISTRATION" />
<category android:name="YOUR.PACKAGE.NAME" /><!-- This should equal your application's
package name! -->
</intent-filter>
</receiver>
```

```
<!-- This alert dialog is included by default in our SDK
but you can also build your own custom notification -->
<activity android:name="com.geomoby.logic.DisplayNotification"
android:theme="@android:style/Theme.Dialog" > </activity>
<meta-data android:name="com.google.android.gms.version"
android:value="@integer/google_play_services_version/">
<!-- ****** END: GeoMoby Activities ******* -->
```

NOTE: You can define your own theme for the alert and notification dialogs. See our online documentation or refer to *STEP FOUR: Catch a push message*.

STEP THREE: Start and stop the service

You can start the GeoMoby service like starting any other Android Service.

You would need to stop the service by calling the specific activity ***GeomobyStopService*** that will ensure that the background service is cleared properly.

The code below shows how to start and stop the background service. Once started, the service will automatically start requesting location updates from the system.

```
//Start the tracking service:
startService(new Intent(MainActivity.this, GeomobyStartService.class));

//Stop the tracking service:
startService(new Intent(MainActivity.this,GeomobyStopService.class));
```

NOTE: Replace *MainActivity* by the name of the current Activity

STEP FOUR (OPTIONAL): Catch a Push Message

GeoMoby's server will return the following JSON structure:

```
[
{       "title":"A new discount for you today!",
        "link":"http://www.yourwebsite.com/product?id=123456",
        "description":"Use this coupon code on tour favorite brand",
        "imageurl":"https://www.yourwebsite.com/photos/large/72320370.jpg",
        "geofence_latitude":-33.856905,       "geofence_longitude":151.215268,
        "notification_id":963817175,
        "proimity":2,
        "beacon_uuid":"E2C56DB5-DFFB-48D2-B060-D0F5A71096E0",     "major":30654,
        "minor":34812
}
]
```

NOTE: You can choose to use our embedded alert dialog in ***com.geomoby.logic.DisplayNotification*** and play with the theme OR you might want to catch the push message straight from our server. Here is how to do it!

1. Let's get started

Add your custom **GCM broadcast receiver** and **custom notification activity** in your Android Manifest file

```
<receiver android:name="YOUR.CUSTOM.RECEIVER.HERE"
android:permission="com.google.android.c2dm.permission.SEND" >
<intent-filter>
<action android:name="com.google.android.c2dm.intent.RECEIVE" />
<action android:name="com.google.android.c2dm.intent.REGISTRATION" />
<category android:name="YOUR.PACKAGE.NAME" /><!-- This should equal your application's
package name! -->
</intent-filter>
</receiver>

<activity android:name="YOUR.CUSTOM.NOTIFICATION.ALERT.DIALOG.HERE"
android:theme="@android:style/Theme.Dialog" > </activity>
```

2. Now, let's have a look at your custom broadcast receiver

```
public class CustomReceiver extends BroadcastReceiver {
private static int notifyID = 1; // Sets an ID for the notification, so it can be
updated
private static final String TAG = "CustomReceiver";

@Override
public void onReceive(Context context, Intent intent) {
String message = intent.getExtras().getString("message");
          // Here you can catch our message and generate a local notification
          if(message != null){
          generateNotification(context, message);                    }
          setResultCode(Activity.RESULT_OK);
          }

/**
* Issues a notification to inform the user that the GeoMoby server has sent a message.
*/
@SuppressWarnings("deprecation")
private static void generateNotification(Context context, String message) {
          // Parse the GeoMoby message using our embedded GeoAlert.class and gson
          Gson gson = new Gson();
          JsonParser parser = new JsonParser();
          JsonArray Jarray = parser.parse(message).getAsJsonArray();
          ArrayList<GeoAlert> alerts = new ArrayList<GeoAlert>();
          for(JsonElement obj : Jarray ){
          GeoAlert gName = gson.fromJson(obj,GeoAlert.class);
          alerts.add(gName);
          }

          // Send Notification
          NotificationManager notificationManager =
          (NotificationManager)context.getSystemService(Context.NOTIFICATION_SERVICE);
          Intent i = new Intent(context, YourCustomNotification.class); // This is
where you will be able to customise the notification
          i.putParcelableArrayListExtra("GeoAlert", (ArrayList<GeoAlert>) alerts);
          PendingIntent pendingIntent = PendingIntent.getActivity(context,
0,i,PendingIntent.FLAG_UPDATE_CURRENT);
```

```
          // Read from the /assets directory
          int icon = resources.getIdentifier("your_icon" , "drawable",
          context.getPackageName());

          // Manage notifications differently according to Android version
          if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.HONEYCOMB) {
          NotificationCompat.Builder builder = ne NotificationCompat.Builder(context);
          builder
          .setSmallIcon(icon)
          .setContentTitle("Your Alert Title!")
          .setContentText("Click Now!")
          .setTicker("Your Alert Title!")
          .setContentIntent(pendingIntent)
          .setAutoCancel(true);

          builder.setDefaults(Notification.DEFAULT_ALL); // Vibrate, Sound and Led

        Notification(icon,"Your Alert Title!",System.currentTimeMillis());
        // Setting Notification Flags
        notification.defaults |= Notification.DEFAULT_ALL;
        notification.flags |= Notification.FLAG_AUTO_CANCEL;
          // Set the Notification Info
        notification.setLatestEventInfo(context, "Your Alert Title!", "Click Now !",
        pendingIntent);

          // Send the notification
          // Because the ID remains unchanged, the existing notification is updated.
          notificationManager.notify(notifyID, notification);
          }
}
}
```

2. Build your custom notification using an Activity

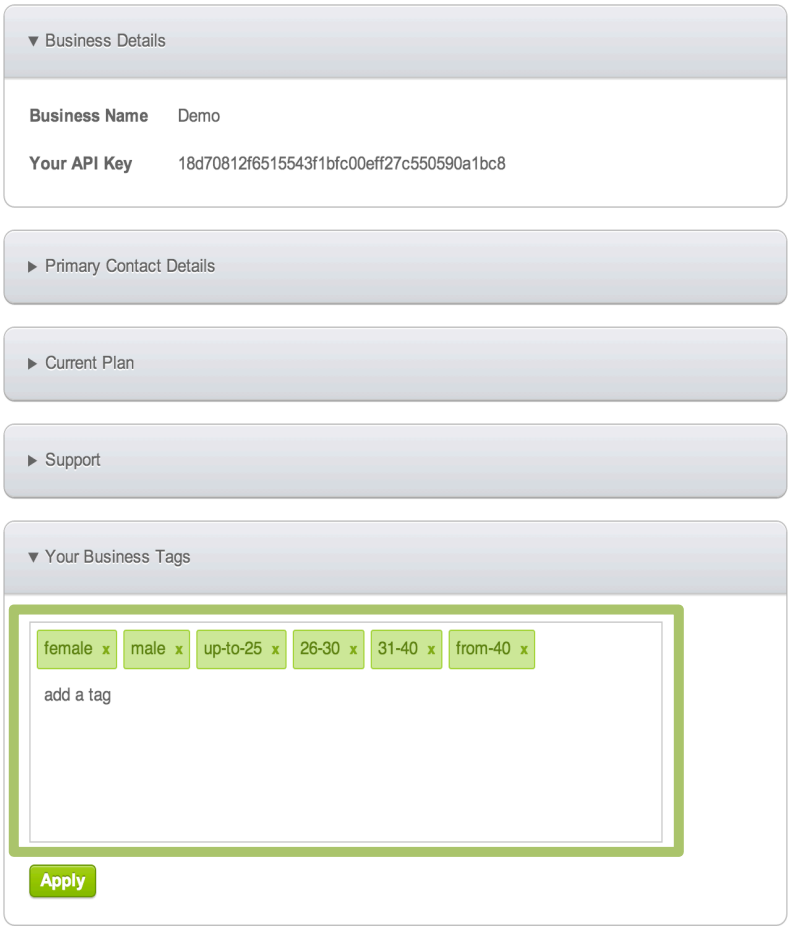
```
public class YourCustomNotification extends Activity {
          private final String SETTING_LNG="longitude";
          private final String SETTING_LAT="latitude";

@Override
protected void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          Intent intent = getIntent();

          // Fetch information from your custom GCM broadcast receiver
          ArrayList<GeoAlert> geoAlert =
          intent.getParcelableArrayListExtra("GeoAlert");
          // Get information from the GeoMoby server - VALID FROM VERSION 1.01
          String title = geoAlert.get(0).title;
          String link = geoAlert.get(0).link;
          String image_url = geoAlert.get(0).imageurl;
          String description = geoAlert.get(0).description;
          final double latitude = Double.valueOf(geoAlert.get(0).geofence_latitude);
          final double longitude = Double.valueOf(geoAlert.get(0).geofence_longitude);
          int notification_id = geoAlert.get(0).notification_id;
```

```
        // Example of how the message can be displayed in the Alert Dialog
        String message = "<head><style type='text/css'>body{margin:auto auto;text-
        align:center;} </style></head><body><b>" + description + "</b>";

        if(!"".equals(link) && link != null)
                message += "<br>Link: <a href='"+link+"'>Click for More</a>";
        if(!"".equals(image_url) && image_url != null)
                message += "<br><img src='"+image_url+"'/>";
        message+="</body>";

        AlertDialog.Builder alertDialog = new AlertDialog.Builder(this);

        // Setting Dialog Title
        alertDialog.setTitle(title);
        WebView webView = new WebView(this);
        webView.loadDataWithBaseURL(null, message, "text/html", "utf-8", null);
        webView.getSettings().setLayoutAlgorithm(LayoutAlgorithm.SINGLE_COLUMN);

        // Setting Dialog WebView
        alertDialog.setView(webView);

        // on pressing OK button
        alertDialog.setPositiveButton("OK", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int which) {
                //Notify GeoMoby server that user has opened the notification
                dialog.cancel();
                YourCustomNotification.this.finish();
                }
        });
        // Showing Alert Message
        alertDialog.show();
}
```

HOW TO USE YOUR BUSINESS TAGS

GeoMoby Android SDK allows use to define your own business tags that will be used by our platform to segment your audience when you will be creating your alerts.

NOTE: Tags are set in your account profile and must match in order to use our service properly.



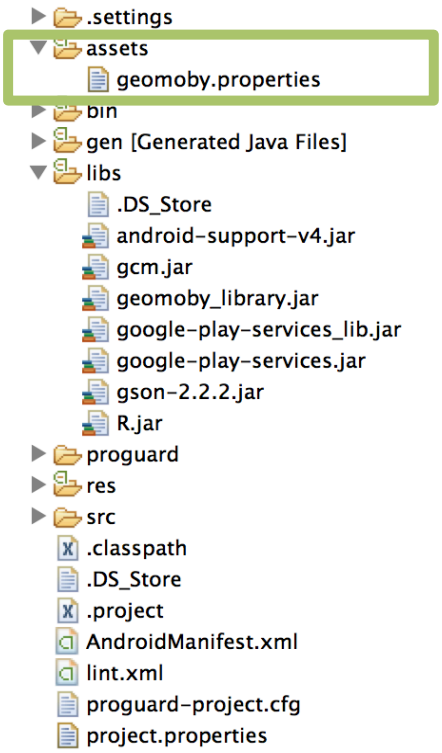

```
private static final String PREF = "GeoMobyPrefs";
[...]
// save the selection in the Shared Preferences in private mode
SharedPreferences settingsActivity = getSharedPreferences(PREF, MODE_PRIVATE);
SharedPreferences.Editor prefEditor = settingsActivity.edit();
prefEditor.putString("tags", <YOUR_STRING_COMMA_SEPARATED>);
prefEditor.commit();
```

Example:

```
prefEditor.putString("tags", “female,up-to-25,gluten free”);
```

CREDENTIALS

Before starting the service you need to provide your app credentials. The preferred method is to create a ***geomoby.properties*** file in your application's ***/assets*** directory.



The properties file should follow the format below:

```
# Your GeoMoby API Key
business_key=<YOUR_API_KEY>

# The name of the drawable to be displayed with push notifications (basically, your
logo). The image must be located in one of your res/drawable folders. # Do not add the
file extension
push_icon=geomoby_icon

# Stop GeoMoby service when Low Battery Level is reached (in %)
battery_level=10

# Filter events based on users activity (still,walking,cycling,driving,tilting -
default:walking - debug:tilting) # You can also filter several activities using '|' as
a separator (tilting|walking)
motion_filter=walking

# This setting corresponds to the minimum time interval between 2 GeoMoby location
service calls in seconds (Recommended: >30).
updateIntervalSeconds=120

# Silence Time is the time window when no notifications can be sent (24H)
silence_start=24 silence_stop=05

# Choose the geofencing mode you want to activate (yes/no) # outdoor=yes will activate
the optimised battery management for software geofencing: 7-10m outdoor and 30m indoor
# indoor=yes works in combination with iBeacons (hardware geofencing) and is 2-3
inches accurate within the first 5 meters. Range is up to 50m. # outdoor=yes and
```

```
indoor=yes means that the system will use the best mode available depending on current
environment
outdoor=yes
indoor=yes

# Turn development mode on and off (yes/no) # dev_mode=yes consumes more battery  #
dev_mode=no is the production mode as it gets the most out of our optimised battery
management
dev_mode=yes
```

HOW TO USE PROGUARD

If you are using Proguard to obfuscate and optimize your project, make sure to include the following parameters:

```
##---------------Begin: proguard configuration for GeoMoby  ----------

##---------------Begin: proguard configuration for Gson  ----------
# Gson uses generic type information stored in a class file when working with fields.
Proguard removes such information by default, so configure it to keep all of it.
-keepattributes Signature

# For using GSON @Expose annotation
-keepattributes *Annotation*

# Gson specific classes
-keep class sun.misc.Unsafe { *; }

# Application classes that will be serialized/deserialized over Gson
-keep class com.google.gson.examples.android.model.** { *; }
-keepclassmembers class fully.qualified.path.to.class$innerclass {
private ;
}
##---------------End: proguard configuration for Gson  ----------

-keep class * implements android.os.Parcelable {
public static final android.os.Parcelable$Creator *;
}
-keep class com.google.android.gms.location.ActivityRecognitionResult {
public *;
protected *;
}

-libraryjars libs/android-support-v4.jar
–libraryjars libs/gson-2.2.2.jar
-libraryjars libs/geomoby_library.jar

-dontwarn com.google.android.gms.analytics.**
-dontwarn com.google.android.gms.auth.GoogleAuthUtil
-dontwarn com.google.android.gms.common.GooglePlayServicesUtil
-dontwarn com.google.android.gms.internal.**
-dontwarn com.google.android.gms.maps.GoogleMapOptions
-dontwarn com.google.android.gms.maps.model.CameraPosition
##---------------End: proguard configuration for GeoMoby  ----------
```

LICENSE

See Our SDK License Agreement

ADDITIONAL NOTES

- Our online documentation is available here: https://www.geomoby.com/admin/doc
- Our user guide is available here: https://www.geomoby.com/admin/doc/user_guide