

TP Android

Google Maps API V2

Objectif

La création d'une application Android qui utilise les cartes Google Maps API v2 Android.

Vue d'ensemble

La création d'une application Android qui utilise les cartes Google Maps API v2 Android nécessite plusieurs étapes.

- 1°) Installez le SDK Android
- 2°) Télécharger et configurer le Google Play services SDK, qui comprend l'API Google Maps Android.
- 3°) Obtenir une clé API . Pour ce faire, vous devrez inscrire un projet dans la console API Google, et d'obtenir un certificat de signature de votre application.
- 4°) Ajoutez les paramètres requis dans le manifeste de l'application.
- 5°) Ajouter une carte pour votre application.

1°) Installez le SDK Android

Vous avez déjà fait cette installation dans les TPs précédents.

2°) Télécharger et configurer le Google Play services SDK

L'API Google Maps v2 Android est distribuée dans le cadre du SDK des services Google Play. Vous pouvez télécharger le SDK des services Google Play via le Gestionnaire SDK Android.

Si vous souhaitez que les terminaux fondés Froyo ou Ginger Bread exécute votre application, il faudra aussi télécharger « google play service avec support FROYO »

Voici un résumé des étapes pour installer et utiliser Google Play Service

- 1°) Installer le SDK des services Google Play.
- 2°) Ajouter Google Play services comme un projet de bibliothèque Android.
- 3°) Référence Google Play services dans le projet de votre application.
- 4°) Ajouter la version des services Google Play dans le manifeste de votre application

Téléchargement Google Play services

Google a fait une nouvelle API appelée « Google Maps V2 » dans le cadre des services **Google Play** du SDK. Donc, avant de commencer à élaborer des cartes nous avons besoin de télécharger les services de Google Play. Vous pouvez ouvrir le gestionnaire de SDK soit à partir d'Eclipse.

Ouvrir **Eclipse** ⇒ ⇒ **de Windows SDK Android Manager** et vérifiez si vous avez déjà téléchargé Google Play services ou non.

Si vous souhaitez que les terminaux fondés Froyo ou Ginger Bread exécute votre application, il faudra aussi télécharger « google play service avec support FROYO »

Importation Google Play services dans Eclipse

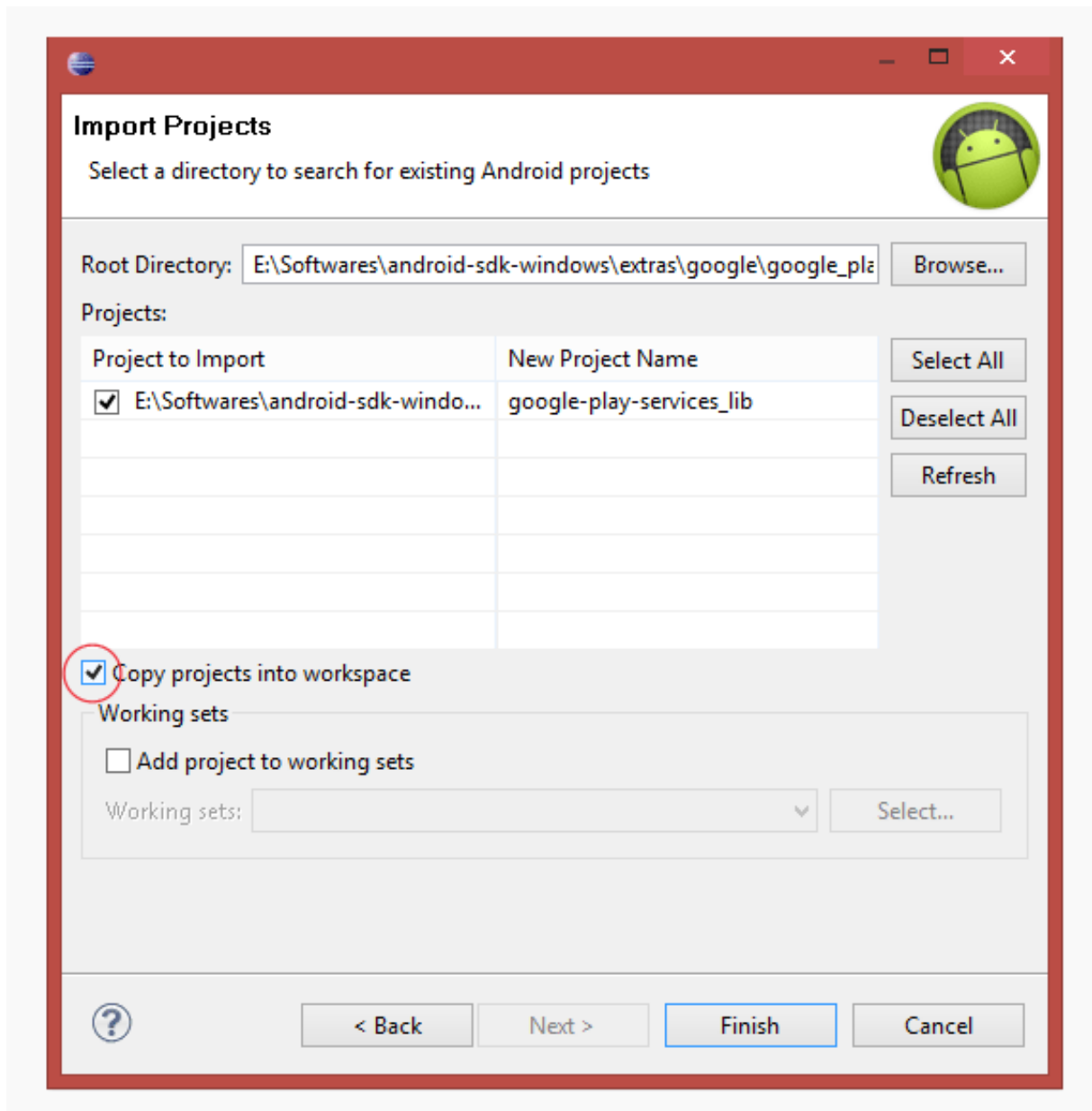
Après avoir téléchargé les services, nous devons les importer dans Eclipse pour les utiliser comme une bibliothèque pour notre projet.

1. Dans Eclipse : Fichier ⇒ ⇒ Importer Android ⇒ existing Android code dans l'espace de travail

2. Cliquez sur Parcourir et sélectionnez :

android-sdk-windows \ extras \ google \ google_play_services \ libproject \ google play-services_lib

3. Il est important lors de l'importation des projets de copier les fichiers dans l'espace de travail comme indiqué dans l'image ci-dessous.

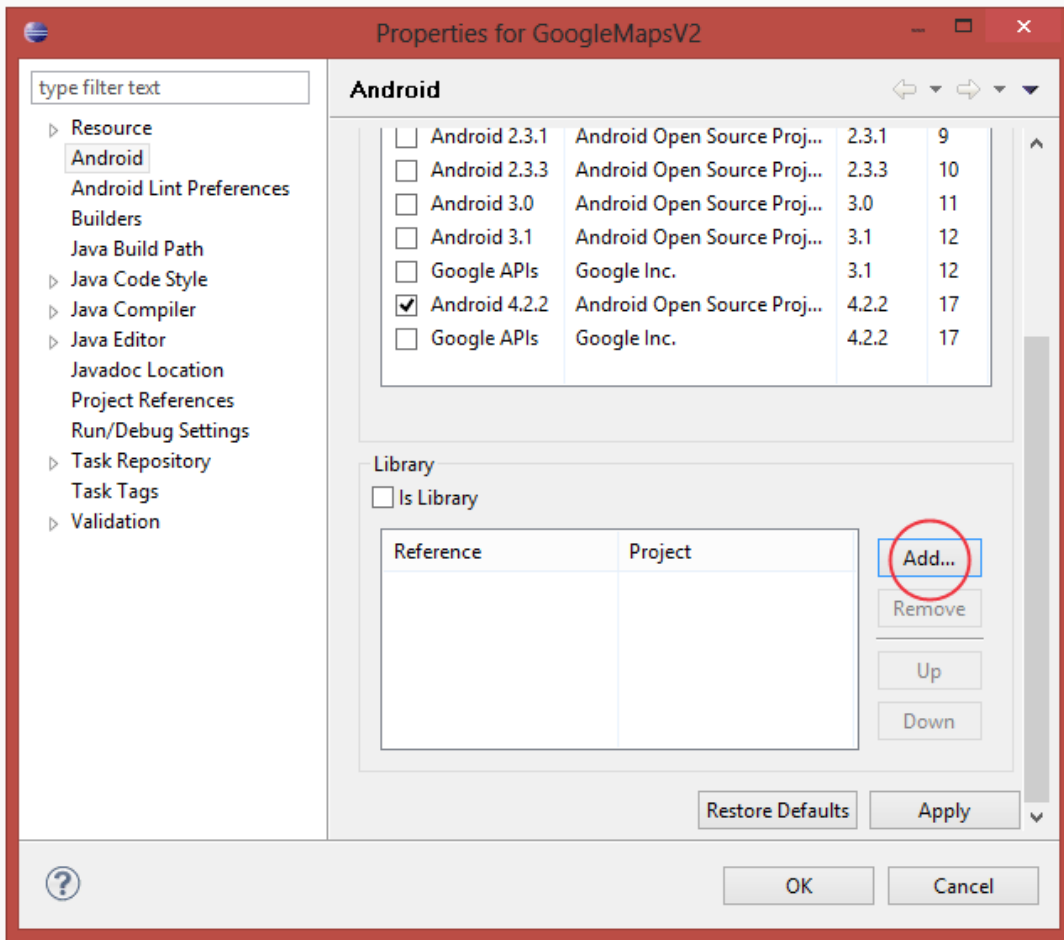


Création d'un nouveau projet

Après avoir terminé la configuration requise, il est temps de commencer notre projet.

1. Dans Eclipse créer un nouveau projet en cliquant sur Fichier ⇒ ⇒ Nouveau projet d'application Android et remplir les détails requis.

Linking Google Play Services to Project



The screenshot shows the 'Properties for GoogleMapsV2' dialog box. The 'Android' tab is active, displaying a list of Android versions. The 'Android 4.2.2' version is selected with a checkmark. Below the list is a 'Library' section with an 'Add...' button circled in red. The 'Add...' button is used to link Google Play Services to the project.

Version	Provider	Level	Count
<input type="checkbox"/> Android 2.3.1	Android Open Source Proj...	2.3.1	9
<input type="checkbox"/> Android 2.3.3	Android Open Source Proj...	2.3.3	10
<input type="checkbox"/> Android 3.0	Android Open Source Proj...	3.0	11
<input type="checkbox"/> Android 3.1	Android Open Source Proj...	3.1	12
<input type="checkbox"/> Google APIs	Google Inc.	3.1	12
<input checked="" type="checkbox"/> Android 4.2.2	Android Open Source Proj...	4.2.2	17
<input type="checkbox"/> Google APIs	Google Inc.	4.2.2	17

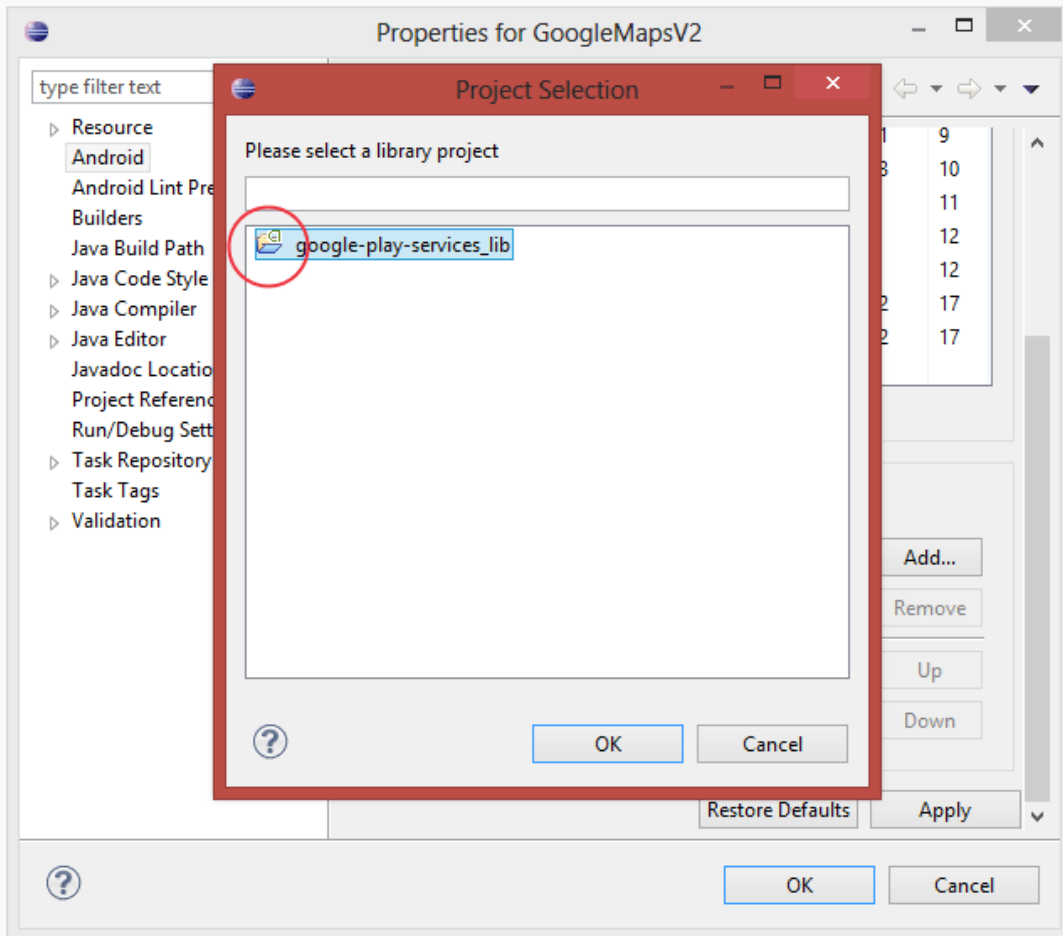
Library

Is Library

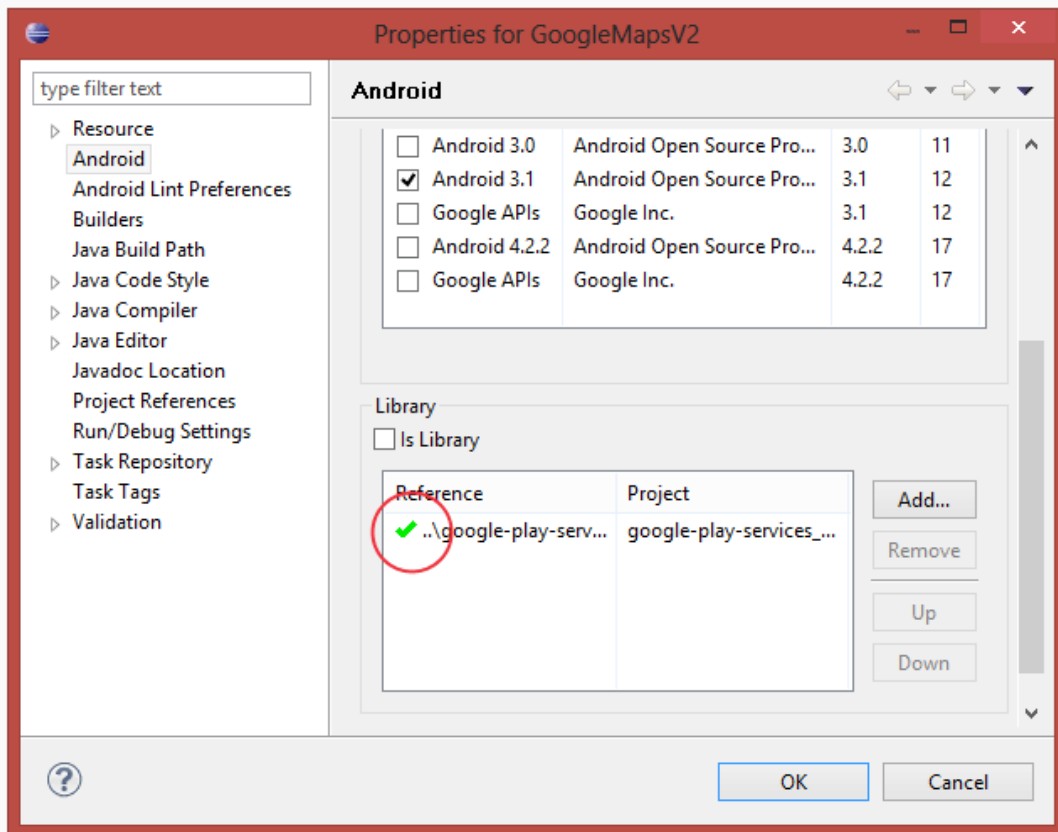
Reference	Project
-----------	---------

Buttons: Add... (circled in red), Remove, Up, Down, Restore Defaults, Apply, OK, Cancel

Linking Google Play Services to Project



Linking Google Play Services to Project



3. Ajouter la clé de la carte dans le fichier manifeste. Ouvrez AndroidManifest.xml et ajoutez le code suivant avant la fin du tag APPLICATION. Nous verrons ensuite comment obtenir une clé pour utiliser le service Google Maps.

```
<!-- Cela permet d'inclure la version des services Google Play. />
<meta-data
    android:name="com.google.android.gms.version"
    android:value="@integer/google_play_services_version" />
```

```
<!-- Google Maps API Key -->
<meta-data
    android:name="com.google.android.maps.v2.API_KEY"
    android:value="mettre ici votre clé pour votre carte" />
```

4. Les Google maps ont besoin des autorisations et des fonctionnalités suivantes :

ACCESS_NETWORK_STATE - Pour vérifier l'état du réseau si les données peuvent être téléchargées ou pas
INTERNET - Pour vérifier l'état de la connexion Internet
WRITE_EXTERNAL_STORAGE - Pour écrire dans la mémoire externe
ACCESS_COARSE_LOCATION - Pour déterminer l'emplacement de l'utilisateur en utilisant le WiFi et données mobile cellulaire

ACCESS_FINE_LOCATION - Pour déterminer l'emplacement de l'utilisateur en utilisant le GPS

OpenGL ES V2 - Nécessaire pour Google Maps V2

Il faut placer toutes ces permissions avant le début de la balise « APPLICATION »

```
<permission
    android:name="mon package.permission.MAPS_RECEIVE"
    android:protectionLevel="signature" />

    <uses-permission android:name="mon package.permission.MAPS_RECEIVE" />

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

    <!-- Required to show current location -->
    <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
/>

    <!-- Required OpenGL ES 2.0. for Maps V2 -->
    <uses-feature
        android:glEsVersion="0x00020000"
        android:required="true" />
```

4. Obtenir une clé pour les API google maps v2

Nous devons générer l'empreinte SHA-1 en utilisant Java keytool.

Ouvrez votre terminal et exécutez la commande suivante pour générer l'empreinte SHA-1.

1. L'empreinte SHA-1

Sous Windows

```
keytool -list -v -keystore "%USERPROFILE%\android\debug.keystore" -alias
androiddebugkey -storepass android -keypass android
```

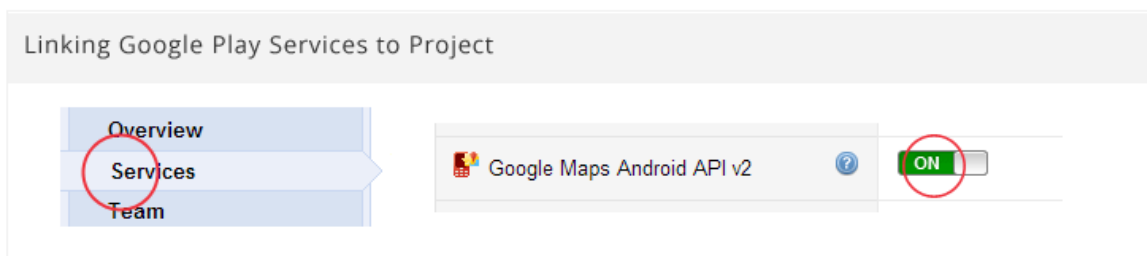
Sous Linux ou Mac OS

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

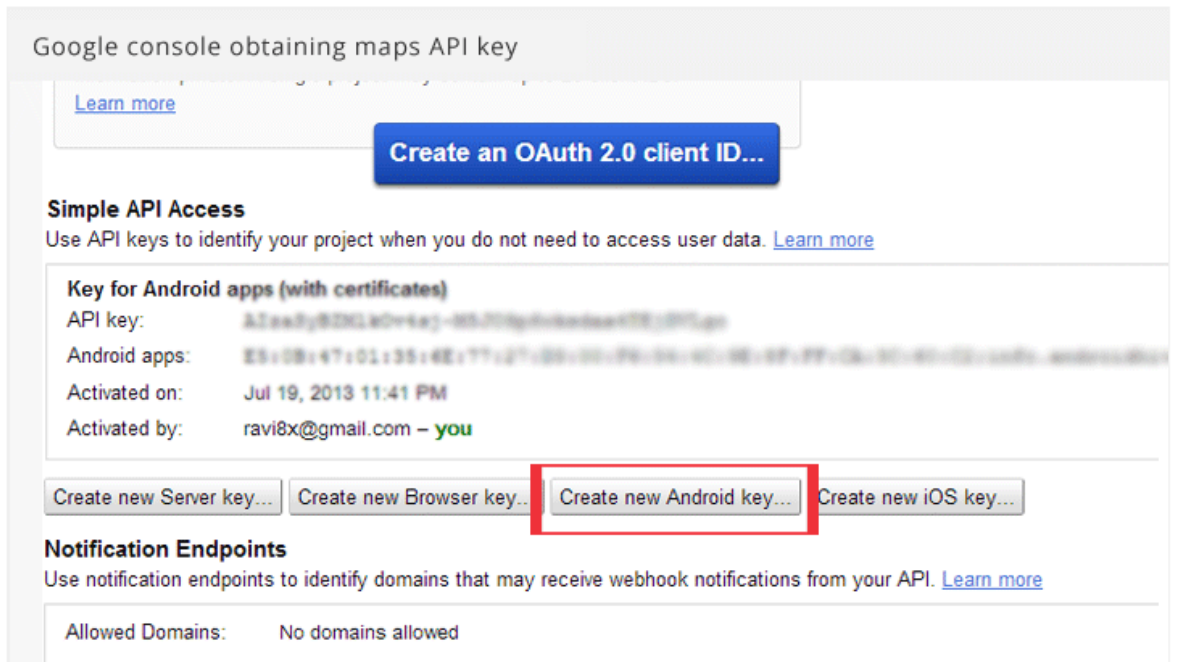
```
Generating SHA 1 fingerprint using keytool
Alias name: androiddebugkey
Creation date: May 13, 2013
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=Android Debug, O=Android, C=US
Issuer: CN=Android Debug, O=Android, C=US
Serial number: 6f3eb719
Valid from: Mon May 13 02:13:08 IST 2013 until: Wed May 06 02:13:08 IST 2043
Certificate fingerprints:
  MD5: 7F:21:AF:F4:0B:67:4F:88:12:90:54:69:5E:6D:BE:DE
  SHA1: E5:0B:47:01:35:6E:77:27:D3:00:F6:54:4C:9E:8F:FF:CA:3C:60:C2
  SHA256: 3E:23:20:84:9A:77:66:52:1E:1B:E2:D5:EB:13:BE:35:FA:A8:9F:B3:86:97:F6:15:E
:13:EE:AD:F9
Signature algorithm name: SHA256withRSA
Version: 3
```

2 . Maintenant ouvrez la console Google API via l'URL suivante (dans votre navigateur)
<https://code.google.com/apis/console/>

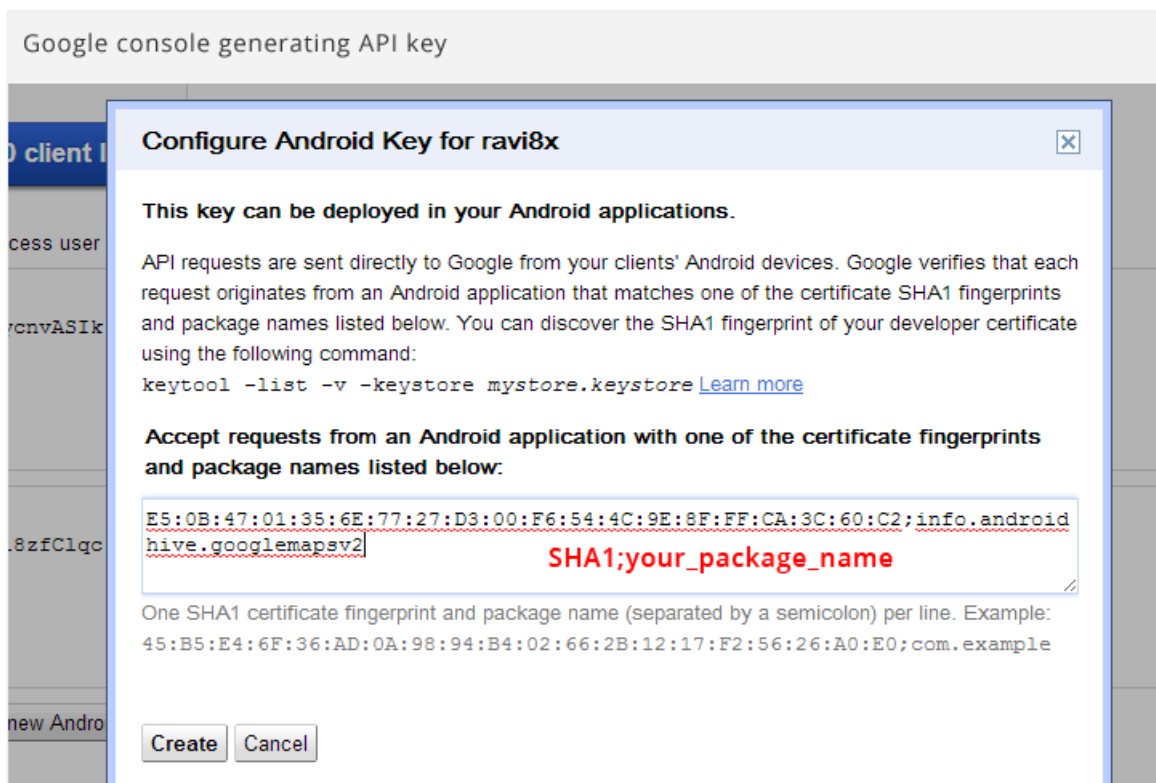
3 . Sélectionner le service Google Maps Android API v2



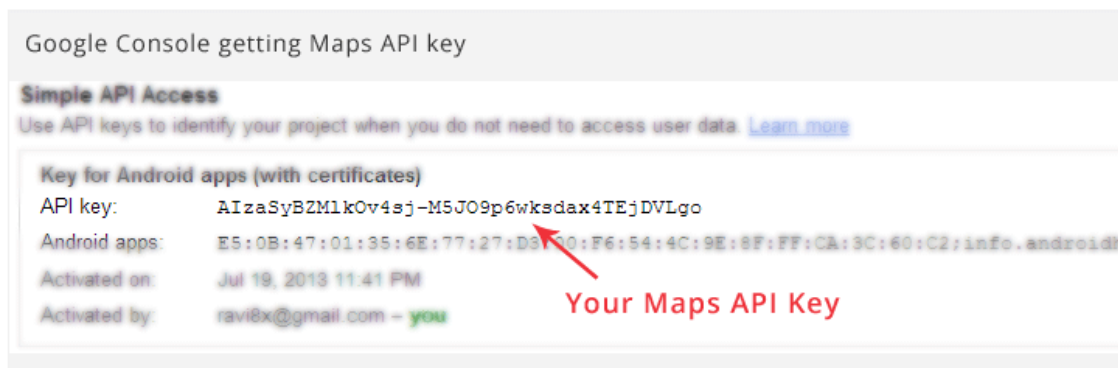
4 . Créer une clé API en cliquant sur « Create new Android key »...



Entrer votre code SHA-1 et le nom de package de votre application séparé par un point virgule.



Copier / Coller la clé dans votre application (dans le fichier manifest)



5. De retour au code

La layout :

Rajouter les lignes suivantes dans votre fichier activity_main.xml.

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

Le code JAVA

Dans votre fichier MainActivity.java, il suffit de charger le fichier activity_main.xml. La méthode `initilizeMap()` permet simplement de vérifier si l'objet `MapFragment` est correctement chargé.

```
public class MainActivity extends Activity {

    // Google Map
    private GoogleMap googleMap;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        try {
            // Loading map
            initilizeMap();

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```

}

/**
 * function to load map. If map is not created it will create it for you
 */
private void initilizeMap() {
    if (googleMap == null) {
        googleMap = ((FragmentManager) getFragmentManager().findFragmentById(
            R.id.map)).getMap();

        // check if map is created successfully or not
        if (googleMap == null) {
            Toast.makeText(getApplicationContext(),
                "Sorry! unable to create maps", Toast.LENGTH_SHORT)
                .show();
        }
    }
}

@Override
protected void onResume() {
    super.onResume();
    initilizeMap();
}
}

```

Exécution

Lancer l'application sur un vrai terminal avec un accès à Internet.

Placer un Marker

```

// latitude and longitude
double latitude = ;
double longitude = ;

// create marker
MarkerOptions marker = new MarkerOptions().position(new LatLng(latitude,
longitude)).title("Hello Maps ");

// adding marker
googleMap.addMarker(marker);

```

Une alernative

```
LatLng sydney = new LatLng(-33.867, 151.206);

googleMap.setMyLocationEnabled(true);
googleMap.addMarker(new MarkerOptions()
    .title("Sydney")
    .snippet("The most populous city in Australia.")
    .position(sydney));
```

Remarque :

Ce tutorial est une fusion des deux tutoriels suivants :

https://developers.google.com/maps/documentation/android/start#getting_the_google_maps_android_api_v2

<http://www.androidhive.info/2013/08/android-working-with-google-maps-v2/>