

GOOGLE ANDROID

Localisation de points d'intérêts



Sommaire

I.	Préparation de l'environnement de développement	2
1.	Installation du SDK.....	2
2.	Ajout du répertoire du SDK dans la variable d'environnement PATH.....	2
3.	Plugin Eclipse	2
II.	SDK	3
1.	Lancement et utilisation du SDK	3
2.	Se procurer une clé pour utiliser l'API Google Map.....	4
3.	Entrer une géolocalisation	7
III.	Quelques conseils	10
1.	Terminate	10
2.	Log cat	10
IV.	Hierarchie d'un projet	11
V.	Hello world	14
VI.	Désinstallation d'une application.....	19

I. PREPARATION DE L'ENVIRONNEMENT DE DEVELOPPEMENT

1. Installation du SDK

Étant donné qu'Android est basé sur le langage Java, il faut tout d'abord installer le JDK 6 sur le site de Sun.

<http://java.sun.com/javase/downloads/widget/jdk6.jsp>

Ensuite, on peut télécharger le SDK de Google Android.

<http://code.google.com/android/download.html>

On décompresse l'archive. Le dossier s'appellera :

`android_sdk_windows_<releases>_<build>`

On y fera référence sous la dénomination `<répertoire_sdk>`.

2. Ajout du répertoire du SDK dans la variable d'environnement PATH

- Clic droit sur le **Poste de travail** et sélectionnez **Propriétés**
- Dans l'onglet **Avancé** cliquez sur le bouton **Variables d'environnement**
- Une boîte de dialogue apparaît, double-cliquez sur l'entrée **Path** présente dans la partie **Variables Systèmes**. On y ajoute le chemin `<répertoire_sdk>\tools`.

3. Plugin Eclipse

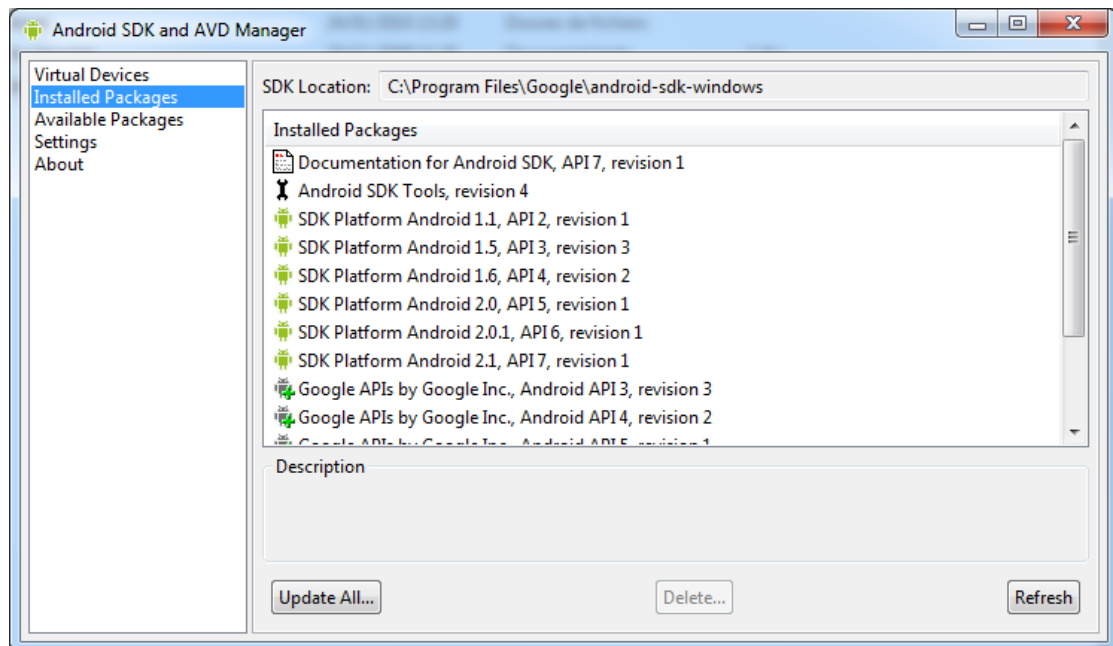
- Démarrez Eclipse puis sélectionnez le menu **Help > Software Updates > Find and Install...**
- Sélectionnez **Search for new features to install** et appuyez sur le bouton **Next**
- Appuyez sur le bouton **New Remote Site**
- Dans la boîte de dialogue qui apparaît, indiquez un nom (par exemple Android Plugin) et l'URL <https://dl-ssl.google.com/android/eclipse/>
- Appuyez sur le bouton **OK**
- Le site est ajouté à la liste et sélectionné. Appuyez sur le bouton **Finish**
- Sélectionnez **Android Plugin** et appuyez sur le bouton **Next**
- Lisez la licence d'utilisation, sélectionnez l'option **I accept the terms in the license agreement** et cliquez sur le bouton **Next**, puis sur le bouton **Finish**
- Le plugin n'est pas signé, un message vous le signale, appuyez sur le bouton **Install All**
- Redémarrez **Eclipse**
- Une fois Eclipse redémarré, sélectionnez le menu **Window > Preferences...**
- Sélectionnez **Android** dans le panel de gauche
- Indiquez le chemin où vous avez installé le SDK Android (bouton **Browse** pour parcourir le système de fichier)
- Appuyez sur le bouton **OK**



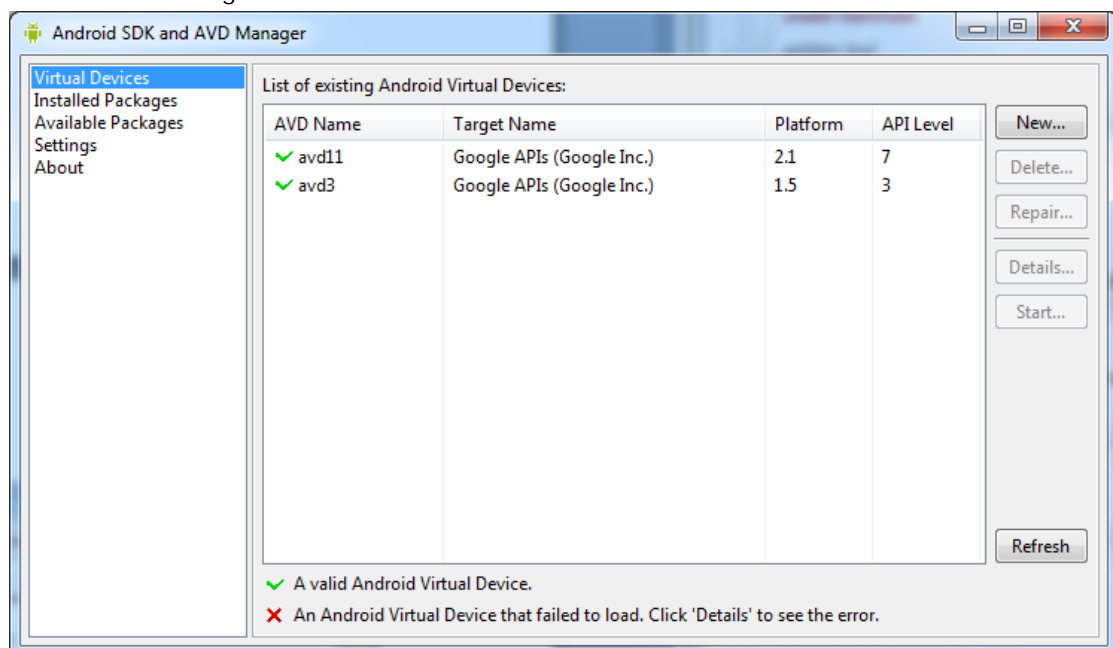
II. SDK

1. Lancement et utilisation du SDK

Lancer le SDK. On arrive sur cette page :

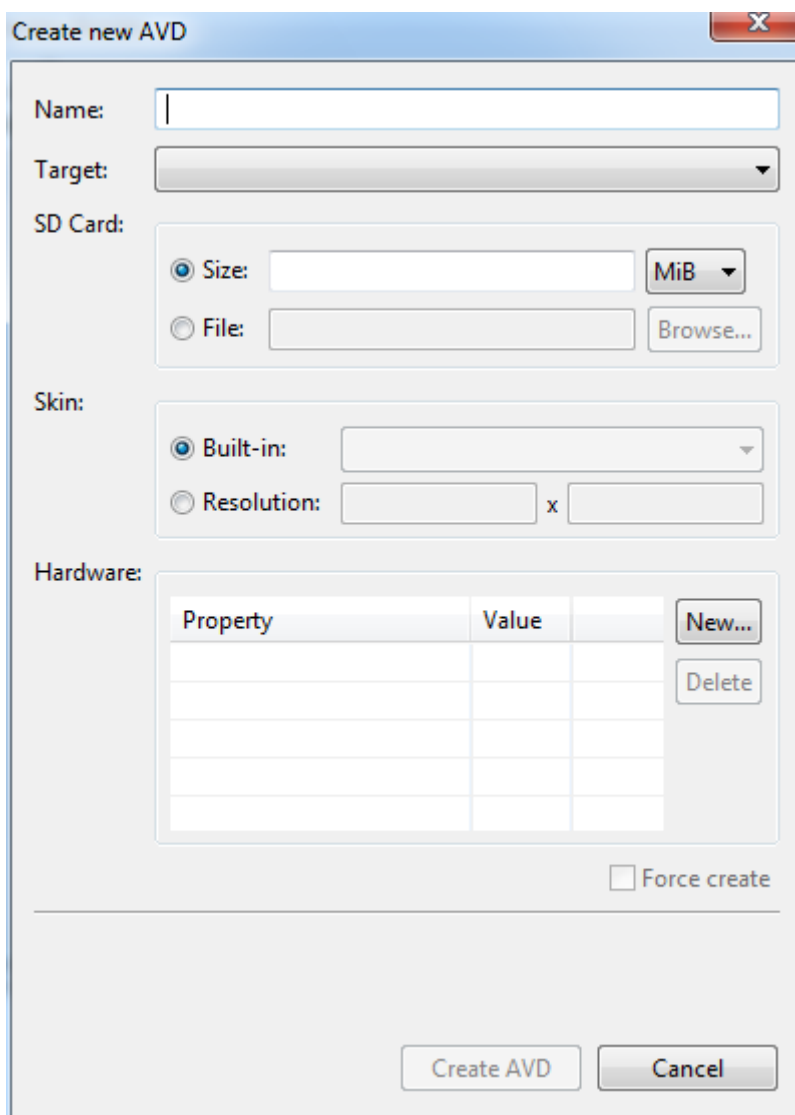


On sélectionne l'onglet « Virtual Devices » afin de créer une nouvelle cible.



On clique sur « New »

On arrive sur cette fenêtre :

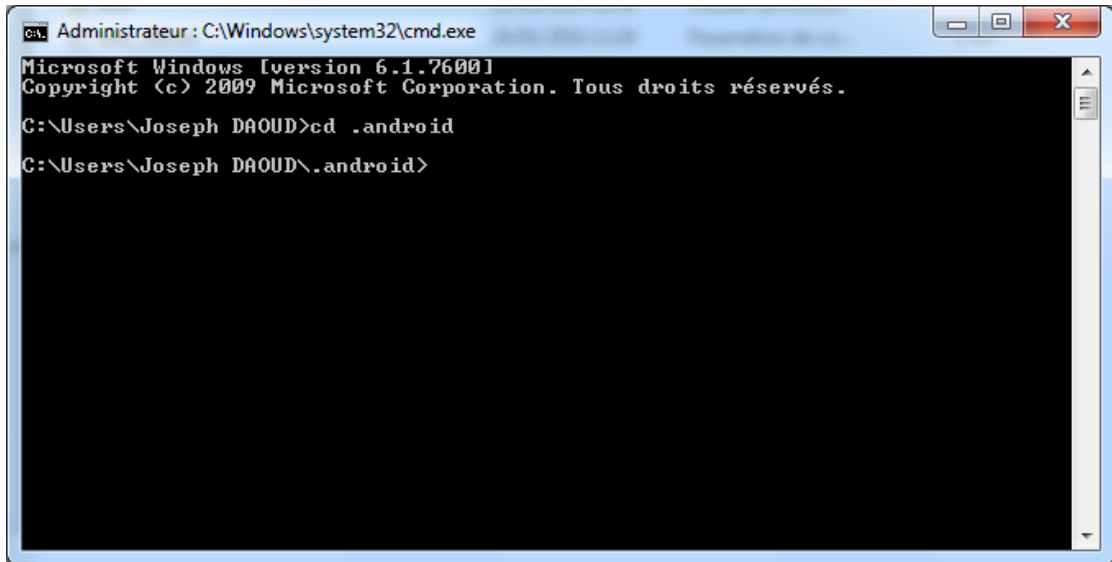


On choisit un nom pour notre émulateur ensuite on choisit la version du firmware. On a également la possibilité de choisir de taille pour la carte mémoire externe SD. On clique ensuite sur Create AVD et voilà, on a créé notre émulateur.

2. Se procurer une clé pour utiliser l'API Google Map

Pour se procurer une clé Google Map. Il faut lancer tout d'abord le terminal. On se place dans le bon dossier qui est celui du dossier tools du SDK.





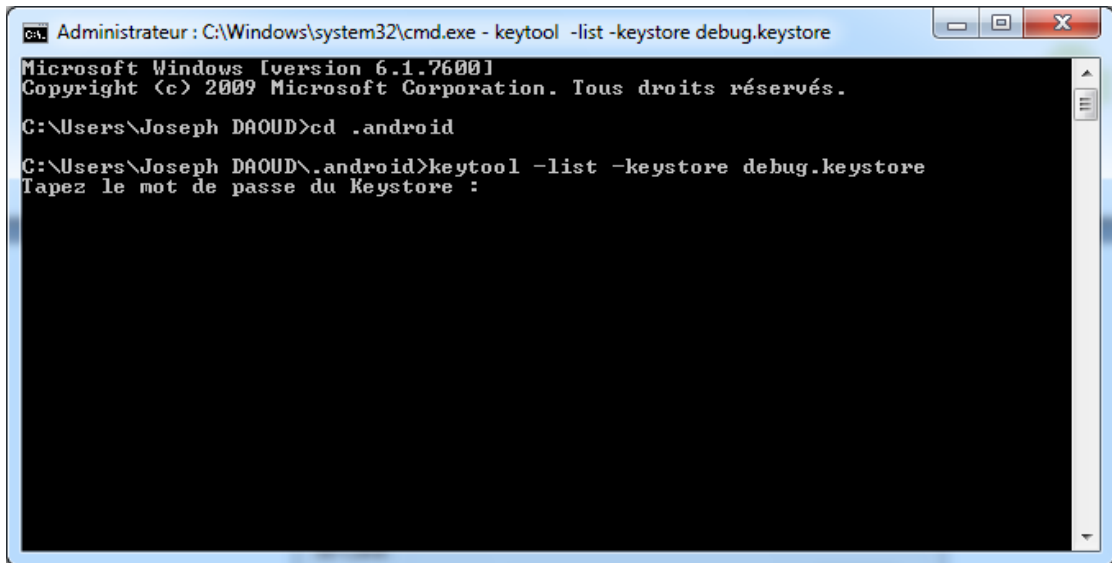
```
Administrateur : C:\Windows\system32\cmd.exe
Microsoft Windows [version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\Joseph DAOUD>cd .android
C:\Users\Joseph DAOUD\.android>
```

Ensuite, on lance la ligne de commande

```
keytool -list -keystore debug.keystore
```

On nous demande d'entrer notre mot de passe. On tape juste entrée sans rien entrer.



```
Administrateur : C:\Windows\system32\cmd.exe - keytool -list -keystore debug.keystore
Microsoft Windows [version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\Joseph DAOUD>cd .android
C:\Users\Joseph DAOUD\.android>keytool -list -keystore debug.keystore
Tapez le mot de passe du Keystore :
```

On nous délivre ensuite un code.

```
Administrateur : C:\Windows\system32\cmd.exe
Microsoft Windows [version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\Joseph DAOUD>cd .android
C:\Users\Joseph DAOUD\.android>keytool -list -keystore debug.keystore
Tapez le mot de passe du Keystore :

***** A U E R T I S S E M E N T *****
* L'intégrité des informations enregistrées dans votre Keystore *
* n'a PAS été vérifiée ! Pour cela, *
* vous devez spécifier le mot de passe de votre Keystore. *
***** A U E R T I S S E M E N T *****

Type Keystore : JKS
Fournisseur Keystore : SUN

Votre Keystore contient 1 entrée(s)

androiddebugkey, 24 févr. 2010, PrivateKeyEntry,
Empreinte du certificat (MD5) : CB:2A:64:E7:FA:EF:93:D4:60:6C:07:BF:9E:F6:ED:95

C:\Users\Joseph DAOUD\.android>
```

Maintenant, copiez ce code et il faut se rendre sur le site :

<http://code.google.com/intl/fr-FR/android/maps-api-signup.html>

The image shows a Google search page with the search bar containing the URL from the previous block. Below the search bar, the "Android Maps API Key Signup" page is displayed. The page title is "Sign Up for the Android Maps API". It contains a code block with the following content:

```
$ 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
```

Below the code block, there is a section for "Android Maps APIs Terms of Service" with a scrollable text area. At the bottom, there is a checkbox labeled "I have read and agree with the terms and conditions (printable version)" which is checked. Below the checkbox, there is a text input field containing the MD5 fingerprint: "94:1E:43:49:87:73:BB:E6:A6:88:D7:20:F1:8E:B5:98". At the bottom right of this section is a button labeled "Generate API Key".

On entre notre code, on coche et on clique sur « Generate API Key ».

On nous donne alors le code à utiliser :



Merci de vous être inscrit pour obtenir une clé API Android Maps!

Voici votre clé :

```
0n41GJgq1uwNq3ewH_x6CEA8Qi10SBBuA69SgUQ
```

Cette clé fonctionne avec les applications signées avec votre certificat ; l'empreinte digitale de celui-ci est :

```
94:1E:43:49:87:73:BB:E6:A6:88:D7:20:F1:8E:B5:98
```

Voici un exemple de code xml qui vous aidera à commencer à exploiter efficacement nos outils cartographiques :

```
<com.google.android.maps.MapView
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:apiKey="0n41GJgq1uwNq3ewH_x6CEA8Qi10SBBuA69SgUQ"
  />
```

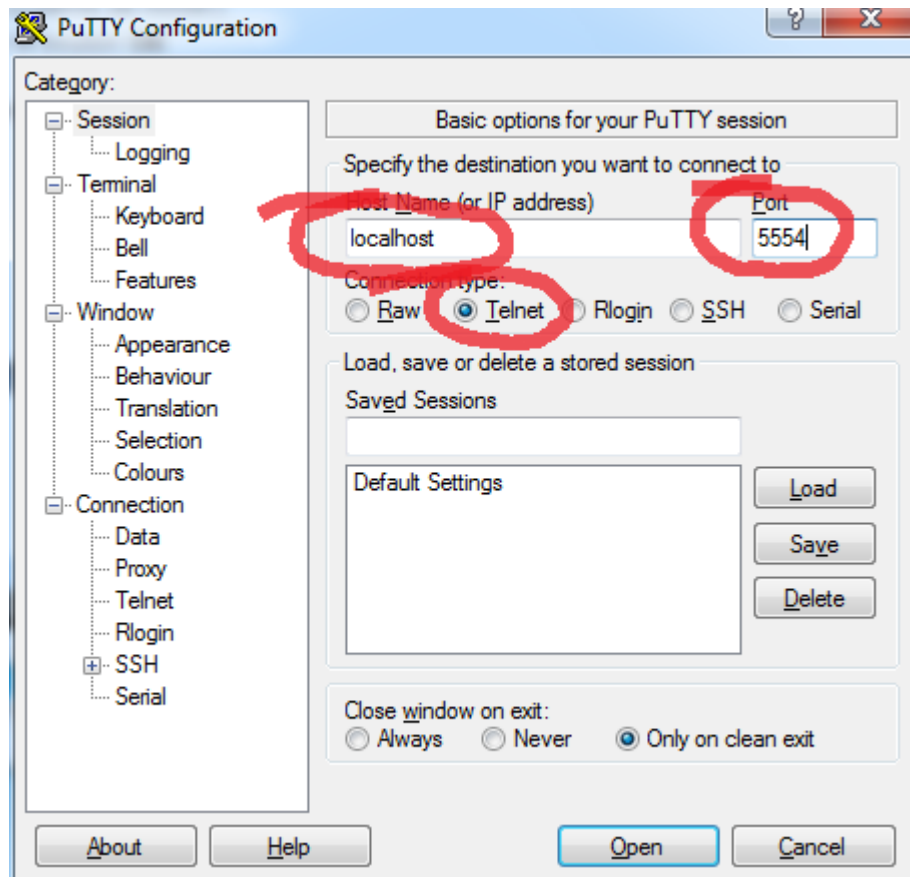
Consultez la [documentation de l'API](#) pour plus d'informations.

3. Entrer une géolocalisation

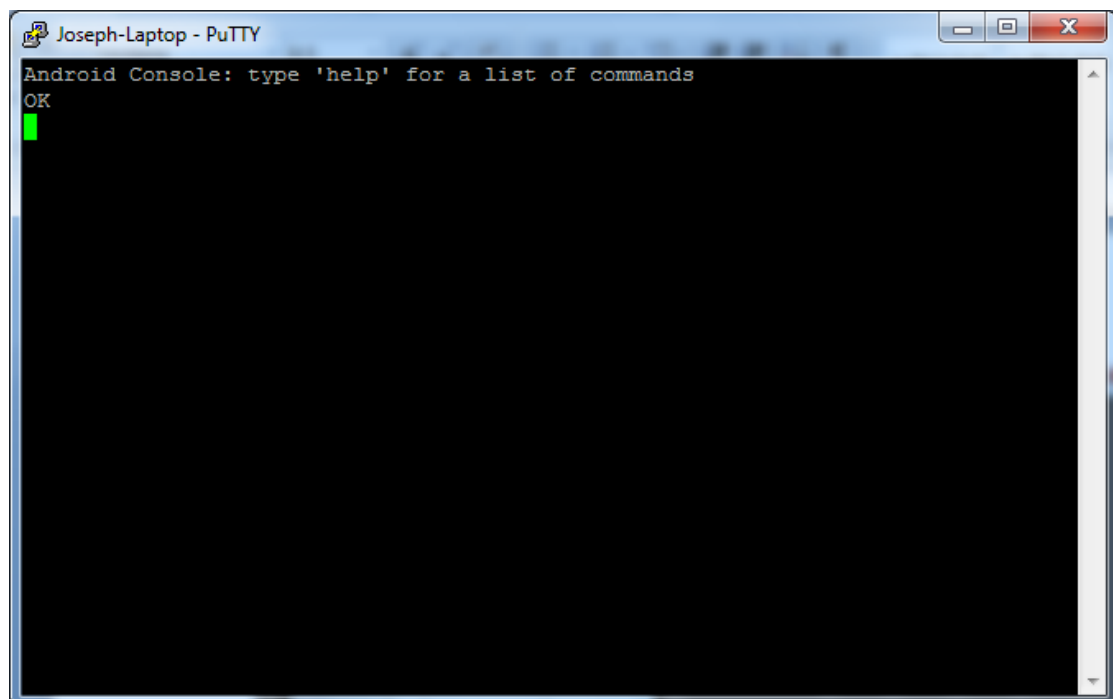
Pour cela, on repère tout d'abord le port de notre émulateur. Par défaut, c'est 5554.



On lance ensuite un TelNet et on entre le port 5554. L'hôte est localhost.



On arrive sur cette fenetre.



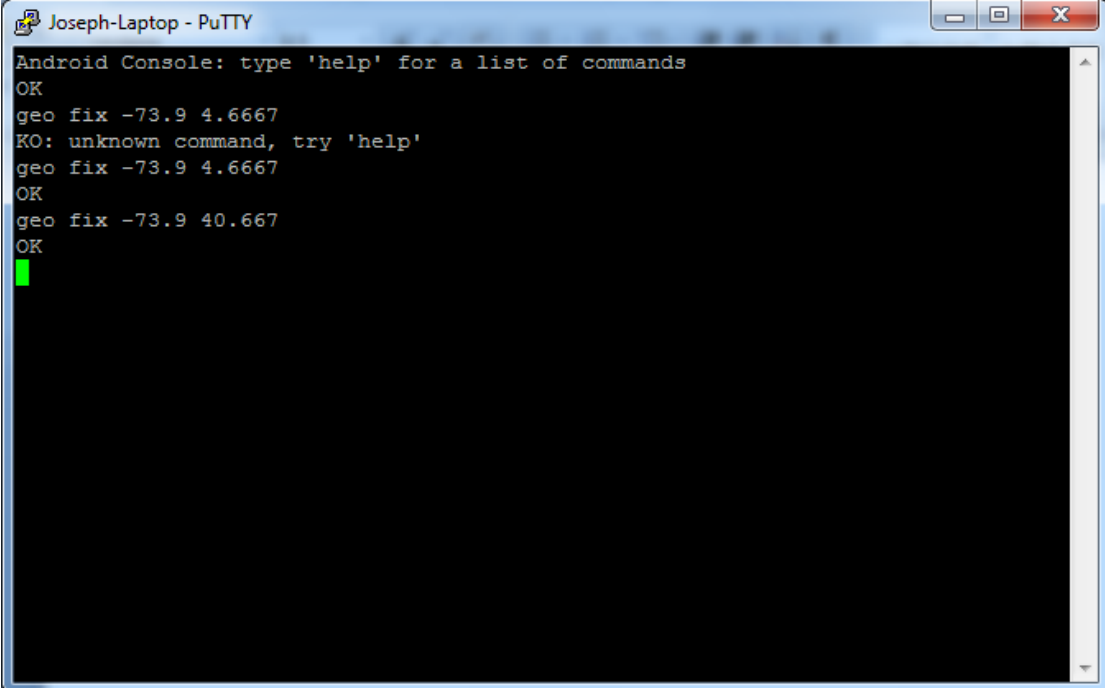
Il suffit ensuite de taper la commande :

```
Geo fix <latitude> <longitude>
```

Par exemple, si l'on rentre les coordonnées de New York City :

```
Geo fix -73.9 40.667
```

On valide (il est normal qu'il nous donne un KO la première fois, après c'est OK).



```
Joseph-Laptop - PuTTY
Android Console: type 'help' for a list of commands
OK
geo fix -73.9 4.6667
KO: unknown command, try 'help'
geo fix -73.9 4.6667
OK
geo fix -73.9 40.667
OK
█
```

Si l'on se rend ensuite dans Google Maps, on voit que l'on est bien à New York City.

III. QUELQUES CONSEILS

1. Terminate

Il se peut que lorsque l'application plante en mode Debug, aucune erreur ne soit renvoyé, il faut alors aller dans la vue Debug et faire un terminate. Alors l'erreur apparaîtra.

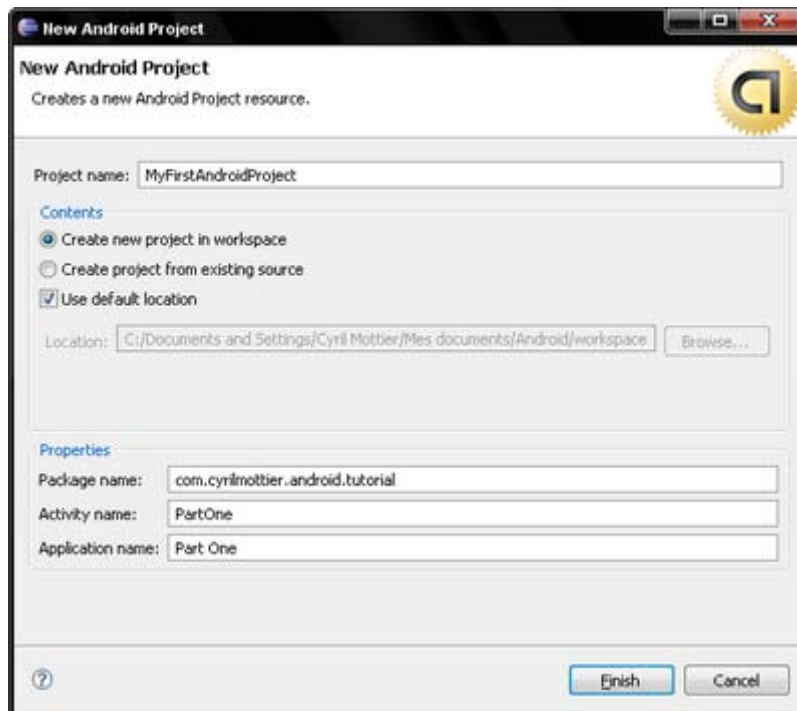
2. Log cat

Le log cat est l'équivalent du `System.out.println` du Java classique. A utiliser pour déboguer.



IV. HIERARCHIE D'UN PROJET

Allez dans **File > New > Other > Android Project**, une boîte de dialogue apparaît :



Il est nécessaire de remplir tous les champs :

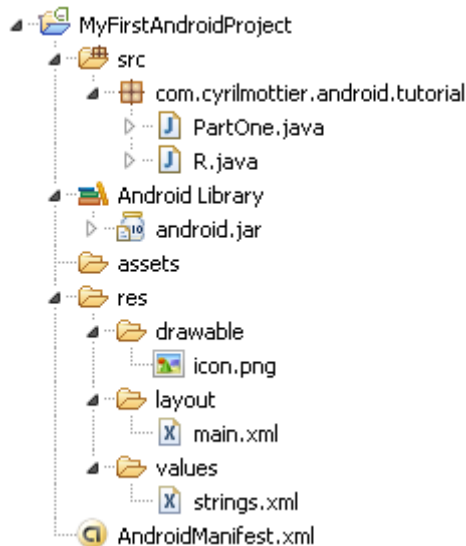
Project Name : Ce nom de projet est totalement indépendant de votre application Android. Ce n'est en fait que le nom donné à votre projet dans Eclipse (nom donné au dossier dans votre workspace).

Package Name : Nom du package principal de l'application. Le package définira en fait le nom interne de l'application sur le terminal Android.

Activity Name : C'est le nom de l'activité qui débutera au lancement de votre application. Le nom d'activité sera automatiquement ajouté au **AndroidManifest.xml** (fichier qui décrit l'application au système Android et qui sera expliqué plus en détail un peu plus loin)

Application Name : Il faut rentrer un nom pour l'application. Cette chaîne de caractère sera également inscrite dans AndroidManifest.xml et sera en fait utilisée par le système lors de l'affichage des applications dans l'écran d'accueil. C'est en réalité le nom de l'application affichée sous l'icône de l'application.

Après avoir validé la boîte de dialogue précédente en cliquant sur "Finish", on se retrouve maintenant en présence d'un projet type Android. Ce projet suit une arborescence bien précise :



Comme à l'accoutumée, Android ne déroge pas à la règle et insère donc les différents packages dans le dossier src. C'est dans ce dossier que l'ensemble des fichiers de code (.java) doivent se trouver. Il est bien sûr possible de créer de nouveaux packages comme les développeurs Java ont l'habitude de le faire. Analysons maintenant le contenu du package com.cyrilmottier.android.tutorial. On y retrouve tout d'abord un fichier PartOne.java qui n'est autre que l'activité (Activity) principale de l'application. Un simple coup d'oeil au code contenu dans le fichier montre que PartOne hérite bien d'Activity :

```
package com.cyrilmottier.android.tutorial;
```

```
import android.app.Activity;
import android.os.Bundle;
```

```
public class PartOne extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

En dessous du dossier src, on retrouve un fichier **android.jar**. C'est tout simplement ce jar qui contient l'ensemble du framework Android. Sans ce dernier, notre projet Android serait tout simplement une coquille vide.

On retrouve ensuite dans le package par défaut le fameux R.java. Ce fichier est automatiquement géré par ADT et n'est donc pas modifiable par l'utilisateur. Ce fichier contient en fait l'ensemble des références vers les ressources de votre projet. Imaginons par exemple que nous souhaitions utiliser une image (Drawable dans le langage Android), on y accèdera par R.drawable.nom_de_l_image.

Au même niveau que le dossier src, on retrouve un dossier nommé assets et qui contient simplement des données qui seront chargées sur le mobile lors de la compilation de votre projet. Les données incluses dans ce dossier ne sont normalement pas en accord avec l'arborescence classique d'un projet Android : cela peut, par exemple, être des fichiers texte décrivant la licence de votre application, des fichiers audio ou vidéo, etc.

Un projet Android contient enfin un dossier res qui regroupe l'ensemble des ressources relatives au projet. C'est ce dossier qui est "lu" par ADT pour créer le fameux R.java. Ce dossier contient lui même des sous-dossiers regroupant les ressources selon leur type. A la création d'un projet Android, les sous-dossiers créés sont :

drawable : regroupe l'ensemble des images (png, jpg, gif), Drawables, etc.

layout : le framework Android a l'avantage d'offrir une technique de création d'interfaces graphiques à l'aide de fichiers XML. C'est dans ce dossier que l'on inclue l'ensemble des fichiers décrivant l'interface.

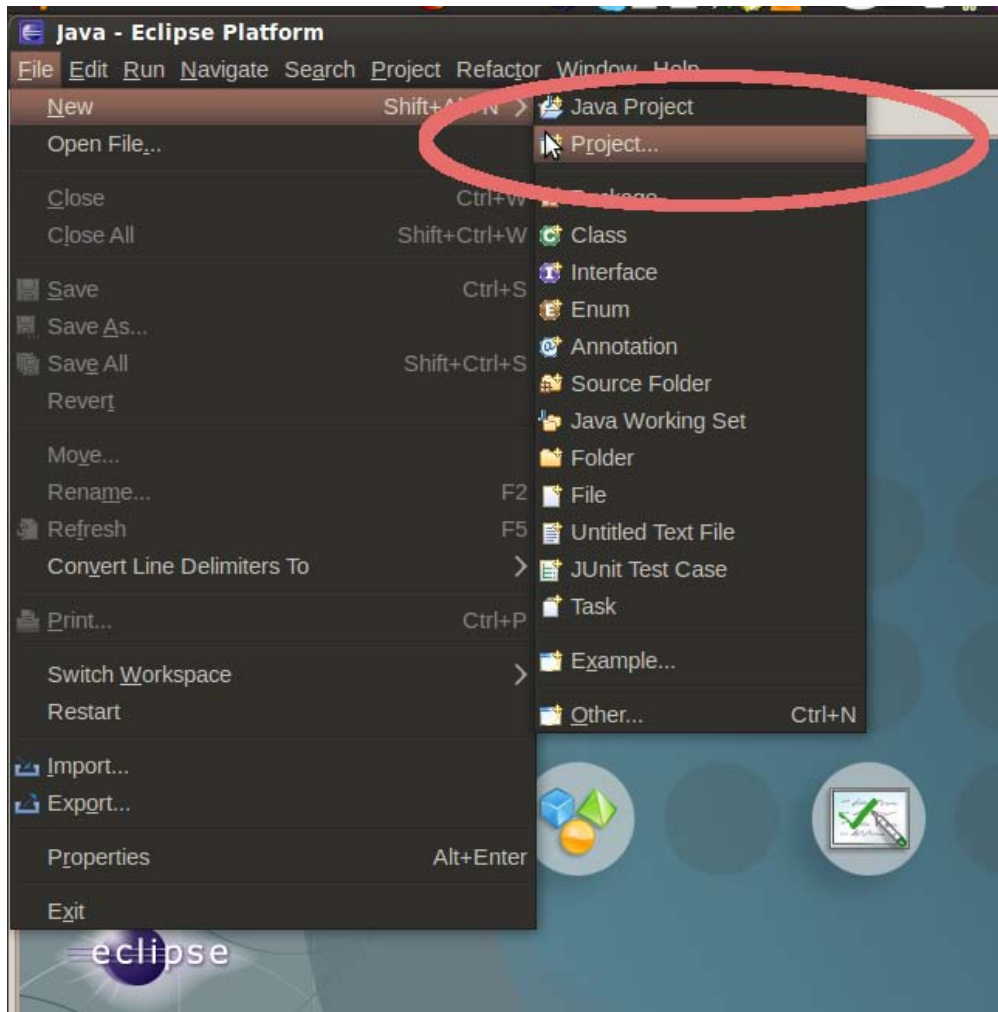
values : ce dossier contient également un ensemble de fichiers décrivant les valeurs utilisées par votre application. On peut, par exemple, y mettre des chaînes de caractères (strings.xml), des tableaux (arrays.xml), des couleurs, des dimensions, etc...

Pour finir, tout projet Android contient un fichier nommé **AndroidManifest.xml** qui définit le comportement de l'application au système Android. Ce fichier définit par exemple, le nom, l'icône (par défaut drawable/icon.png), le thème, la version minimale du système nécessaire à l'exécution de l'application, les activités, les services, etc. de l'application.

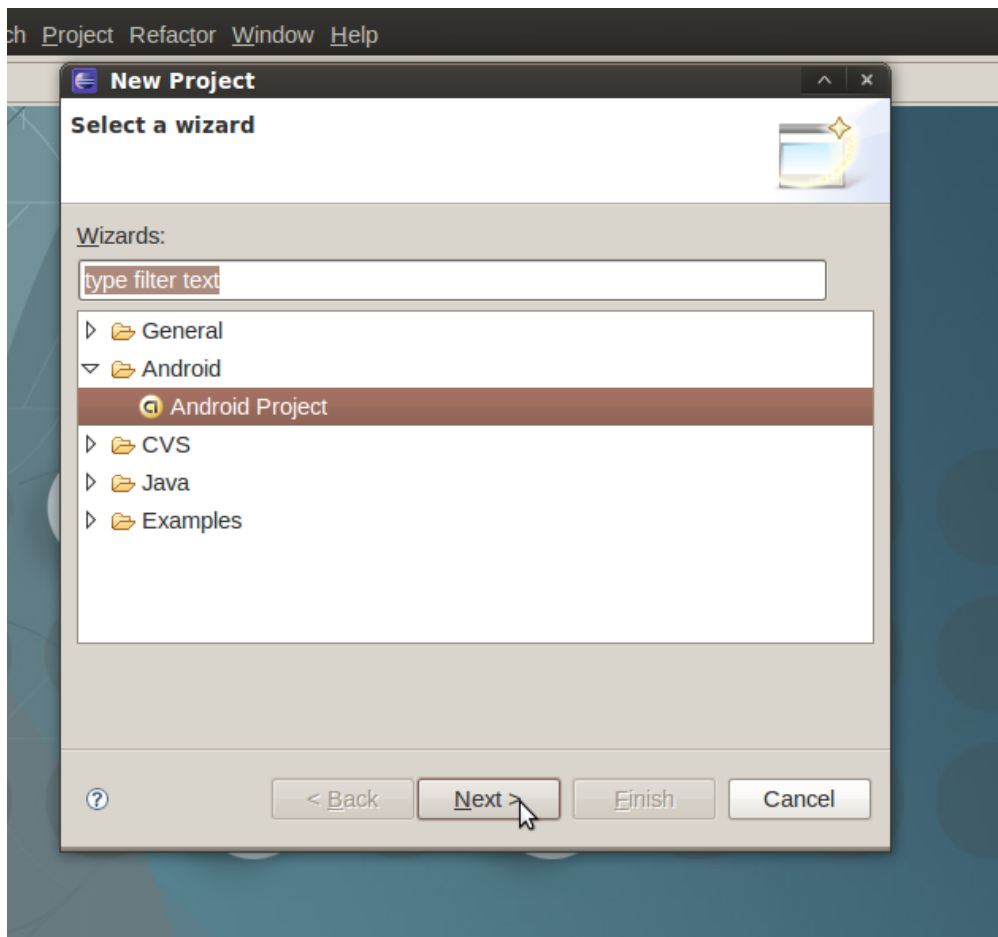


V. HELLO WORLD

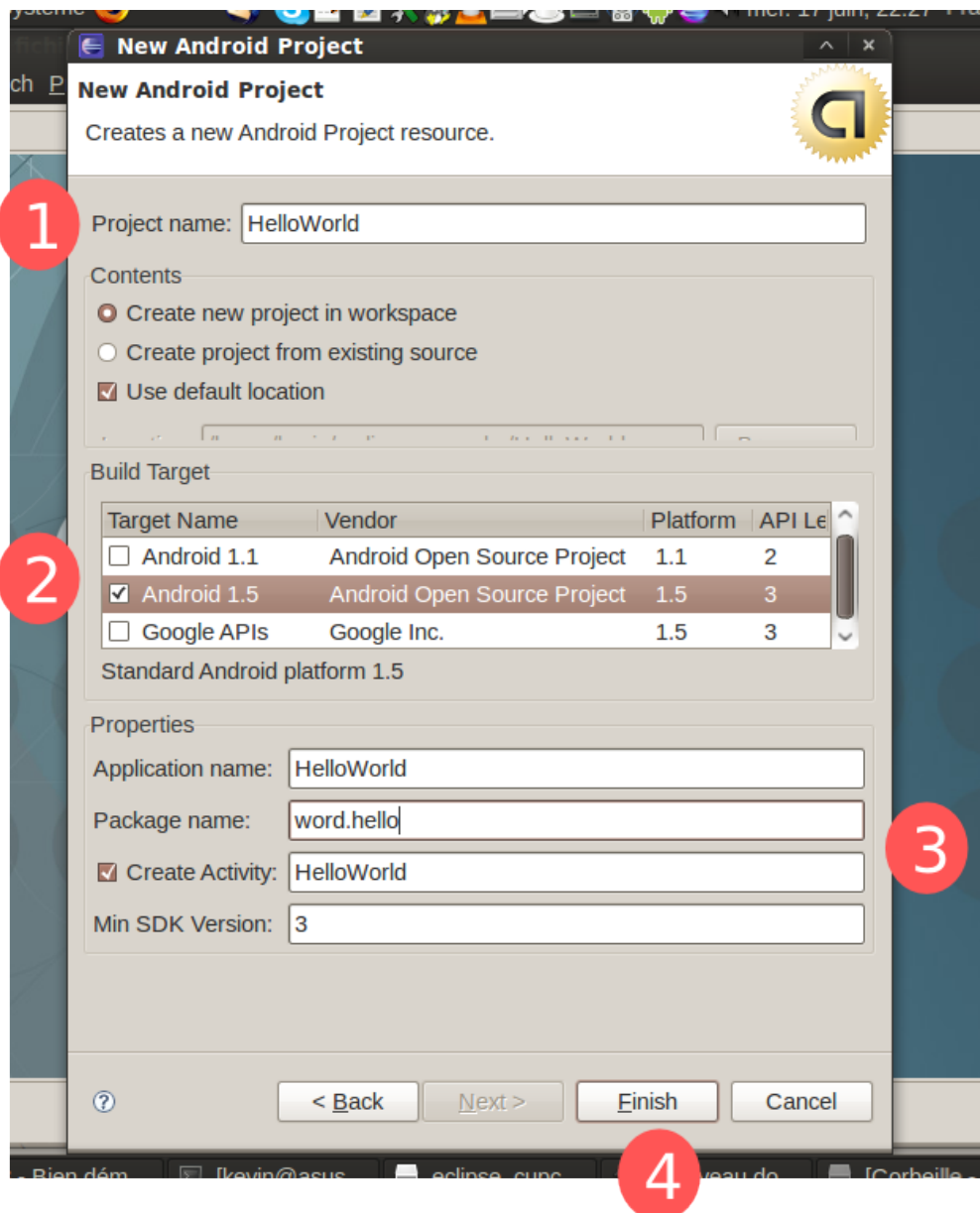
Tout code Android devra se faire dans le cadre d'un projet Eclipse. Pour créer un projet, on va dans "File" > "New" > "Project...".



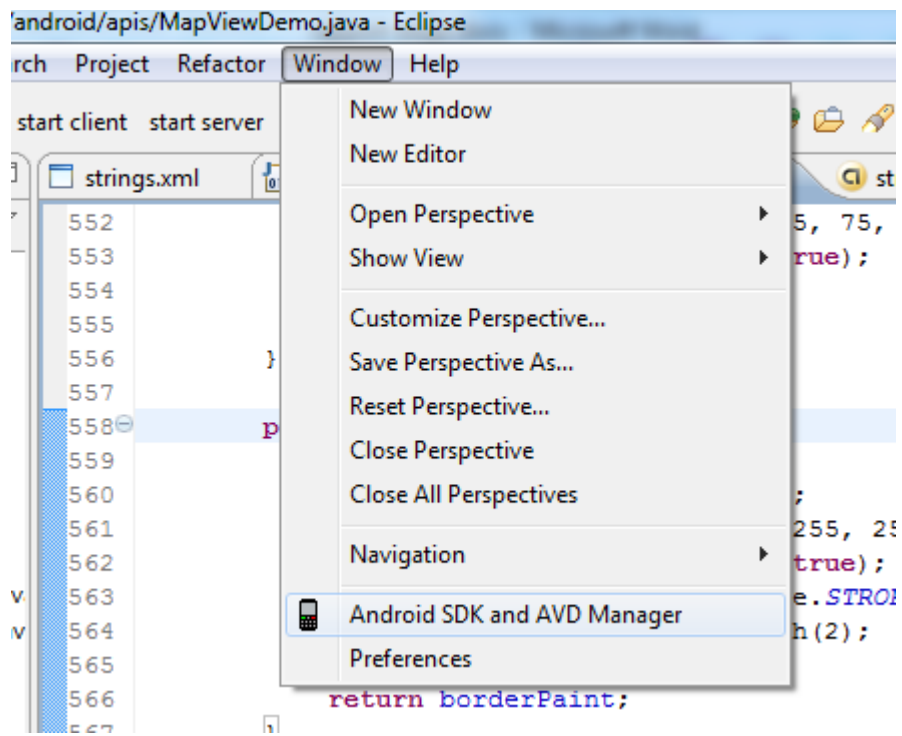
Créez un nouveau projet de type Android :



On passe ensuite à l'étape de configuration du projet :



Il ne reste plus qu'à faire un clic droit sur le projet et sélectionner « Run As > Android Application ».



Il faut choisir dans les options la cible sur laquelle s'exécute notre projet.

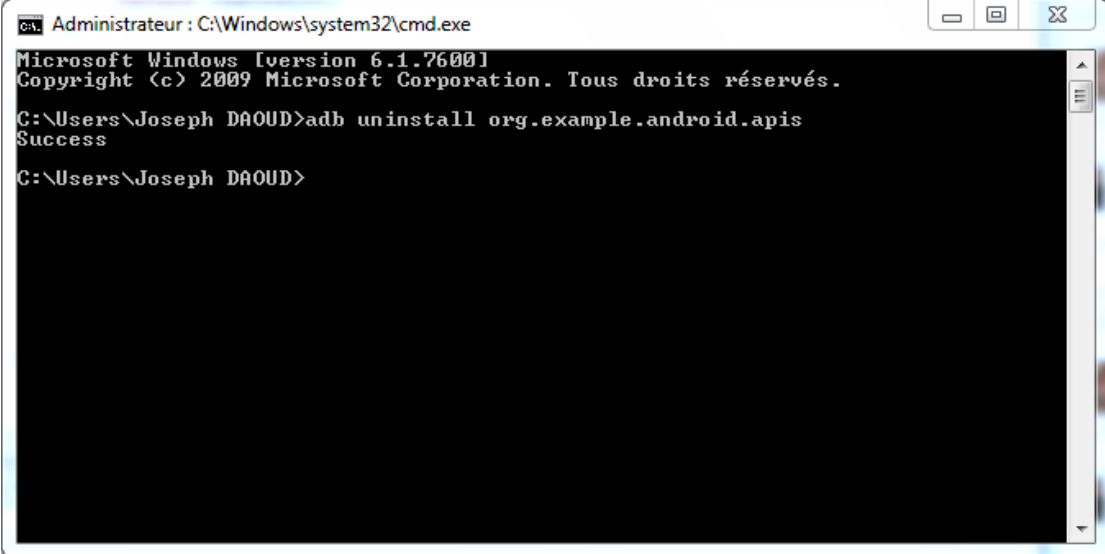


VI. DESINSTALLATION D'UNE APPLICATION

On peut avoir besoin de désinstaller une application. Pour cela, il suffit de lancer un terminal et de lancer l'émulateur où l'on souhaite désinstaller l'application.

Il suffit ensuite d'entrer :

```
adb uninstall nom_du_package
```



```
ca. Administrateur : C:\Windows\system32\cmd.exe
Microsoft Windows [version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.
C:\Users\Joseph DAOUD>adb uninstall org.example.android.apis
Success
C:\Users\Joseph DAOUD>
```