Développement Android
Préparer son environnement de développement

J.-F. Lalande

jean-francois.lalande@centralesupelec.fr

The goal of this part is to prepare your environement for being able to use Android Studio with a smartphone or an emulator. We list here the required software components and in the appendix the problem you may encounter.

First, you can refer to this documentation :

— The official install process [1]

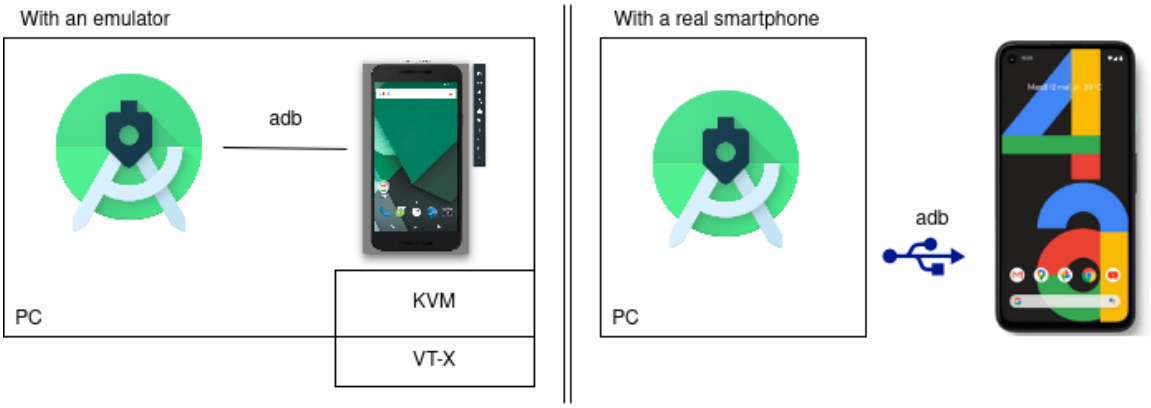
— This process for Windows with the setup of an emulator [2]

0.1 Configuration types

Two configuration can be used :

— Android Studio and an emulator that needs a virtualization backend (needs more resources)

— Android Studio and a real smartphone, that dialogs through an USB connection.



0.2 Needed components under Linux

— AndroidStudio 3.+.

— Kvm (only if you use an emulator, cf appendix, Section 4.1.2).
Type `kvm` ⇒ a window should pop up.

— (Optional) Java 8 (cf appendix, Section 4.1.1). A version of OpenJDK is bundled with Android Studio but you can have your own version of Java installed, preferably the same version [3].
Type `javac --version` ⇒ the version should be displayed

1. https://developer.android.com/studio/install
2. https://blogs.technet.microsoft.com/karanrustagi/2017/08/15/how-to-setup-android-emulator-using-android-studio/
3. https://stackoverrun.com/fr/q/12920224

0.3 Needed components under Windows

- AndroidStudio 3.+.
- (Optional) Only when using an emulator :
  - Hyper-V Hypervisor should be disabled (may conflict with the hypervisor backend, cf appendix, Section 4.2.3)
  - Intel x86 Emulator (HAXM) should be installed. Most recent versions of Windows 10 have this. (cf appendix, section 4.2.2)
  - Intel-VT should be activated in the BIOS. (cf appendix, Section 4.3.1)
- (Optional) An USB driver may be necessary to handle your smartphohe (cf appendix, Section 4.3.2)
- (Optional) Java 8 (cf appendix, Section 4.2.1). A version of OpenJDK is bundled with Android Studio but you can have your own version of Java installed, preferably the same version [4].

1 Ma première application

Exercice 1 Créez un nouveau projet Android Studio appelé "TDHelloWorld" ayant comme activité principale une "Empty Activity". Choisissez le niveau d'API minimum pour votre application. Une application Android 4.4 (KitKat) est un bon choix mais vous pouvez vérifier votre version d'Android sur votre téléphone personnel. Choisissez le langage Java. Cela permet de générer un code minimaliste pour démarrer.

En cas d'erreur :

```
Error:Failed to find target with hash string 'android-XX' in: ~/Android/Sdk
Install missing platform(s) and sync project
```

C'est parce qu'il vous manque le SDK dans la version cible XX. Un click sur "Install missing platform" va vous permettre de télécharger cette version du SDK.

Le problème peut aussi survenir pour les outils de compilation :

```
Error:Failed to find Build Tools revision XX
Install Build Tools XX and sync project
```

Exercice 2 Vérifiez ce qui a été installé par Android Studio dans le SDK d'Android (on n'installe jamais l'intégralité de ce qui est possible, sinon cela ferait des centaines de Go !). Pour se faire aller dans Fichier > Settings > Apparence > System Settings > Android SDK et cochez l'option "Show Package Details".

Si tout se passe bien, au bout de quelques secondes, Android Studio doit finir de construire le projet et afficher l'activité principale "MainActivity".

Exercice 3 Certains outils sont utiles en ligne de commande : `adb`, `sdkmanager` et `avdmanager`.

Sous Linux, ajoutez à votre `~/.bashrc` :

```
export PATH=$PATH:$HOME/Android/Sdk/platform-tools/
export PATH=$PATH:$HOME/Android/Sdk/tools/bin
```

et recharger votre `~/.bashrc` :

```
source .bashrc
```

Sous Windows, modifiez votre PATH (remplacez username !) :

```
setx PATH "%PATH%; C:\Users\<username>\AppData\Local\Android\sdk\platform-tools; C:\Users\<
    username>\AppData\Local\Android\sdk\tools/bin"
```

4. https://stackoverrun.com/fr/q/12920224

et redémarrez votre console.

Sous Mac OS, ajoutez à votre fichier `~/.bash_profile` :

```
export PATH="~/Library/Android/sdk/tools/bin/:$PATH"
export PATH="~/Library/Android/sdk/platform-tools/:$PATH"
```

et redémarrez votre console.

2 Comprendre l'architecture d'un projet

2.1 Les vues dans Android Studio

Plusieurs vues sont possible d'un même projet Android Studio. Sans rentrer dans les détails de chaque vue, vous pouvez tester les vues suivantes :

— La vue "Project" : arborescence brut des fichiers du projet ;
— La vue "Android" (par défaut) ;

2.2 Gradle

AndroidStudio utilise Gradle comme outil de build. Si vous êtes familier du développement Java, vous connaissez sans doute Maven ou Ant. Gradle résout des problématiques similaires.

Exercice 4 Pour comprendre rapidement ce que fait Gradle, vous pouvez parcourir cette page [5]. Vous pouvez même tester le tutoriel dans un répertoire à part (cela vous prendra 8 minutes !). Par défaut Android Studio utilise maintenant un Gradle Wrapper [6] mais il est possible d'installer une version locale à la machine de Gradle et de configurer Android Studio pour l'utiliser [7].

Exercice 5 Basculez la vue de votre projet Android en mode "Project". Trouvez les deux fichiers `build.gradle` qui configurent la construction de votre projet. L'un définit le paramétrage pour l'ensemble des applications (s'il y en a plusieurs) et l'autre définit les règles de construction pour votre application située dans le sous-répertoire "app".

Exercice 6 Dans le fichier `build.gradle` spécifique à l'application, coupez la ligne qui contient la dépendance "constraint-layout" [8], re-synchronisez votre projet (Sync now) et compilez l'application (Make Project). Que se passe-t-il ? Pourquoi cette dépendance externe est-elle indispensable à notre projet, qui pourtant ne contient que du code généré par défaut) ? Restaurez la ligne en question.

Les dépendances du projet permettent d'inclure des bibliothèque à votre application. Pour les dépendances de librairies Google, cela permet aussi de backporter des fonctionnalités récentes d'Android dans des versions antérieurs d'Android (ligne contenant "appcompat").

2.3 Les sources

Exercice 7 Parcourez les sources de votre applications et ouvrez les différents éléments situés dans :

— src/main/java
— src/main/res/layout
— src/main/res/values

5. https://guides.gradle.org/building-java-applications/
6. https://docs.gradle.org/6.4/userguide/gradle_wrapper.html
7. https://medium.com/@mydogtom/tip-reduce-the-number-of-gradle-daemon-instances-by-using-local-distribution-56f645cf2c97
8. C'est une librairie qui permet de définir des contraintes au sein d'une activité Android. Par exemple, on peut accrocher un bouton aux bords de l'écran avec des ressorts, définissant ainsi 4 contraintes, ce qui va automatiquement le centrer.

Exercice 8 Renommez le fichier `activity_main.xml` par `activity_main2.xml` à l'aide de la fonction de *refactoring* de votre IDE. Que se passe-t-il dans le fichier `MainActivity.java` ?

3 Exécuter son application

Pour exécuter votre application, deux solutions s'offrent à vous : un téléphone réel ou bien l'émulateur. Il faut préférer l'utilisation d'un téléphone réel, pour des questions de performances mais aussi pour des facilités de développement lorsqu'on utilise les services Google. Si vous utilisez une machine personnelle qui est moyennement puissante, préférez un téléphone réel car sinon vous risquez de mettre à genoux votre ordinateur !

3.1 Sur un téléphone réel

Exercice 9 Connectez votre téléphone sur l'USB et vérifiez sa présence à l'aide de la commande `adb devices`. En cas de problème, se reporter à l'annexe technique. Vous devez obtenir :

```
adb devices
List of devices attached
xxxx device
```

où xxx est l'identifiant de votre téléphone.

Exercice 10 Construisez et exécutez votre application en utilisant l'icône d'exécution en haut (triangle vert).

3.2 Dans un émulateur

Attention : la configuration d'un émulateur va copier l'image (∼1Go) dans `~/.android`...

Exercice 11 Rendez-vous dans l'AVD Manager (icône en haut à droite avec un téléphone rectangulaire et une tête de Droid (il est partout)). Paramétrez un émulateur en choisissant une configuration de téléphone et l'image du système que vous venez de télécharger (onglet x86 images). Lancez ce émulateur (colonne action).

Exercice 12 Construisez et exécutez votre application en utilisant l'icône d'exécution en haut à droite (triangle vert).

Exercice 13 Si vous avez configuré correctement votre `PATH` au début de ce TD, vous pouvez normalement invoquer la commande `avmanager` dans un terminal. Testez la commande `avdmanager list avd`.

3.3 A propos de l'émulateur sous Linux

Exercice 14 Rendez-vous dans le SDK Manager (icône en haut à droite avec une flèche vers le bas et une tête de Droid). En affichant le détail des paquets, vous pourrez installer un émulateur appelé "Google APIs Intel x86 Atom System Image" qui, une fois déployé, occupera plus de 3 Go [9].

Exercice 15 Allez dans le répertoire `~/Android/Sdk/system-images/android-XX/google_apis/x86$` (où XX est la version des APIs installées, par exemple 27 pour Oreo). Vous trouverez les fichiers utilisés par Qemu pour lancer l'image virtuelle :

- kernel-ranchu ($< 10Mo$) : le kernel de votre système virtualisé
- system.img ($> 2Go$) : l'image du système (lecture seule)
- ramdisk.img ($\sim 1Mo$) : la partition de boot

9. Si votre ordinateur est peu puissant, se rabattre sur une version ancienne de l'émulateur, par exemple la version 4.4 Kitkat qui est plus légère en terme de poids et moins gourmande en terme de ressource.

— userdata.img($\sim 500Mo$) : la partition de données initiale

Exercice 16 Allez dans le répertoire `/.android/avd/votre-AVD.avd$`. Listez les fichiers :

— userdata-qemu.img($500Mo$) : la partition de données, copie de userdata.img et qui est modifiable. Montée dans `/data`.

— cache.img : la partition de cache, montée dans `/cache`. Cette partition est utilisée par certaines applications comme le navigateur pour les téléchargements temporaires.

— sdcard.img : simule une carte SD optionnelle.

3.4 Log et Debug

Exercice 17 Ajoutez à la fin de la méthode `onCreate()` la ligne suivante qui permet de loguer "Une ligne de log" pour le tag "JFL" :

```
Log.i("JFL", "Une ligne de log");
```

Exercice 18 Dans AndroidStudio, ouvrez l'onglet Logcat (en bas). Entrez le tag "JFL" dans le champs de recherche afin de filtrer le bruit. Vérifiez que Logcat est bien connecté sur le bon périphérique (téléphone ou émulateur).

Exercice 19 Exécutez votre application. Que constatez-vous ?

Exercice 20 Posez un point d'arrêt dans votre code. Testez l'exécution en mode debug. Avancez en pas à pas.

4 Technical appendix

4.1 GNU/Linux

4.1.1 Java

```
sudo apt install openjdk-8-jdk
```

4.1.2 Kvm

```
sudo apt-get install qemu-kvm
```

4.1.3 About the SDK Android and the 64 bits architecture

When the SDSK is installed, the different tools (AVD manager, adb, etc.) uses old 32 bit libraries. For example, you can check that the `aapt` tool is working :

```
cd ~/Android/Sdk # ou autre emplacement suivant votre install
cd build-tools/23.0.1 # ou autre emplacement suivant la version
./aapt
error=2, No such file or directory
```

If you obtain this error, you should install these libraries :

```
sudo dpkg --add-architecture i386
sudo apt-get install libc6:i386 libncurses5:i386 libstdc++6:i386 lib32z1 libbz2-1.
```

or

```
sudo yum install zlib.i686 ncurses-libs.i686 bzip2-libs.i686
```

4.1.4 Permission error when using adb

Symptoms :

```
adb devices
List of devices attached
* daemon not running. starting it now at tcp:5037 *
* daemon started successfully *
xxxx    no permissions (verify udev rules); see [http://developer.android.com/tools/
    device.html]
```

This problem is linked to the fact that your peripheral is associated to the `root` user and groups, and not to the `udev` group. You will need to add an udev rule to your system.

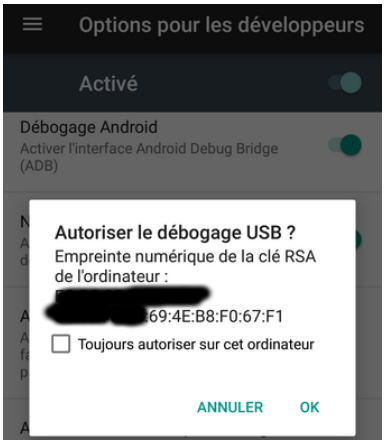
Solution 1 (cf https://github.com/M0Rf30/android-udev-rules) :

```
# Clone this repository
git clone https://github.com/M0Rf30/android-udev-rules.git
# Copy udev rules
sudo cp android-udev-rules/51-android.rules /etc/udev/rules.d/51-android.rules
# Change file permissions
sudo chmod a+r /etc/udev/rules.d/51-android.rules
# Restart UDEV
```

```
sudo udevadm control --reload-rules
sudo service udev restart
# Restart the ADB server
adb kill-server
# Replug your Android device and verify that USB debugging is enabled in developer
adb devices
# You should now see your device
```

Solution 2 : if the previous solution does not work, it seems that your device is so particular that is not listed in this file. Thus, you need to create the rule by hand. Please, refer to http://www.janosgyerik.com/adding-udev-rules-for-usb-debugging-android-devices/.

4.1.5 unauthorized error with adb



If you get such an output with adb :

```
adb devices
List of devices attached
d65a8b0a        unauthorized
```

You just have to authorize the USB debut of your smartphone when this popup appears when the cable is plugged in.

4.2 Windows

4.2.1 Java

You have to download and instal Java SE Development Kit 8 for Windows 64 bit [10].

4.2.2 Virtualization driver HAXM

For some Windows version, the HAXM may be missing, preventing to access the Intel virtualization technology : the emulator cannot be launched. In the SDK Manager, Extra section, you can install HAXM. You may have to lauch the install process manually by running the binary `androidhaxm-android.exe`. [11]

4.2.3 Check that Hyper-V is not installed

In the window for installing/uninstalling programs, click on the left menu "Turn Windows features on or off". In the new window, uncheck Hyper-V -> Hyper-V Platform -> Hyper-V Hypervisor.

4.2.4 The `adb` command is not recognized

The adb tool is useful when used in a command prompt. The tool is probably installed in the system if you have installed Android Studio but you miss it in the PATH. You can probably find it in one of the following locations :

— `C:\Android\sdk\platform-tools`
— `C:\Program Files\android-sdk-windows\platform-tools`

10. https://www.oracle.com/java/technologies/javase-jdk8-downloads.html
11. http://www.ambracode.com/index/show/1394904

— `C:\Users\<username>\AppData\Local\Android\sdk`
— `%USERPROFILE%\AppData\Local\Android\sdk\platform-tools`

For putting the location in your PATH, you can do [12] :

— Click 'Start'.
— Type 'Edit environment variables for your account', and click it.
— Double click PATH.
— Click 'New'.
— Paste that path in.
— Click 'OK', click 'OK', and restart any command prompts you have open.

```
setx PATH "%PATH%; C:\Users\<username>\AppData\Local\Android\sdk\platform-tools; C:\
    Users\<username>\AppData\Local\Android\sdk\tools/bin"
```

Further discussion here [13].

4.3 For all platforms

4.3.1 Intel VT is not activated

You should enable Intel-VT in your BIOS. Depending of your hardware, the procedure to go into the BIOS may vary. You can have a look to these pages [14] [15].

4.3.2 The smartphone is invisble in Android Studio

Several reasons can cause this issue :

1. You have not unlocked the developer options and the USB debugging in the smartphone : Go to the parameters, in the "About" section and click 5 times on "Build number". Return to the parameters and click on "Developer options" : activate the USB debugging. Unplug and re-plug the smartphone.
2. If the problem persist on Windows, you may need to install a USB pilote for your device. [16]

4.3.3 Graphical acceleration

If you have no graphic card with hardware acceleration with Open GL drivers, the emulator will be slow but you can use it anywhere. You have to configure your virtual device by unckecking the box "Host GPU".

4.3.4 SDK Manager

When the SDK have been installed, you can check that these components are installed :

— Tools
  — Android SDK Tools
  — Android SDK Platform-tools
  — Android SDK Build-tools (la plus récente version)
— In Extras :
  — Android Support Repository
  — Android Support Library

12. https://stackoverflow.com/a/40754030/1156363
13. https://stackoverflow.com/questions/20564514
14. http://f4b1.com/materiel-hardware/comment-activer-la-virtualisation-vtx-amd-v-dans-bios
15. https://support.bluestacks.com/hc/fr-fr/articles/115003910391--Comment-puis-je-activer-la-virtualisation-VT-sur-mon-PC
16. https://developer.android.com/studio/run/oem-usb#InstallingDriver

- Google Play Services

For one of the version of Android :

- SDK Platform
- Emulateur Intel x86 Atom_64