

PROGRAMMATION

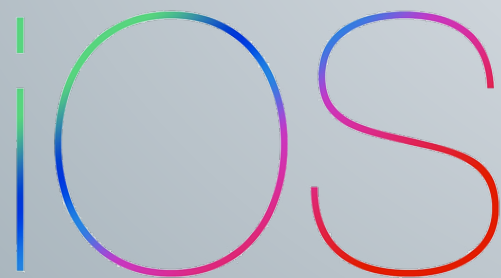
MOBILE

1 - Environnement de développement Android

NIDHAL JELASSI

nidhal.jelassi@fsegt.utm.tn

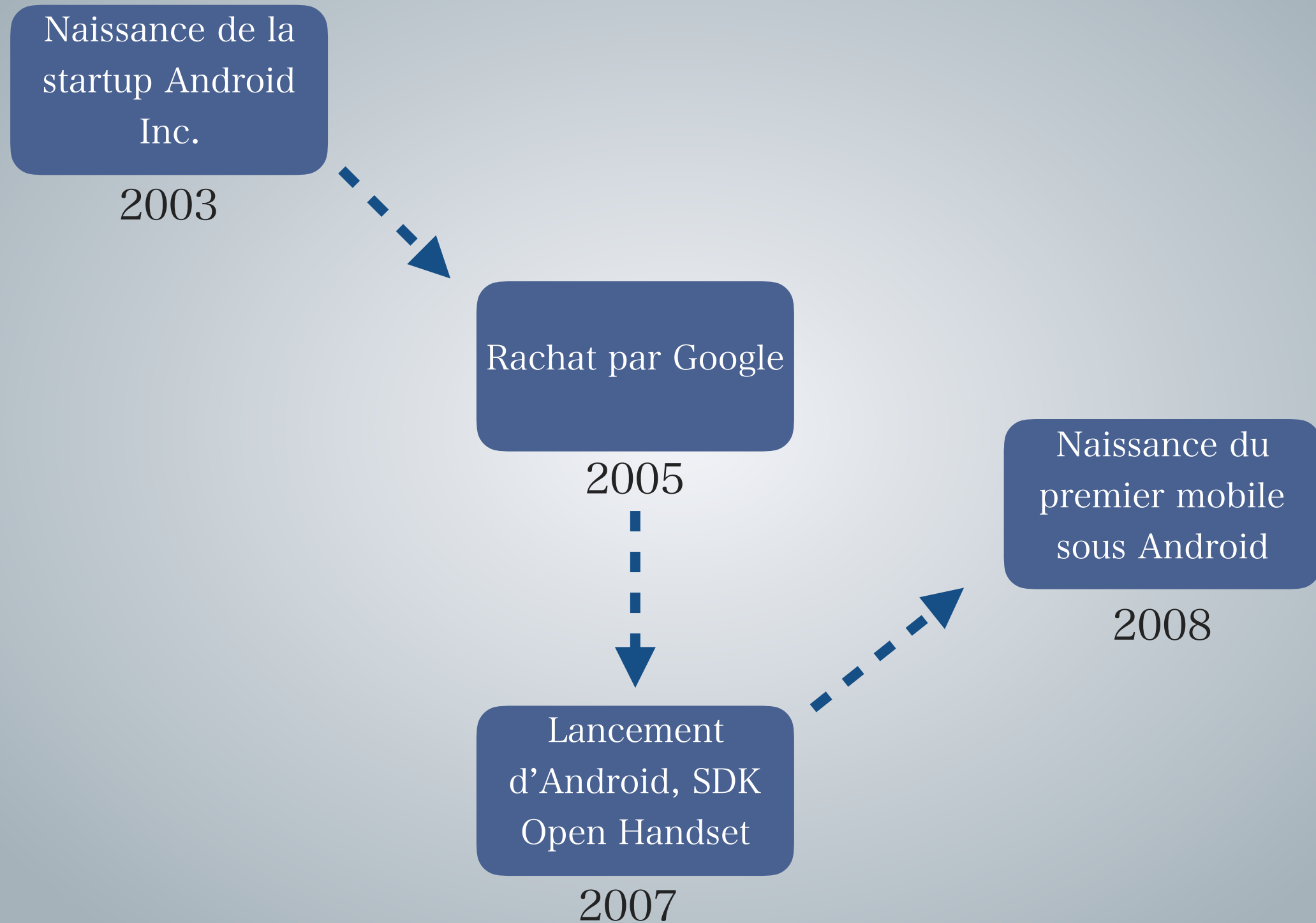
La Prog. Mobile



Android, c'est ...

- Tout ça à la fois :
 1. Un système d'exploitation Open Source pour des terminaux mobiles.
 2. Une plateforme de développement Open Source pour créer des applications mobiles.
 3. Des terminaux qui exécutent le système d'exploitation Android et les applications mobiles conçues pour ce système.

Historique



Périphériques Android

- Smartphones
- Tablettes
- Netbooks
- TV
- Embarqués (montres, voitures, etc.)

Evolution

- Évolution et obsolescence très rapides (c'est voulu)
- Ce que vous allez apprendre sera rapidement dépassé (1 an)
 - Syntaxiquement (méthodes, paramètres, classes, ressources, etc.)
 - Mais pas les concepts (principes, organisation, etc.)

=> Vous êtes condamnés à une auto-formation permanente...

Mobile vs Web

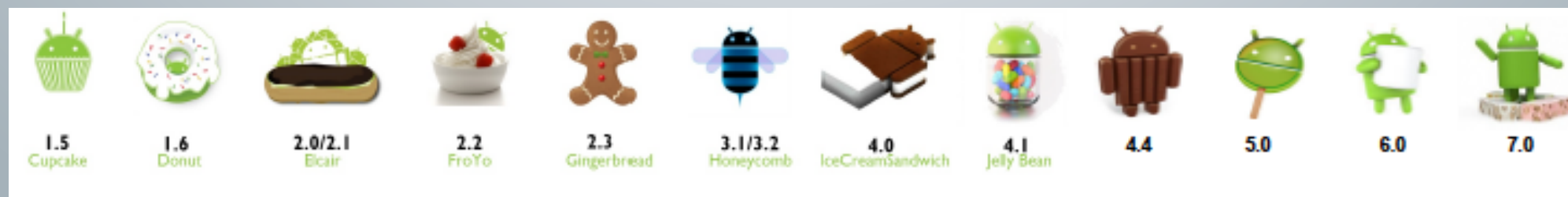
	App Mobile	App Web
Portabilité	Développement spécifique . chaque plateforme	Navigateur Web
Développement	SDK Connaissance d'un langage spécifique	Langage du Web
Mises à jour	Soumission sur le Store Validation Re-téléchargement par le client	Mise à jour rapide du serveur Web
Disponibilité	Mode online et offline	Nécessité obligatoirement une connexion internet
Fonctionnalités	Utilise toutes les fonctionnalités du mobile	Limitées aux possibilités du navigateur

Points forts

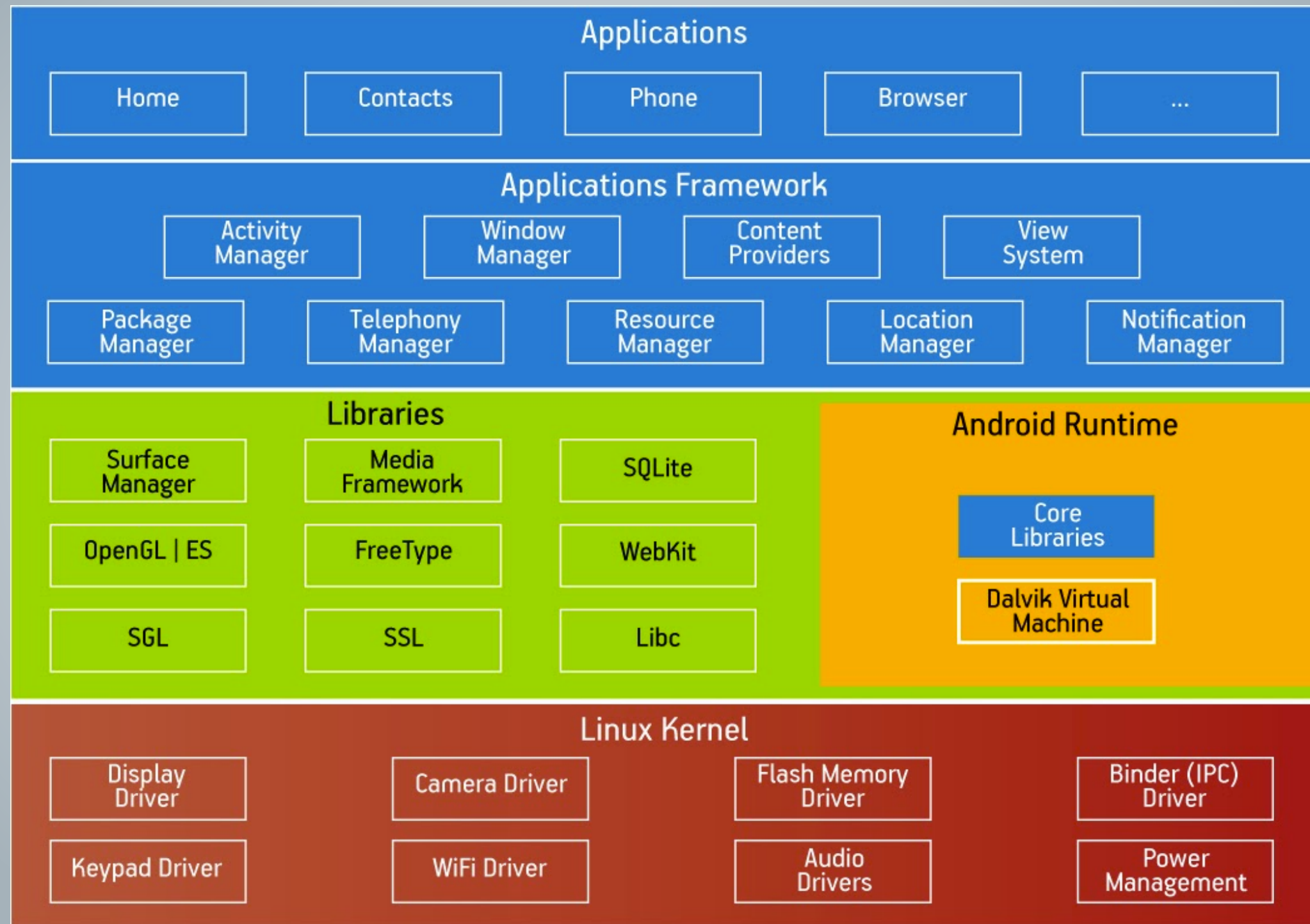
- Côté utilisateur
 - Système fonctionnel
 - Sys. Intuitif
 - Sys. Evolutif
- Côté développeur
 - Syntaxe Java
 - SDK complet

Versions d'Android

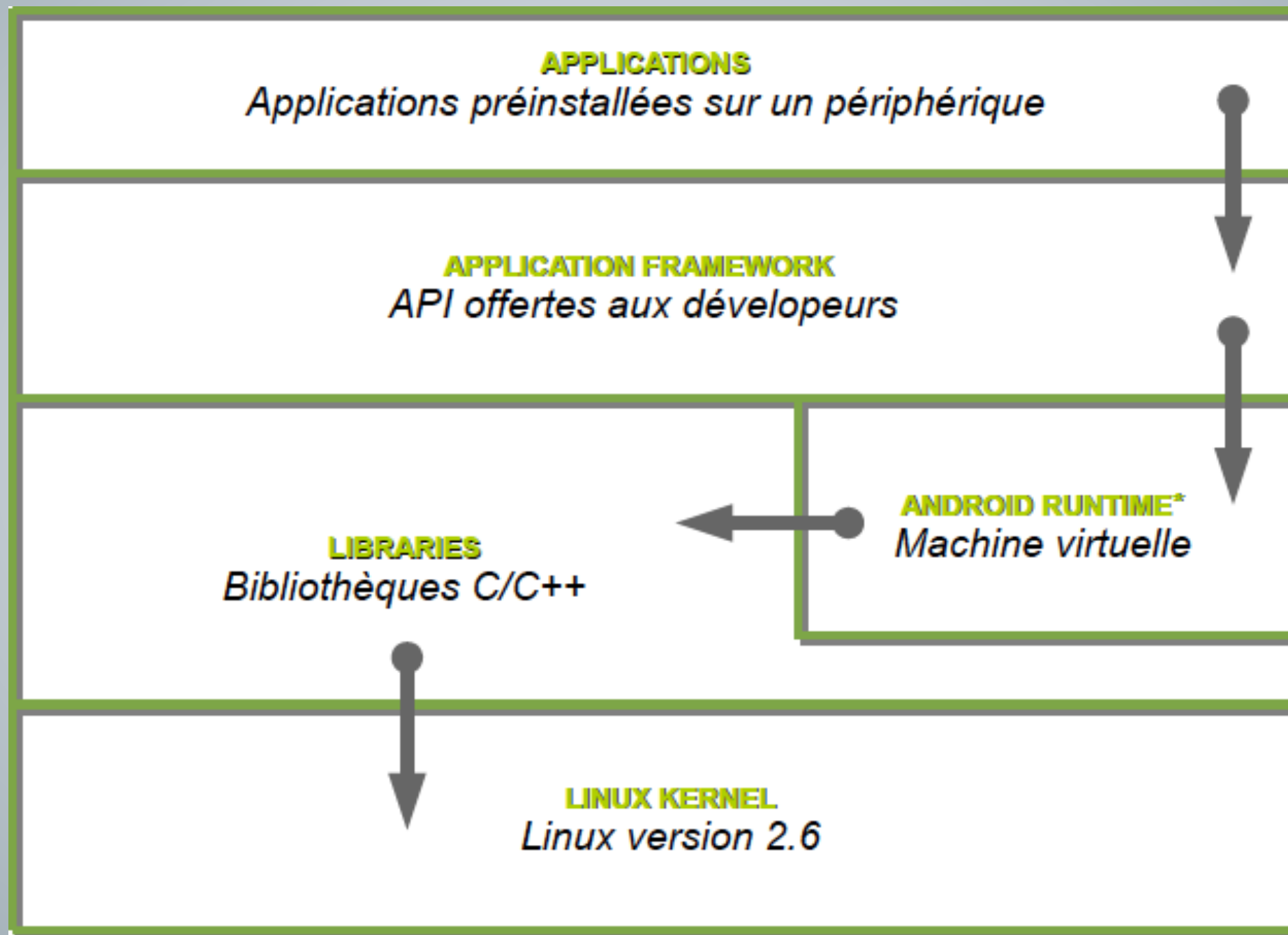
CodeName	Platform	API Level
Cupcake	1.5	3
Donut	1.6	4
Eclair	2.1	7
Frodo	2.2	8
Gingerbread	2.3	9
Honeycomb	3.0	11
Ice Cream Sandwich	4.0	14
Jelly Bean	4.1	16
KitKat	4.4	19
Lollipop	5.0	20
Marshmallow	6.0	23
Nougat	7.0	24
Oreo	8.0	25



Architecture



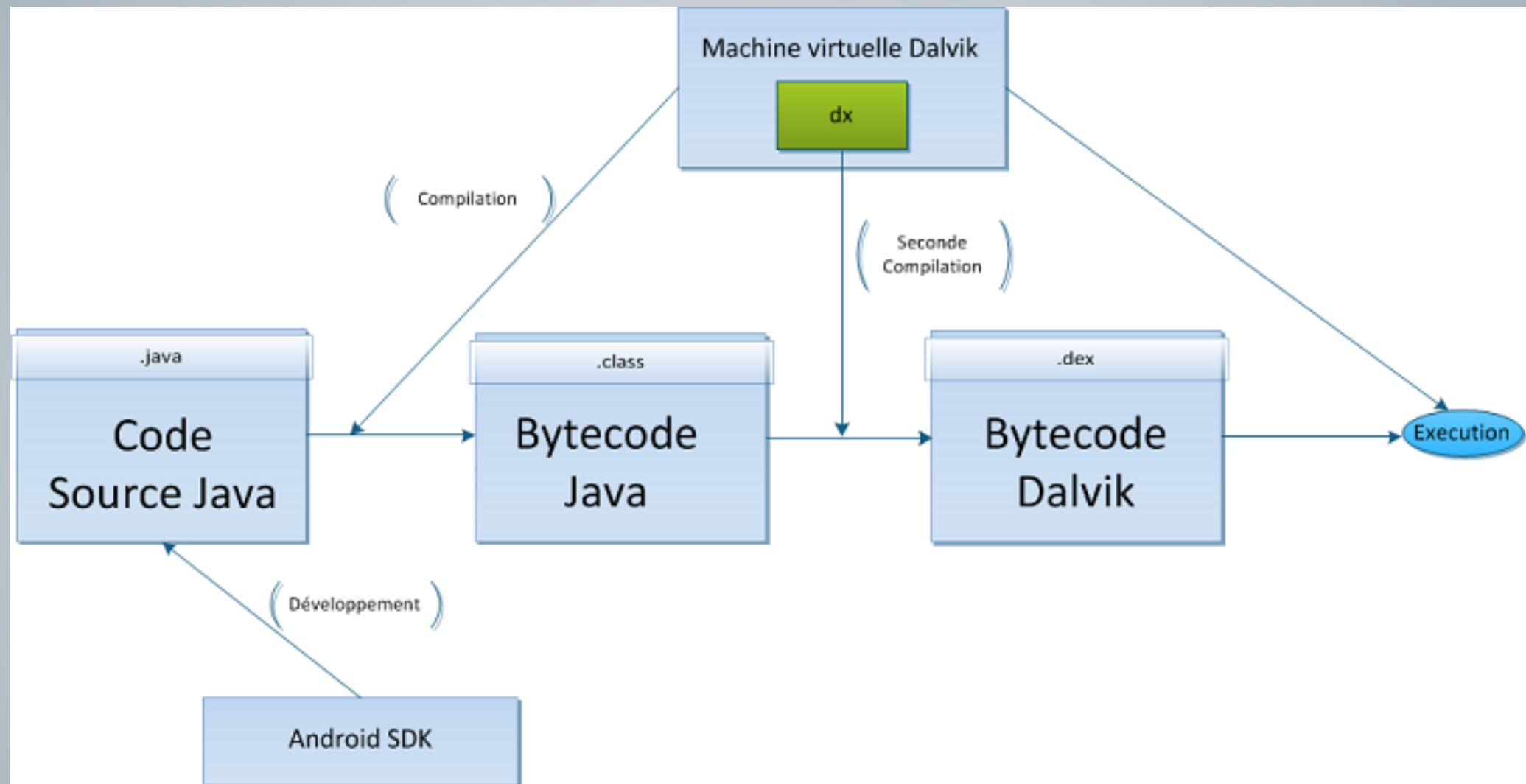
Architecture



- Un ensemble de bibliothèques noyau qui fournissent la plupart des fonctionnalités disponibles dans les bibliothèques noyau du langage de programmation Java
- La VM Dalvik n'est pas une VM Java
 - Optimisée pour mieux gérer les ressources physiques du système (CPU, mémoire)
- La VM crée une instance Dalvik pour chaque application
 - Les applications sont totalement indépendantes

MV Dalvik

ANDROID RUNTIME*
Machine virtuelle



Les étapes nécessaires à la compilation et à l'exécution d'un programme Android standard

Application Android

- Sources Java compilés pour une machine virtuelle « **Dalvik** » (versions ≤ 4.4) ou « **ART** » depuis la version 5.
- Fichiers **XML** appelés ressources : interface, textes, etc.
- Fichiers de données supplémentaires
- **Manifest** = description du contenu du logiciel
 - Fichiers présents dans l'archive
 - Demandes d'autorisations
 - Signature des fichiers, durée de validité, etc.

Application Android

- Tout cet ensemble est géré à l'aide d'un IDE (environnement de développement) appelé Android Studio. Avant, on pouvait utiliser Eclipse, mais son plugin Android n'est plus mis à jour.
- En plus d'Android Studio, on a besoin de :
 - JDK
 - SDK
 - API

JDK

- Un petit rappel technique ne fait de mal à personne. Il existe deux plateformes en Java :
- Le **JRE** (JavaRuntimeEnvironment), qui contient la JVM, les bibliothèques de base du langage ainsi que tous les composants nécessaires au lancement d'applications ou d'applets Java.
- Le **JDK** (JavaDevelopmentKit), qui contient le JRE (afin d'exécuter les applications Java), mais aussi un ensemble d'outils pour compiler et déboguer votre code.

SDK

- Les applications Android sont développées en Java, mais un appareil sous Android ne comprend pas le Java tel quel, il comprend une variante du Java adaptée pour Android.
- Un SDK, un kit de développement dans notre langue, est un ensemble d'outils permettant de développer pour une cible particulière. Par exemple pour développer pour une console de jeu vidéo, on utilise un SDK spécifique pour développer des applications pour cette console. Le SDK Android est donc un ensemble d'outils que met à disposition Google afin de vous permettre de développer des applications pour Android.

API

- Il y a plusieurs versions de API Android.
- Quand on développe une application, il faut prendre en compte ces numéros, puisqu'une application développée pour une version précise d'Android fonctionnera sur les versions suivantes d'Android mais pas sur les versions antérieures.
- A des fins pédagogiques, j'ai choisi de travailler avec une version assez récente d'Android pour vous présenter toutes les possibilités de développement. Mais dans vos développements réels, il vous faudra bien réfléchir à quelle version viser, en fonction du public visé.

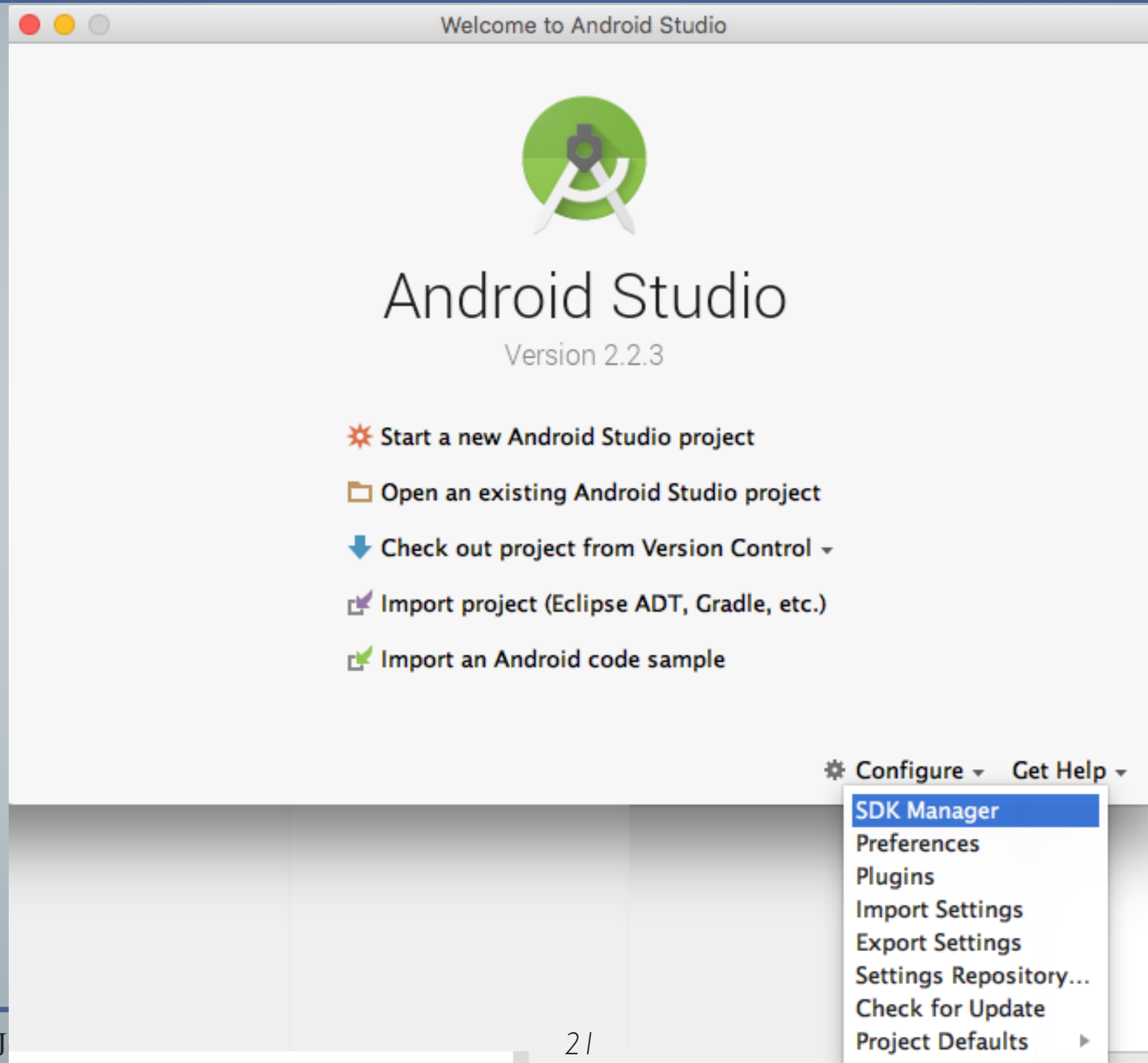
Ressources matérielles

- 2 Go de mémoire RAM, mais on va pas se cacher qu'en dessous de 4Go vous risquez d'être limité.
- Plus de 3Go de mémoire disque pour tout installer.
- Niveau processeur, l'émulation ne peut se faire que sur 1e core de votre processeur. Augmenter le nombre de cores ne vous servira pas à grand chose. C'est vraiment la puissance pure qui compte. Il n'y a donc pas de minima mais le plus rapide sera le mieux.

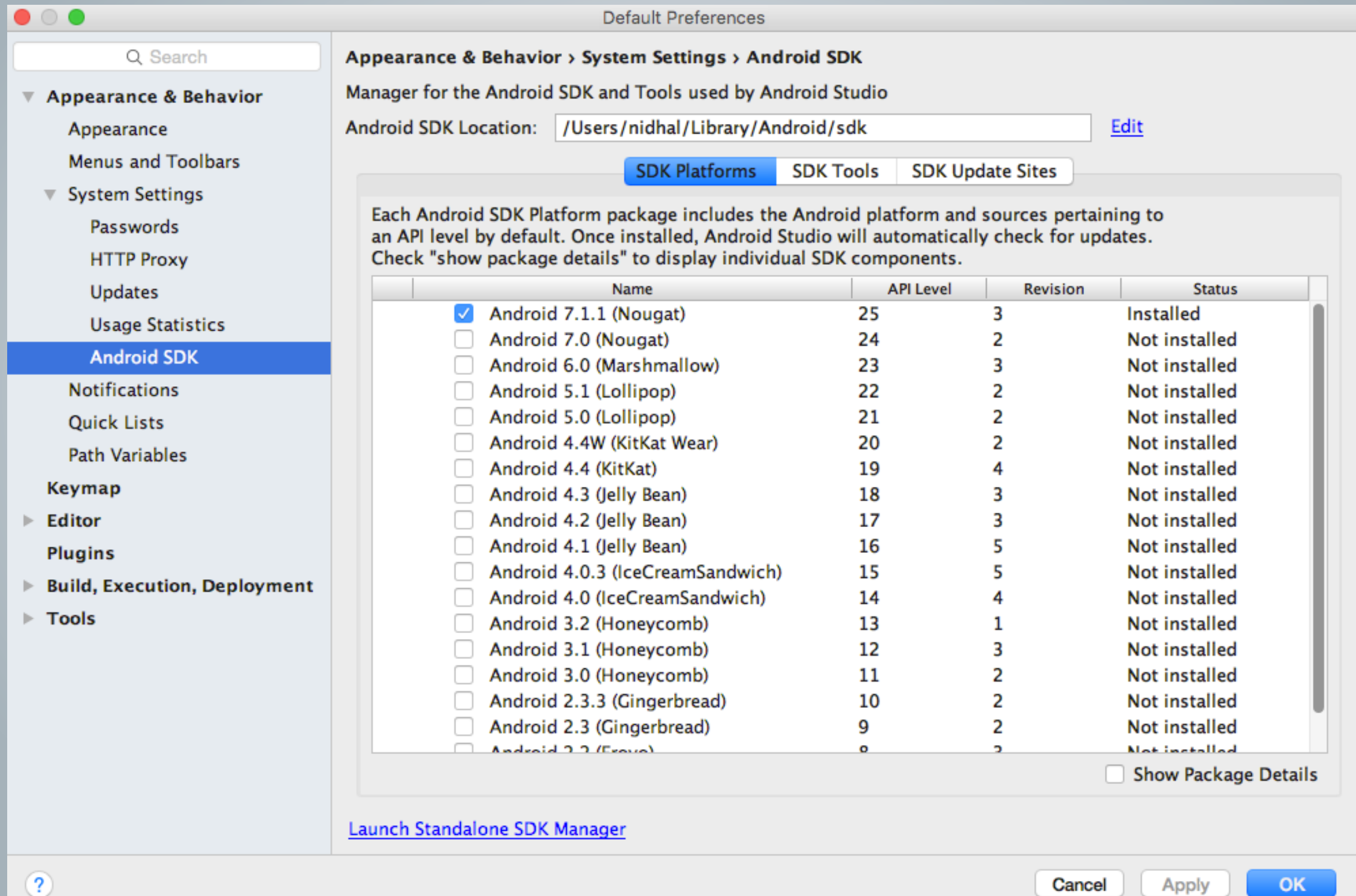
Config. Android Studio



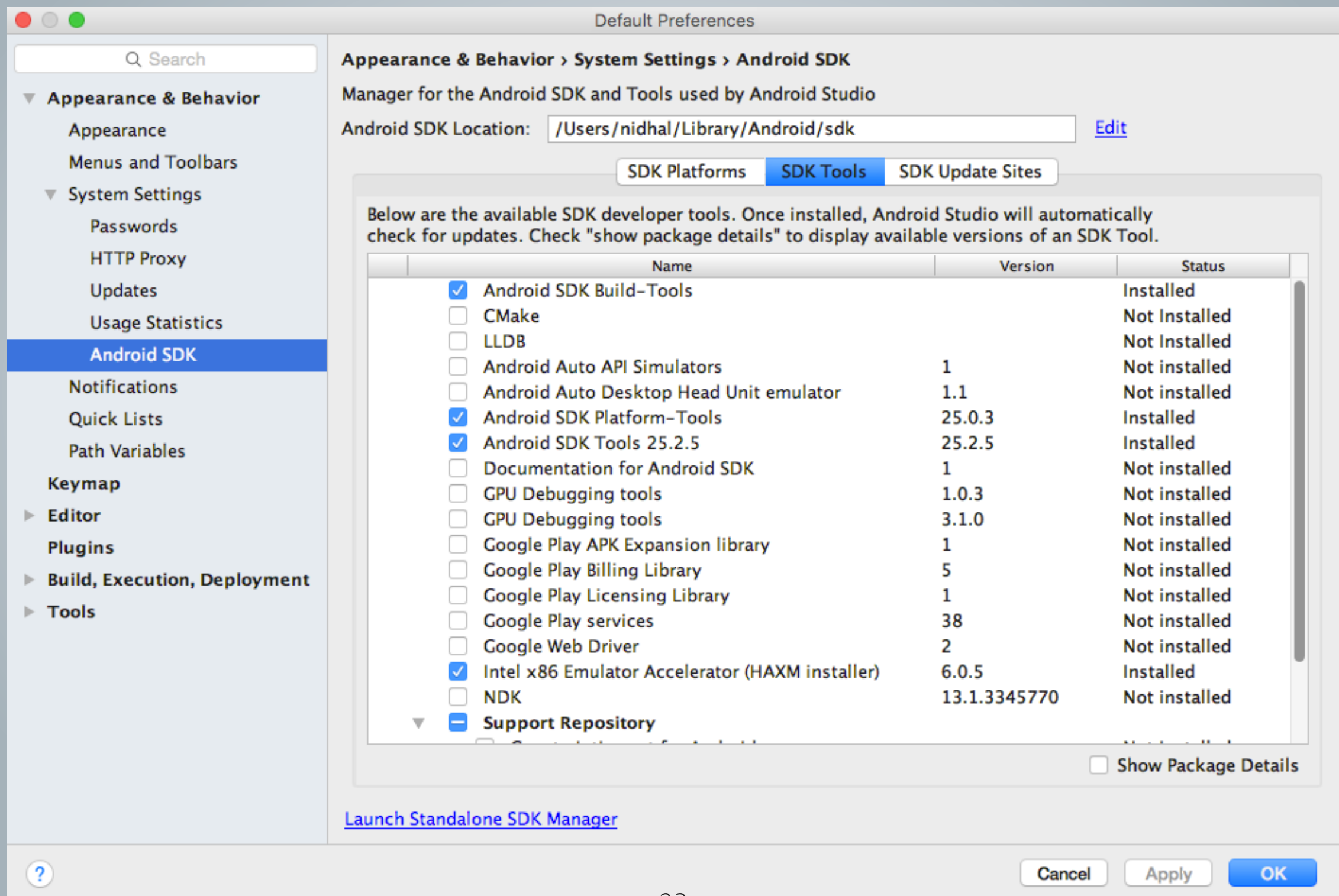
Config. Android Studio



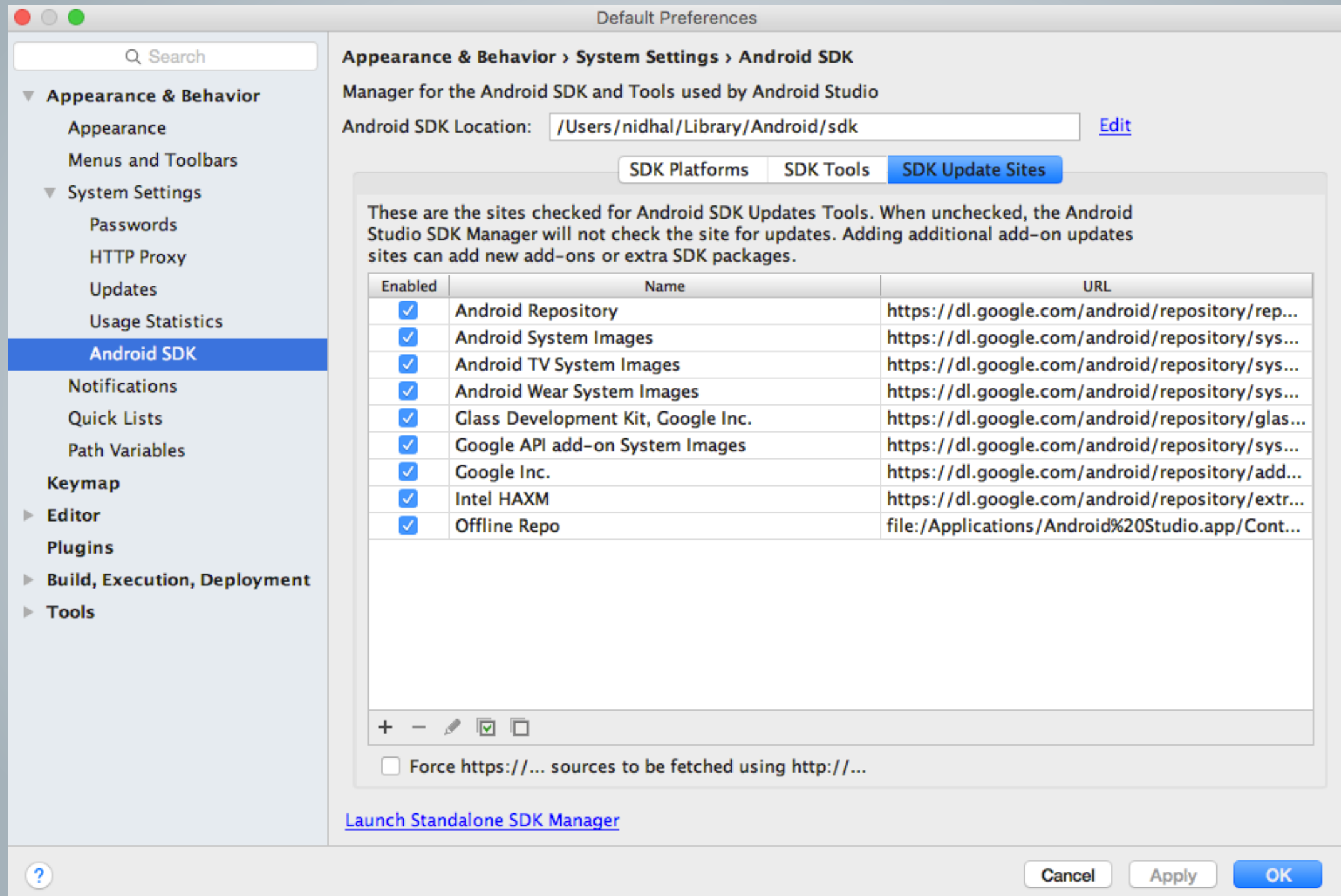
Config. Android Studio



Config. Android Studio




Config. Android Studio



First App

Create New Project

 Target Android Devices

Select the form factors your app will run on

Different platforms may require separate SDKs

☒ Phone and Tablet

Minimum SDK

Lower API levels target more devices, but have fewer features available.
By targeting API 15 and later, your app will run on approximately 97,4% of the devices that are active on the Google Play Store.
[Help me choose](#)

☐ Wear

Minimum SDK

☐ TV

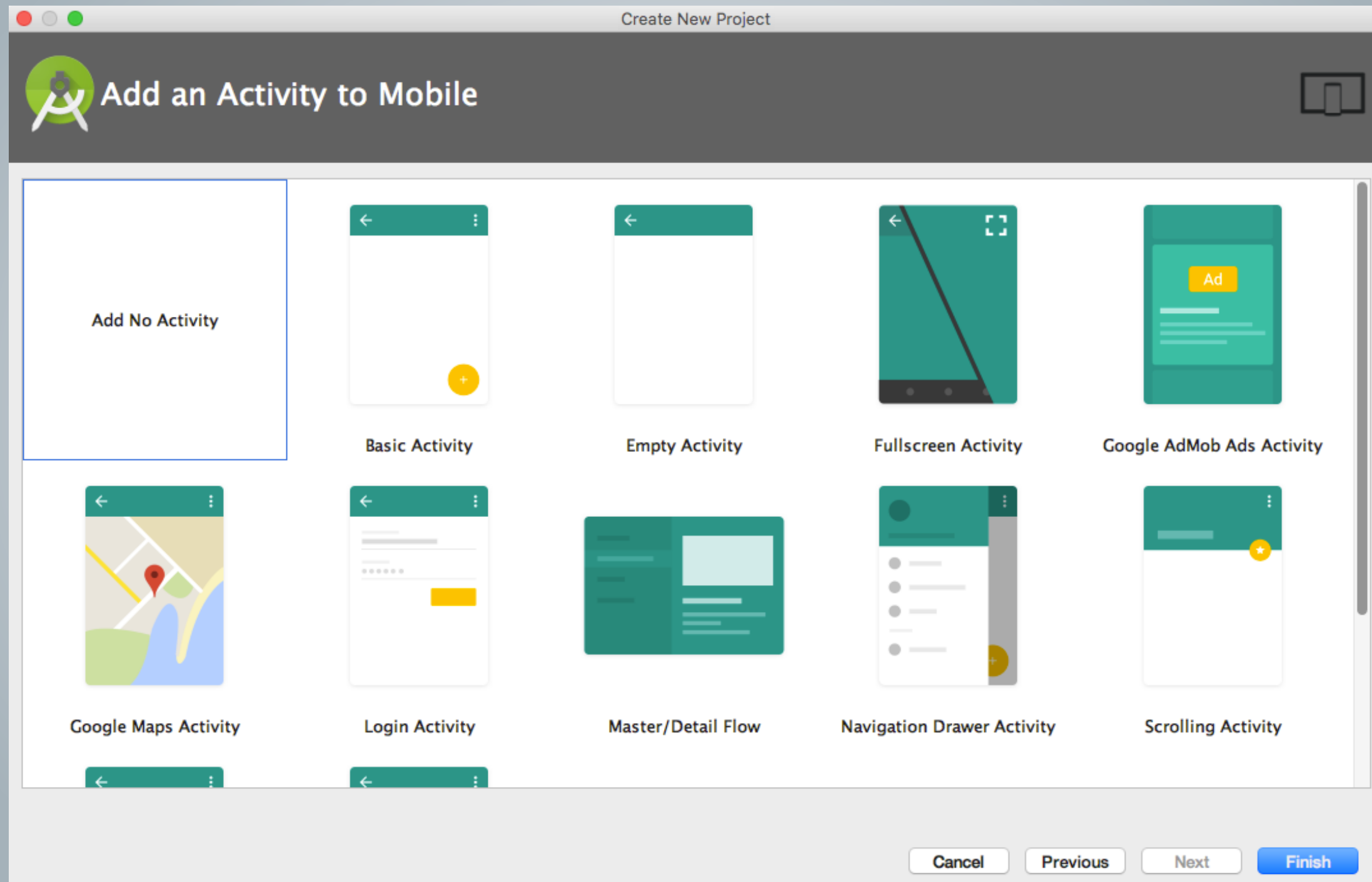
Minimum SDK

☐ Android Auto

☐ Glass

Minimum SDK

Ajouter une activité



Emulateur : AVD

Virtual Device Configuration

Select Hardware
Android Studio

Choose a device definition

Q

Category	Name	Size	Resolution	Density
TV	Nexus S	4,0"	480x800	hdpi
Wear	Nexus One	3,7"	480x800	hdpi
Phone	Nexus 6P	5,7"	1440x2560	560dpi
Tablet	Nexus 6	5,96"	1440x2560	560dpi
	Nexus 5X	5,2"	1080x1920	420dpi
	Nexus 5	4,95"	1080x1920	xxhdpi
	Nexus 4	4,7"	768x1280	xhdpi
	Galaxy Nexus	4,65"	720x1280	xhdpi
	5.4" FWVGA	5,4"	480x854	mdpi
	5.1" WVGA	5,1"	480x800	mdpi
	4.7" WXGA	4,7"	720x1280	xhdpi
	4.65" 720p (Galaxy Nex...	4,65"	720x1280	xhdpi

New Hardware Profile Import Hardware Profiles

Nexus 4

768px

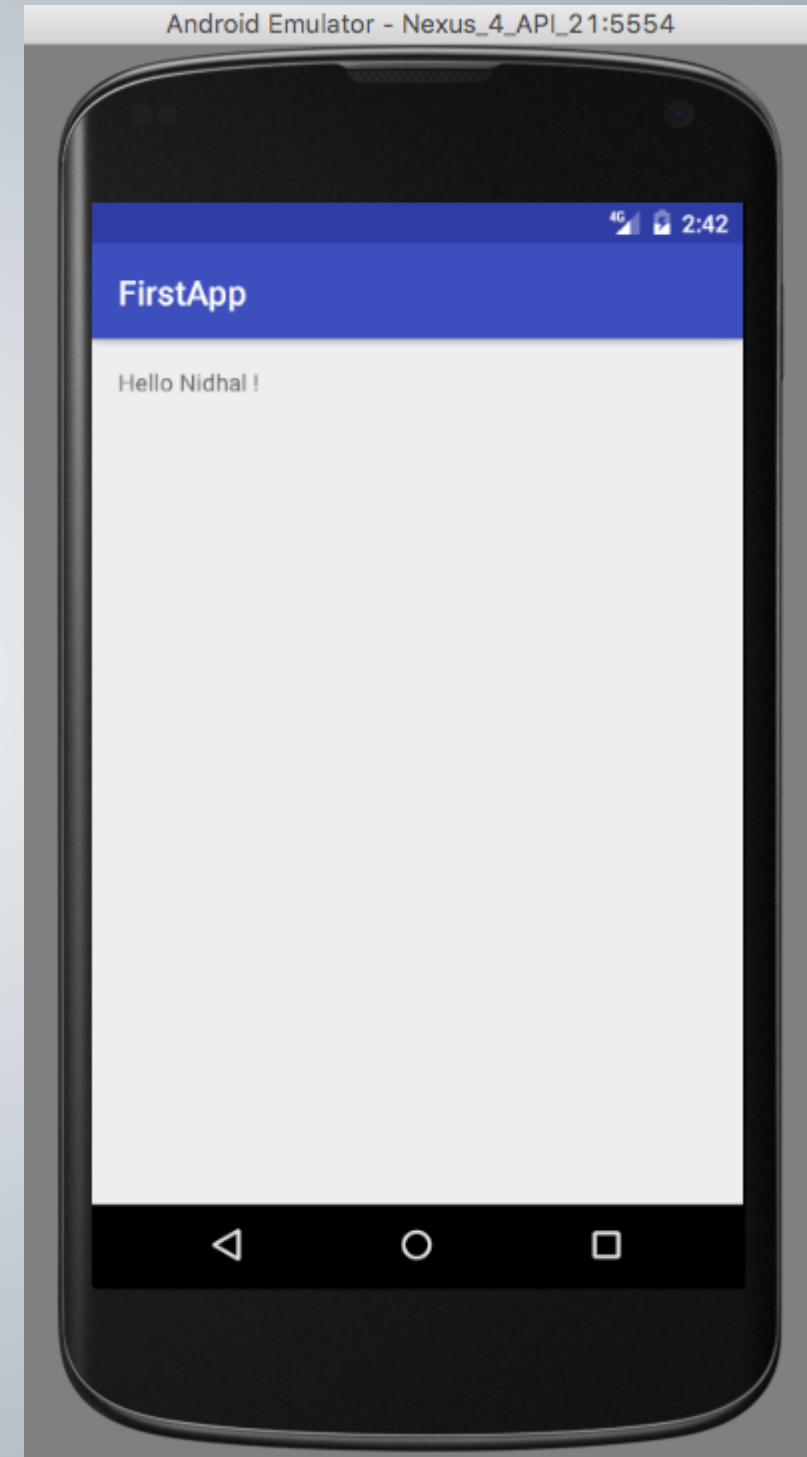
4,7"

1280px

Size: normal
Ratio: long
Density: xhdpi

Clone Device...

Emulateur : AVD



Emulateur : AVD

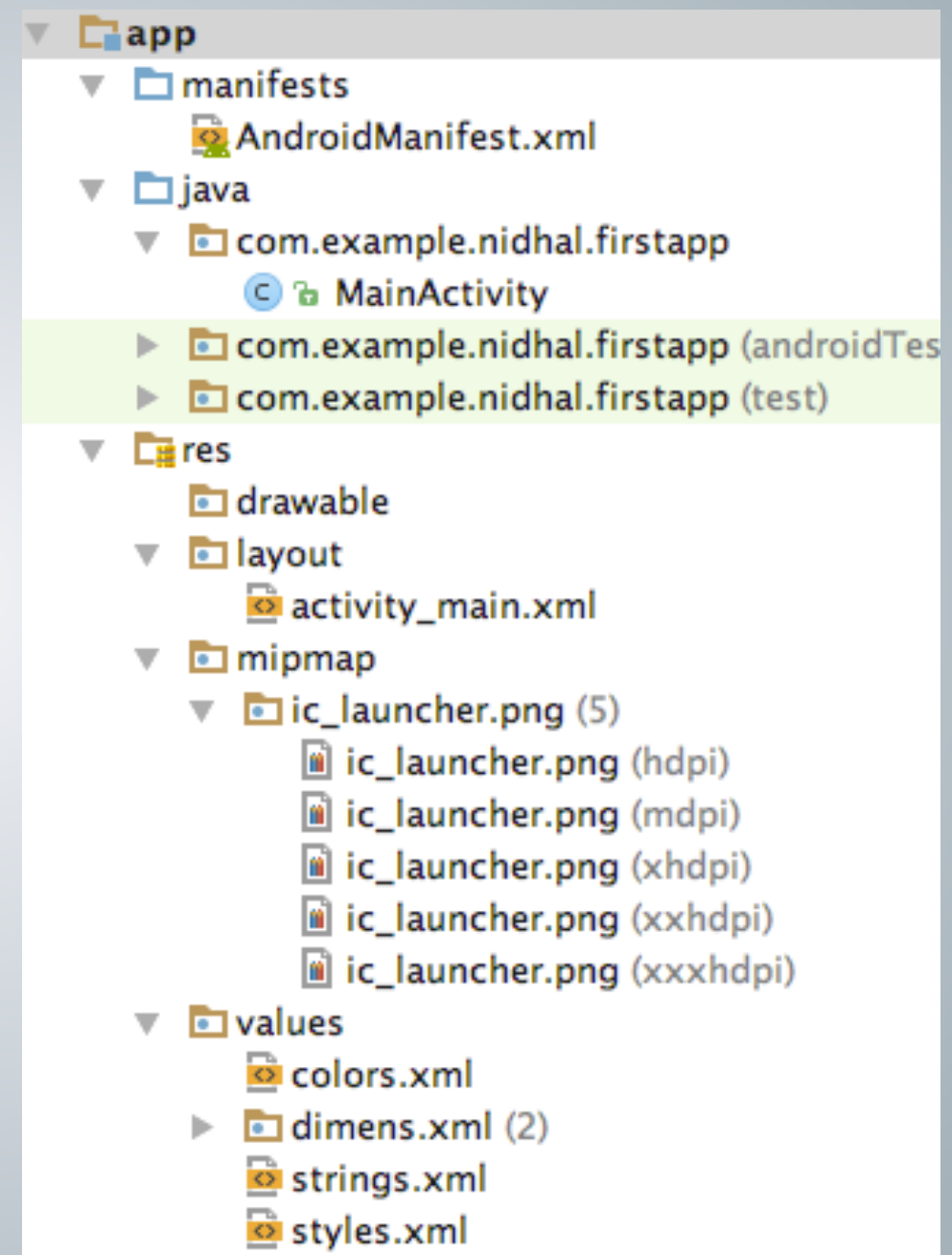
- Lent au démarrage à l'usage
- Ne contient pas les fonctionnalités suivantes :
 - Appareil Photo
 - Vibreur
 - Appels réels
 - Capteurs
 - Connexion USB
 - Evolution de la charge de la batterie

Périphérique physique

- Il est préférable d'utiliser un périphérique physique (au lieu de l'AVD) pour faire tourner vos applications.
- Activez le mode debug du périphérique
 - Appuyez plusieurs fois de suite sur le menu « A Propos » pour faire apparaître le menu caché développeur.
- Apparez votre périphérique en USB
 - Windows et Linux : Installer le driver de l'app.
 - Mac OS : Aucune manipulation.

Arborescence d'un projet

- **AndroidManifest.xml** : définit le comportement de votre application au système Android. Ce fichier définit par exemple le nom, l'icône, la version min du SDK, les activités, les services, etc.



Arborescence d'un projet

- **Layout** : le SDK Android offre une technique de création d'interfaces graphiques à l'aide de fichiers XML. C'est dans ce dossier que vous inclurez l'ensemble des fichiers décrivant vos interfaces.
 - **activity_main.xml** : le fichier principal de votre interface.

Arborescence d'un projet

- **Values** : Dossier contenant les fichiers décrivant les valeurs utilisées dans l'application.
 - **Strings.xml** : fichier qui contient vos déclarations de chaînes de caractères.
- **mipmap** : Icônes de lancement de l'application (classées par résolution d'écran).
- **drawable** : contient toutes les images, bitmaps dont vous avez besoin pour votre application en basse résolution.

AndroidManifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.nidhal.firstapp">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="FirstApp"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

AndroidManifest : Contenu

Contenu (1/2) :

- Précise le nom du package java utilisant l'application. Cela sert d'identifiant unique !
- Il décrit les composants de l'application
- Liste des activités, services, broadcast receivers
- Précise les classes qui les implémentent
- Précise leurs capacités (a quels intents ils réagissent)
- Ceci permet au système de savoir comment lancer chaque partie de l'application afin de satisfaire au principe de réutilisabilité.

AndroidManifest : Contenu

Contenu (2/2):

- Définit les permissions de l'application
- Droit de passer des appels
- Droit d'accéder a Internet
- Droit d'accéder au GPS,...
- Précise la version d'Android minimum nécessaire
- Déclare les librairies utilisées

AndroidManifest : Conventions

Conventions :

- Seuls deux éléments sont obligatoires
- `< manifest >` : contient le package, la version... Englobe tout le fichier
- `< application >` : décrit l'application et contiendra la liste de ses composants.
- Les données sont passées en tant qu'attribut et non en tant que contenu
- Tous les autres attributs commencent par "android:"

AndroidManifest : Ressources

Les ressources

- Au lieu de contenir les données en tant que tel, le fichier manifest peut faire appel a des ressources
- `< activityandroid : icon ="@drawable=smallPic"::: >`
- Ces ressources sont définies dans le répertoire "res" de l'application.

AndroidManifest : Permissions

Permissions (1/2) :

- Une application ne peut pas utiliser certaines fonctionnalités sauf si c'est précisé dans le fichier Manifest
- Il faut donc préciser les permissions nécessaires grâce à :< uses - permission >
- Il est possible de définir ses propres permissions

AndroidManifest : Permissions

Permissions (2/2) :

- Il existe des permission standard :
 - android.permission.CALL EMERGENCY NUMBERS
 - android.permission.READ OWNER DATA
 - android.permission.SET WALLPAPER
 - android.permission.DEVICE POWER

Liens utiles

- <http://www.androidviews.net>
- <http://www.android-app-patterns.com/>
- <http://developer.android.com>

Training

- 5 Units
- 14 + 1 Lessons
- 14 semaines => 3 semaines par 1 Unit
- Un QCM à la fin de chaque Unit