Université Mohamed Boudiaf de M'Sila
Faculté des Mathématique et de l'Informatique
Département d'Informatique

Programmation mobile

Chapitre II

Samir Akhrouf

Année universitaire
2019-2020

Contenu

2. Système d'exploitation Androïde

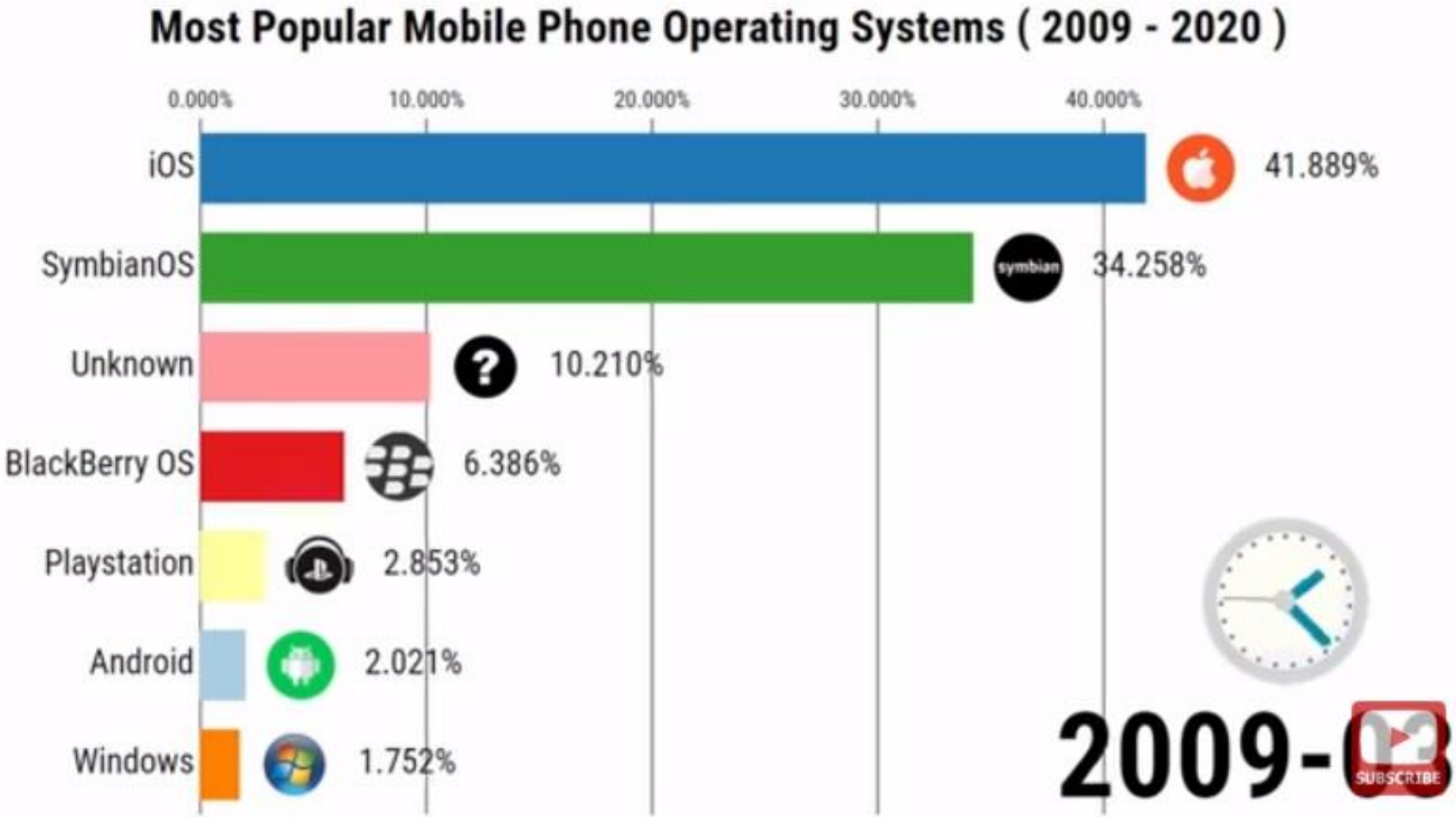
- Historique
- Description (Interface, Applications)
- Développement (Linux, gestion de la mémoire, mise à jour, communauté Open source)
- Sécurité et confidentialité (Sandbox, Permissions, …)
- Licence
- Copyrights et brevets

2.1. Introduction:

Objectifs:

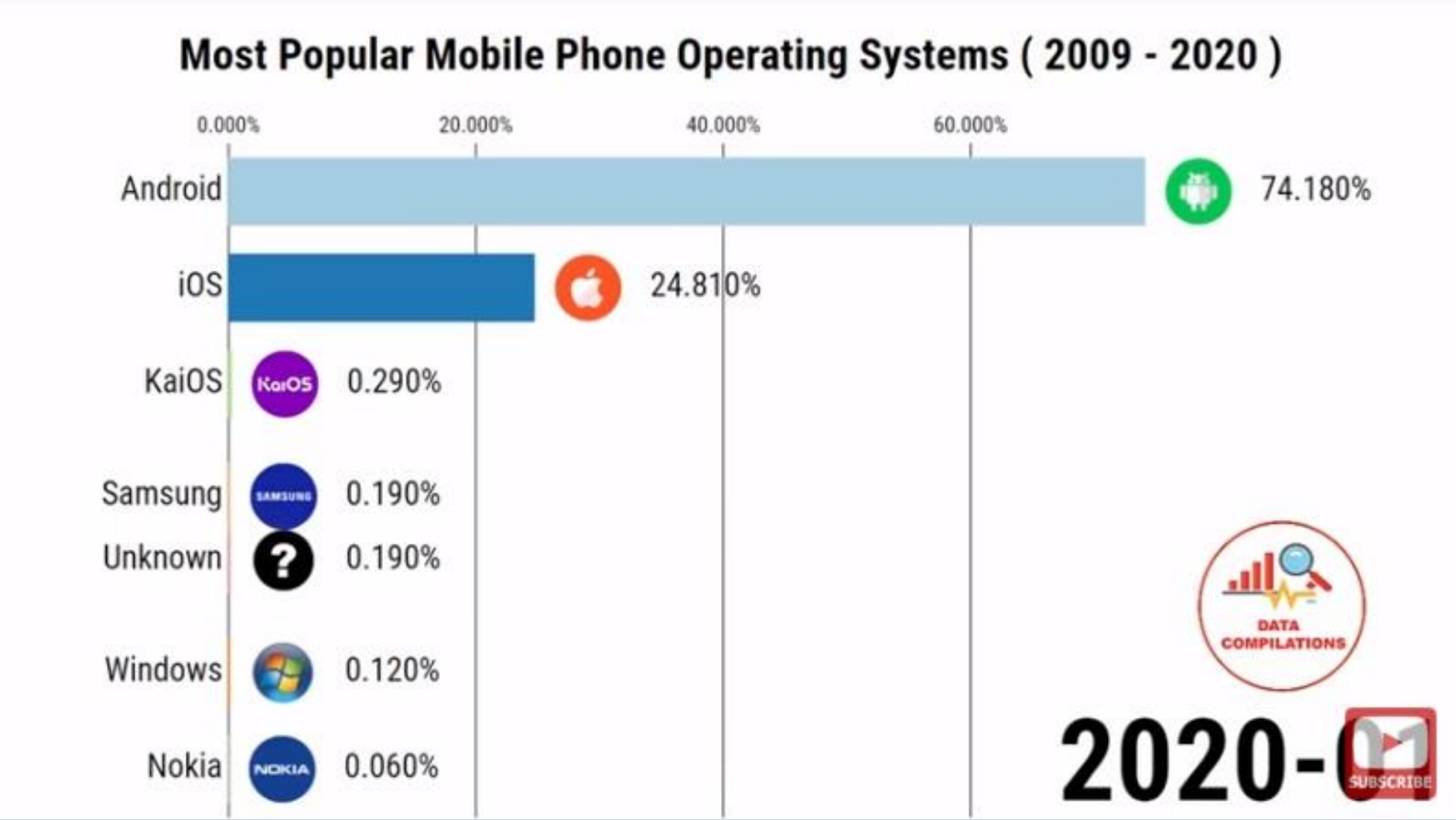
- présenter les systèmes d'exploitations mobiles;
- découvrir les plateformes de développement des applications mobiles;
- maîtriser la chaine de dev apps mobiles

Situation des Systèmes D'exploitations Mobiles Q3-2009



Source : https://www.youtube.com/channel/UCcnr1DqBRaTJfJVy0lV70cA?reload=9

Situation des Systèmes D'exploitations Mobiles Q1-2020



Source : https://www.youtube.com/channel/UCcnr1DqBRaTJfJVy0lV70cA?reload=9

Qu'est ce qu'Android ?

- **Un système d'exploitation orienté mobiles**
  - ensemble de logiciels qui sert d'interface entre le matériel (les composants du téléphone, d'une tablette, d'une montre, …) et les logiciels applicatifs (ceux que vous allez développer).
- **Un système d'exploitation open source**
  - Disponibilité du code,
  - Importante communauté d'utilisateurs et de développeurs
- **Un système d'exploitation basé sur le noyau Linux**
- **Environnement de développement gratuit**
  - Programmation en Java ou en Kotlin (ou autres …)
  - IDE (SDK et NDK Android) en évolution

2.2. Historique:

- Android est issu du travail de la startup américaine du nom d'Android Incorporated, 2003.

- Rachetée en 2005 par Google, ayant pour objectif de développer un système d'exploitation interactif.

- Lancement de l'iPhone en 2007 par Apple,
- véritable révolution, iOS moderne, bien en avancé sur la technologie actuelle.

- Création en Novembre 2007 de l'OHA (Open Handset Alliance)

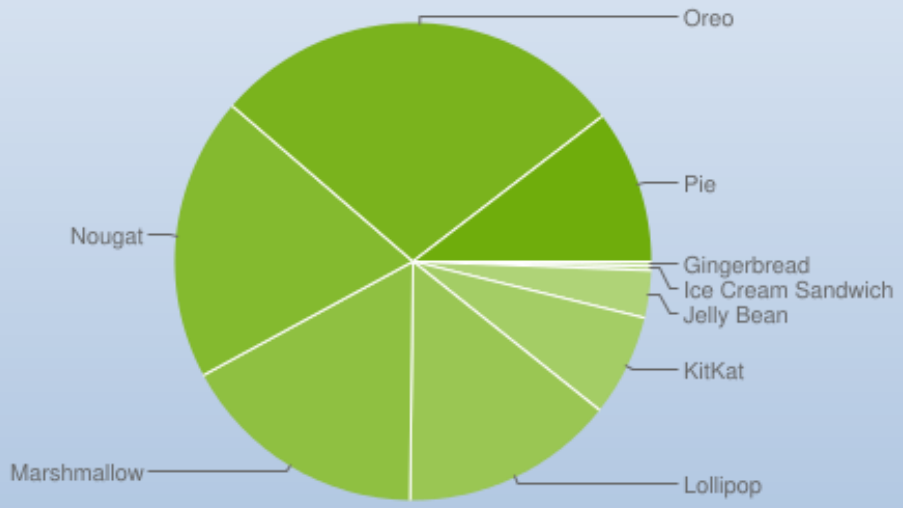
Plusieurs entreprises contribuent à Android à travers l'OHA

- Sortie du premier smartphone Android en Octobre 2008

- Début du succès à partir de 2011, avec la version 3.0

Versions du système

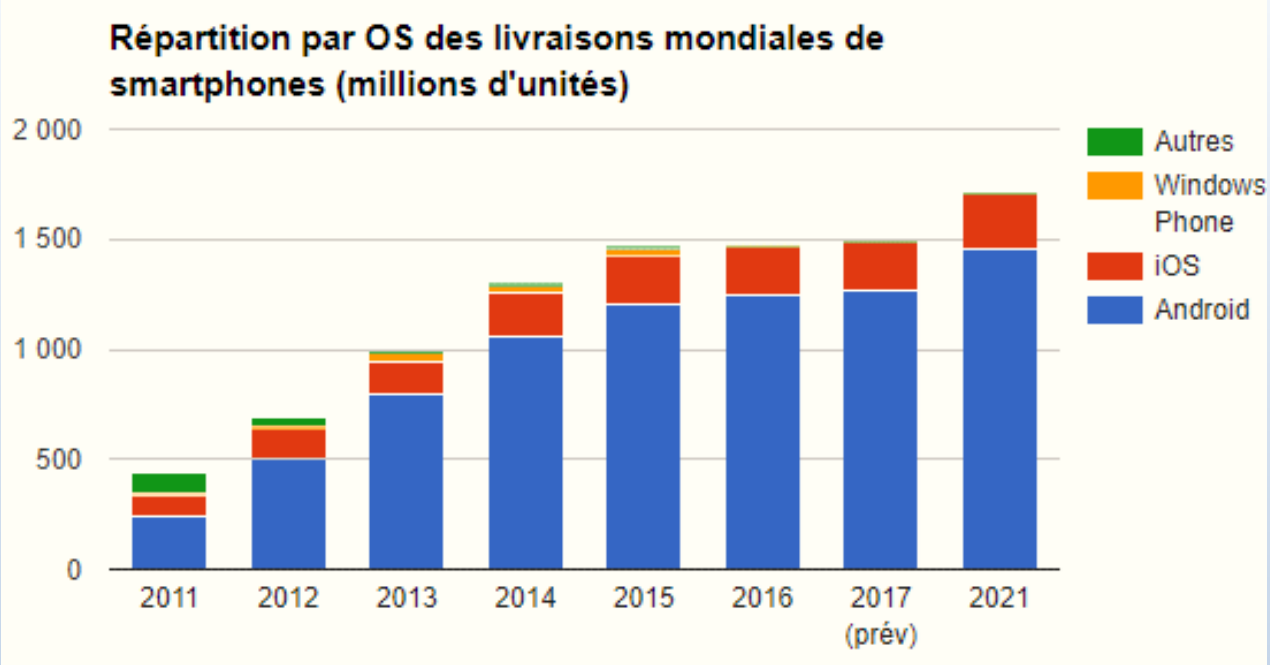
Version	Code Name	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.3%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.3%
4.1.x	Jelly Bean	16	1.2%
4.2.x		17	1.5%
4.3		18	0.5%
4.4	KitKat	19	6.9%
5.0	Lollipop	21	3.0%
5.1		22	11.5%
6.0	Marshmallow	23	16.9%
7.0	Nougat	24	11.4%
7.1		25	7.8%
8.0	Oreo	26	12.9%
8.1		27	15.4%
9	Pie	28	10.4%

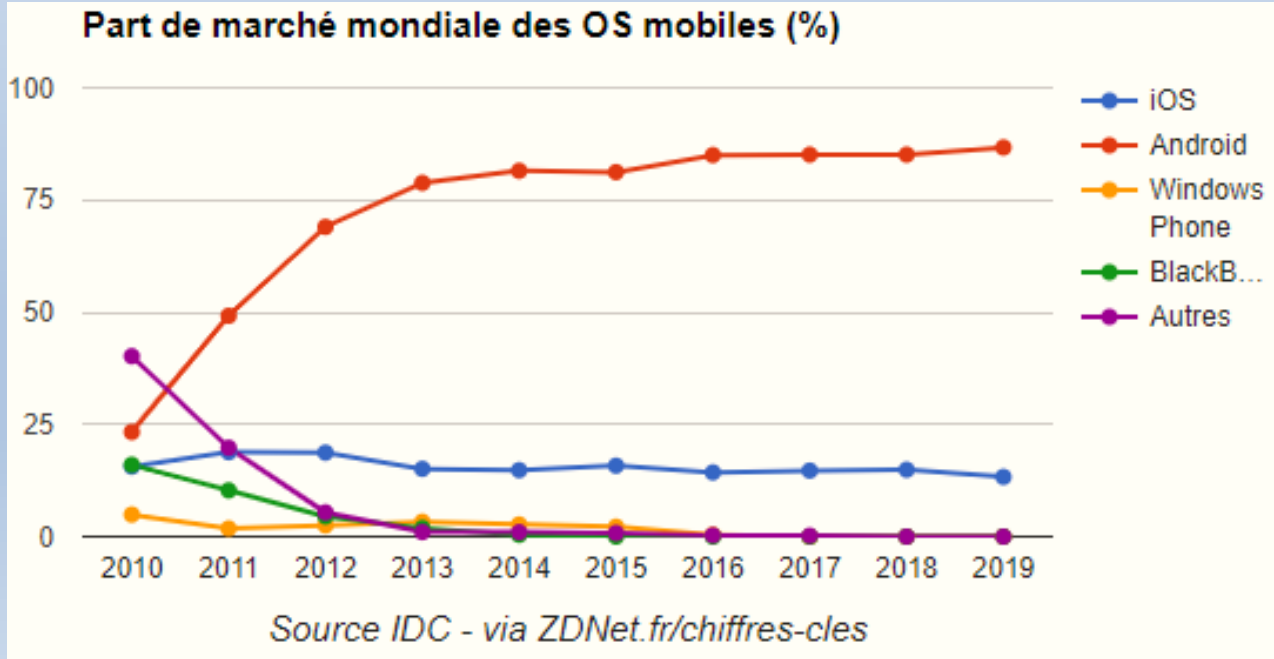


Data collected during a 7-day period ending on May 7, 2019.

Source : http://developer.android.com/about/dashboards/index.html

Les systèmes d'exploitation Android





L' Architecture Android

Une application mobile s'exécute sur un support matériel mobile:

- ressources limitées

  batterie (énergie), interface graphique, CPU, périphériques d'IO, …
- Supports physiques (matériels) très divers
- De très élémentaire au très évolué
- Utilisation ubiquitaire
- Ubiquité géographique
- Ubiquité des utilisateurs

L' Architecture Android

- Android est conçue pour des appareils mobiles au sens large
  - Téléphones mobiles, tablettes, ordinateurs portables, bornes interactives, baladeurs, montres, téléviseurs, voitures, ...

- La plate-forme Android est composée de différentes couches

  Un noyau Linux permettant des caractéristiques multitâches

  Des bibliothèques graphiques, multimédias

  Une machine virtuelle Java open-source : la Davik Virtual Machine

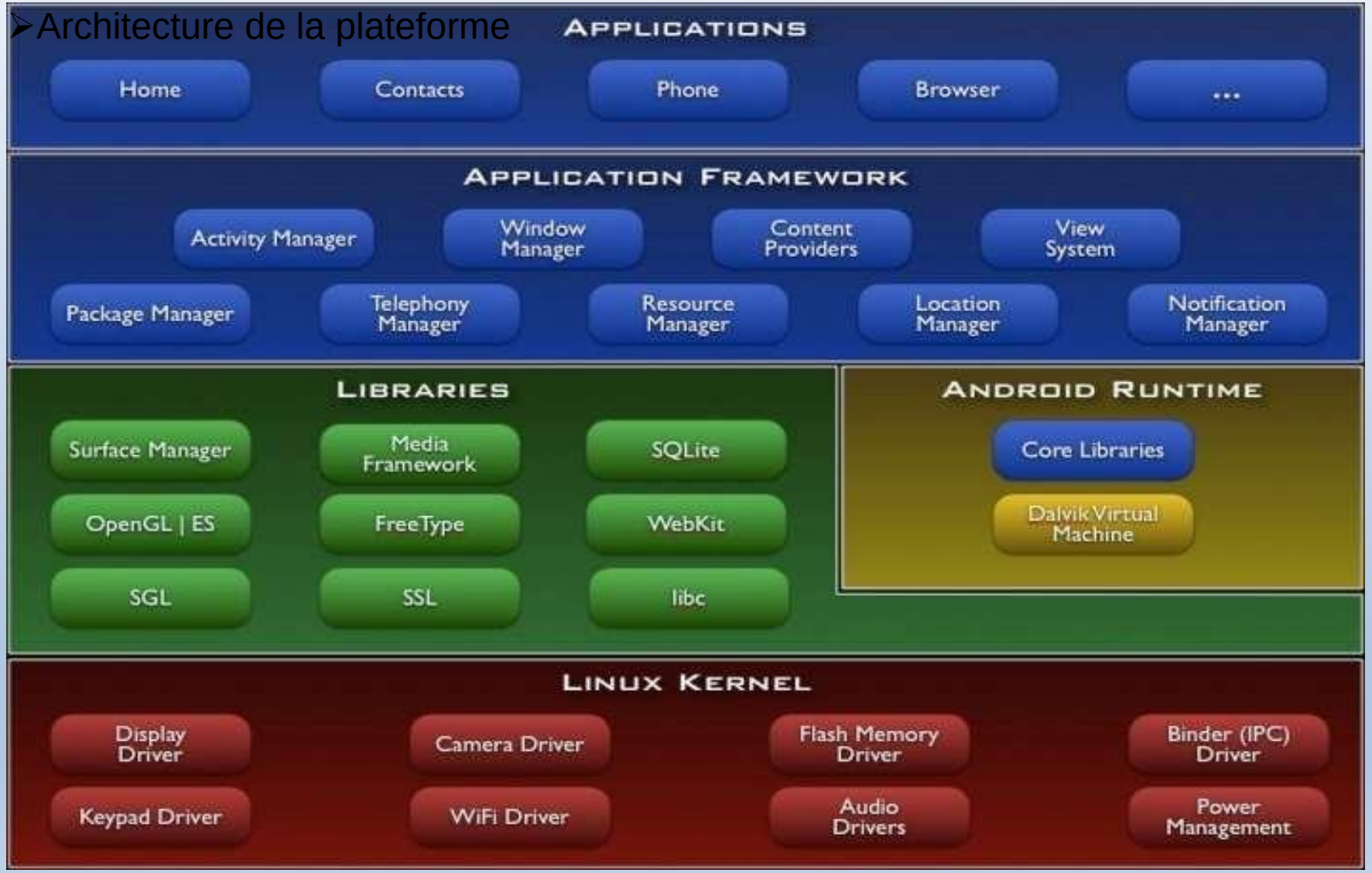
  Il existe un framework natif permettant le développement en C/C++ NDK (Native Development Kit)

  Un framework applicatif proposant des fonctionnalités de gestion de fenêtres, de téléphonie, de gestion de contenu...

  Des applications dont un navigateur web, une gestion des contacts, un calendrier...

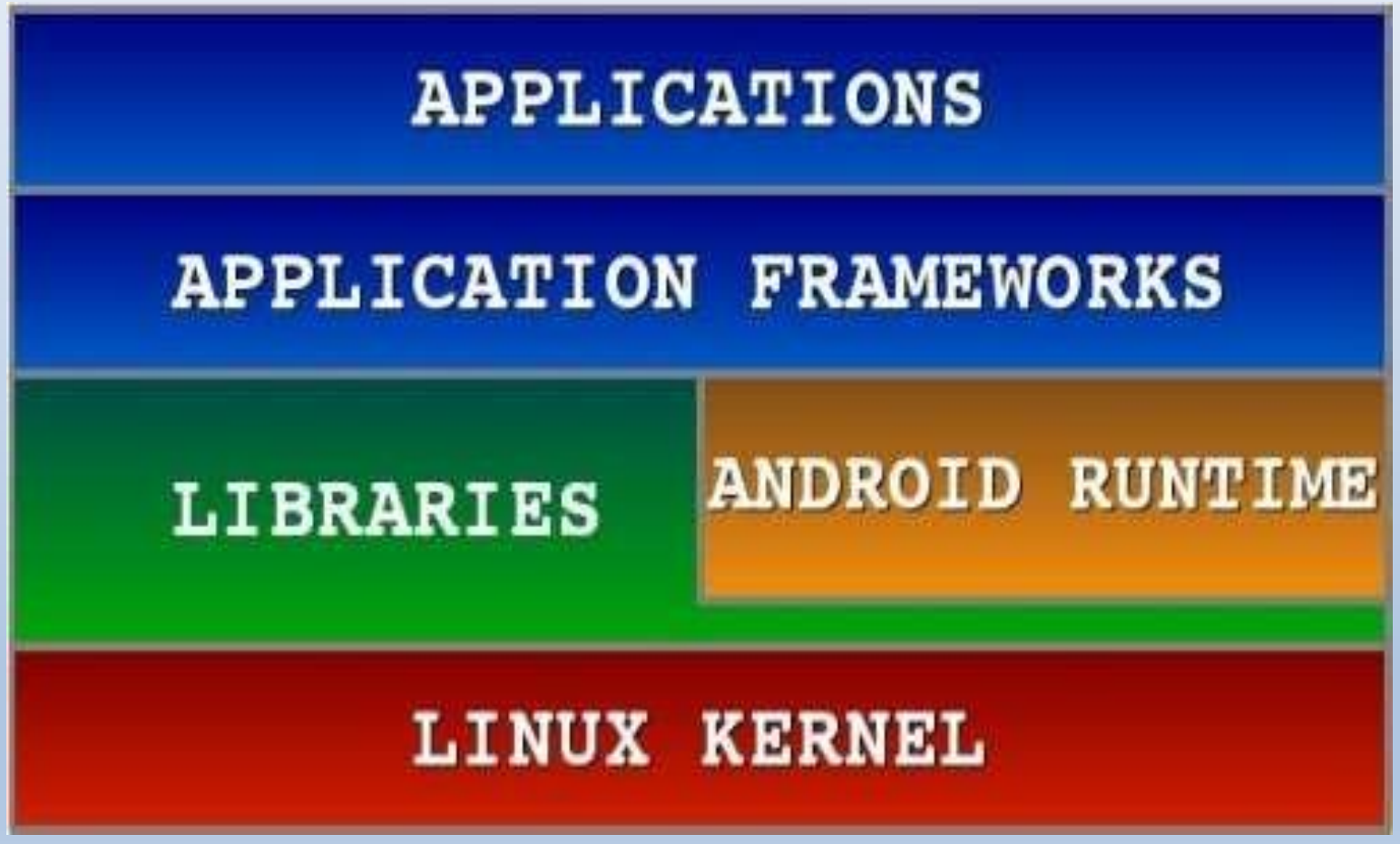
Architecture de la plateforme

- Architecture de la plateforme



1- Pile logicielle de plusieurs couches (layers)

4
3
2
1



Linux Kernel

- Gestion de la mémoire
- Gestion des processus
- Gestion du matériel (écran, clavier, …)
- La pile réseaux (IP)
- Gestion des capteurs (GPS, accéléromètre, …)
- Driver
- Sécurité
- Fournit une couche abstraite

Libraries

- Libc: c standard lib.
- SSL: Secure Socket Layer
- SG: 2D image engine
- OpenGL|ES: 3D image engine
- Media Framework: media codecs
- WebKit: Kernel of web browser
- FreeType: Bitmap and Vector
- SufraceManager: Compose
- Window manager with off-screen
- OpenGL|ES: 3D image engine
- SQLite: Database engine
- Buffering.

Runtime

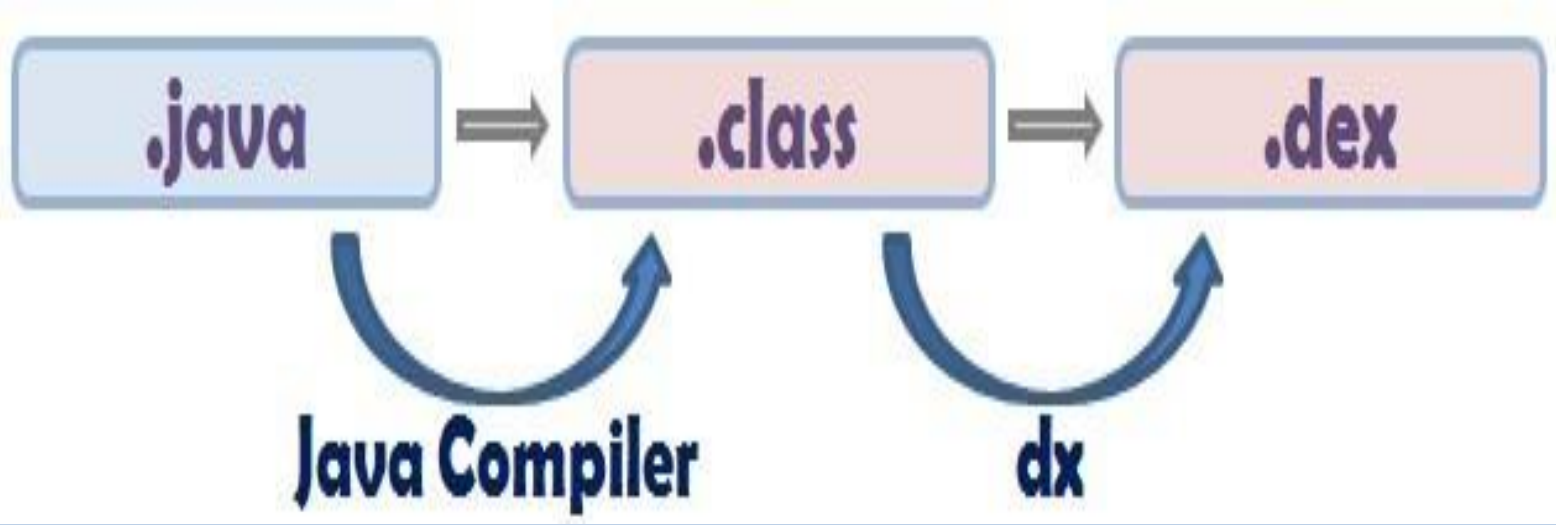
- Core Libraries
  - ❑ APIs
    - Data Structures
    - Utilities
    - File Access
    - Network Access
    - Graphics
    - Etc
  - ❑ Dalvik VM

Runtime

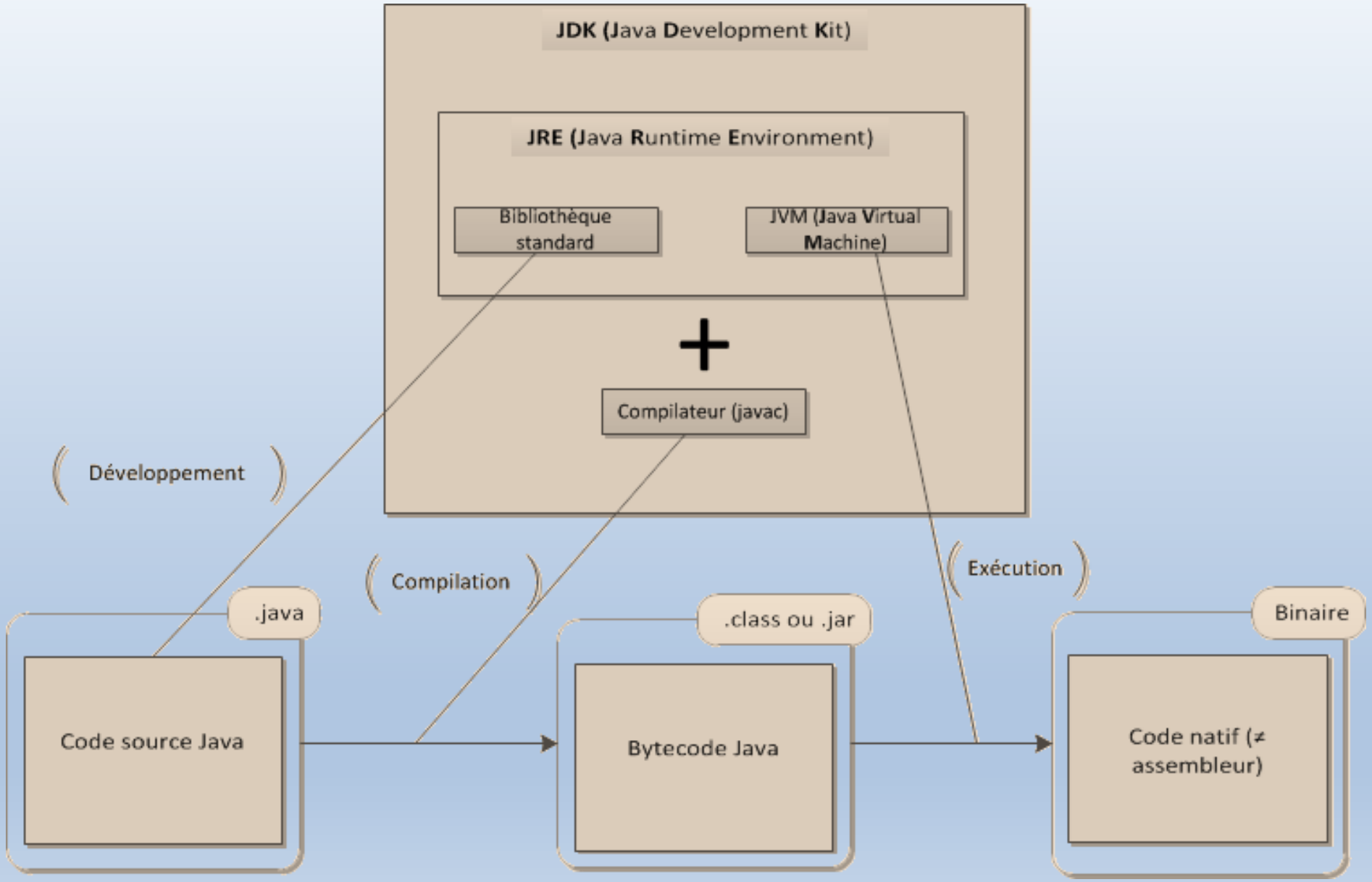
- C'est un environnement virtuel dans lequel s'exécute chaque application.

- Chaque application Android s'exécute avec son propre processus et son propre environnement.
- Le système assure le bon fonctionnement de plusieurs Dalvik VM.
- VM base Registres

Android : Production d'une application (compilation)



DX : Outil de conversion des fichiers .class en .dex fichiers



Source: https://openclassrooms.com/fr/courses/2023346-creez-des-applications-pour-android/2029414-larchitecture-dandroid#/id/r-2035317

Runtime

- Dalvik virtual machine
- Exécute un exécutable sous le format (.dex)
  - .dex est optimisé pour supporter des espaces mémoire faible
- Tout est base sur un noyau Linux, en particulier: Threading
- Gestion mémoire bas niveau

Application Framework

- Permet la reutilistion des composants :Les developpeurs ont le plein accès à tous les API de la framework utilisée par les applications de base (core apps).
- les devs peuvent remplacer ces composants.

Application Framework

- Caractéristiques

	Rôle
View	Utilisé pour construire une application. Ceci inclut : lists, System grids, textboxes, buttons, web browser
Content Provider	Permet à une app d'accéder à des données externes et de partager les siennes avec d'autres applications.
Resource	Permet d'accéder à des ressources telles que (strings, Manager graphics, and layout files)
Notification Manager	Gestionnaire des alertes
Activity Manager	Gère le cycle de vie d'une fenêtre
Telephony manager	Gère tous les appels téléphoniques.
Locations manager	Gère les localisations GPS, opérateur ou Wifi

Applications

- Nos applications appartiennent à cette couche.
- Fournit un ensemble d'applications de base (pré installées par google):
  - Email Client
  - SMS Program
  - Calendar
  - Maps
  - Browser
  - Contacts
  - Etc.

- Toutes ces apps sont écrites en Java.
- En tant que développeurs, on peut remplacer n'importe qu'elle app.

Outils de développement

- SDK
- AVD
- Android Studio
- Schéma de compilation

SDK Android

- **Software Development Kit Android**

- **Ensemble d'outils nécessaires pour créer une application Android**
  - ❑ aapt – Android Asset Packaging Tool (gérer *.APK)
  - ❑ adb – Android Debug Bridge (déploiement de l'application)
  - ❑ ddms – Dalvik Debug Monitor Service : Débogage de l'application
  - ❑ D'autres outils ...

- Il existe autant de version de SDK que de version d'Android

AVD

- **Android Virtual Device**
- **Permet de gérer les émulateurs**
  - ❑ Un émulateur est un dispositif mobile virtuel exécutant Android
  - ❑ Possible de créer autant de configurations voulues (taille écran, version Android, architecture matérielle, …)
- **Sachant que l'émulation consomme beaucoup de ressources**

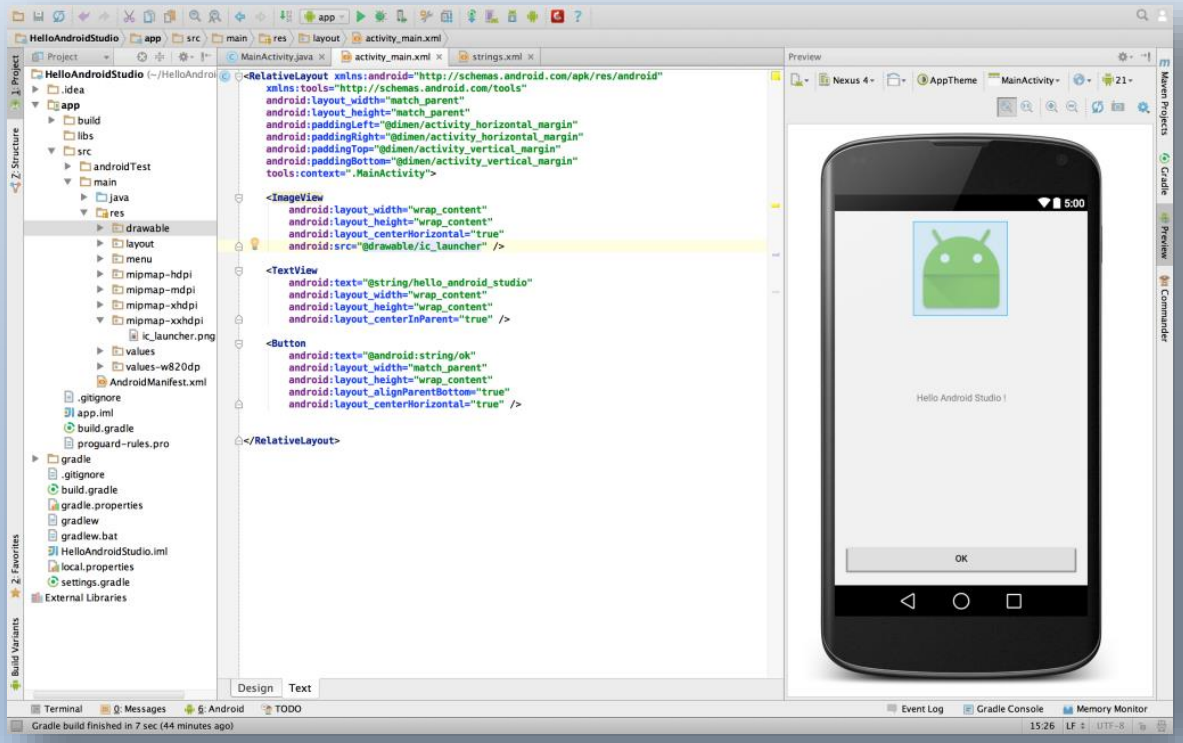
Android Studio

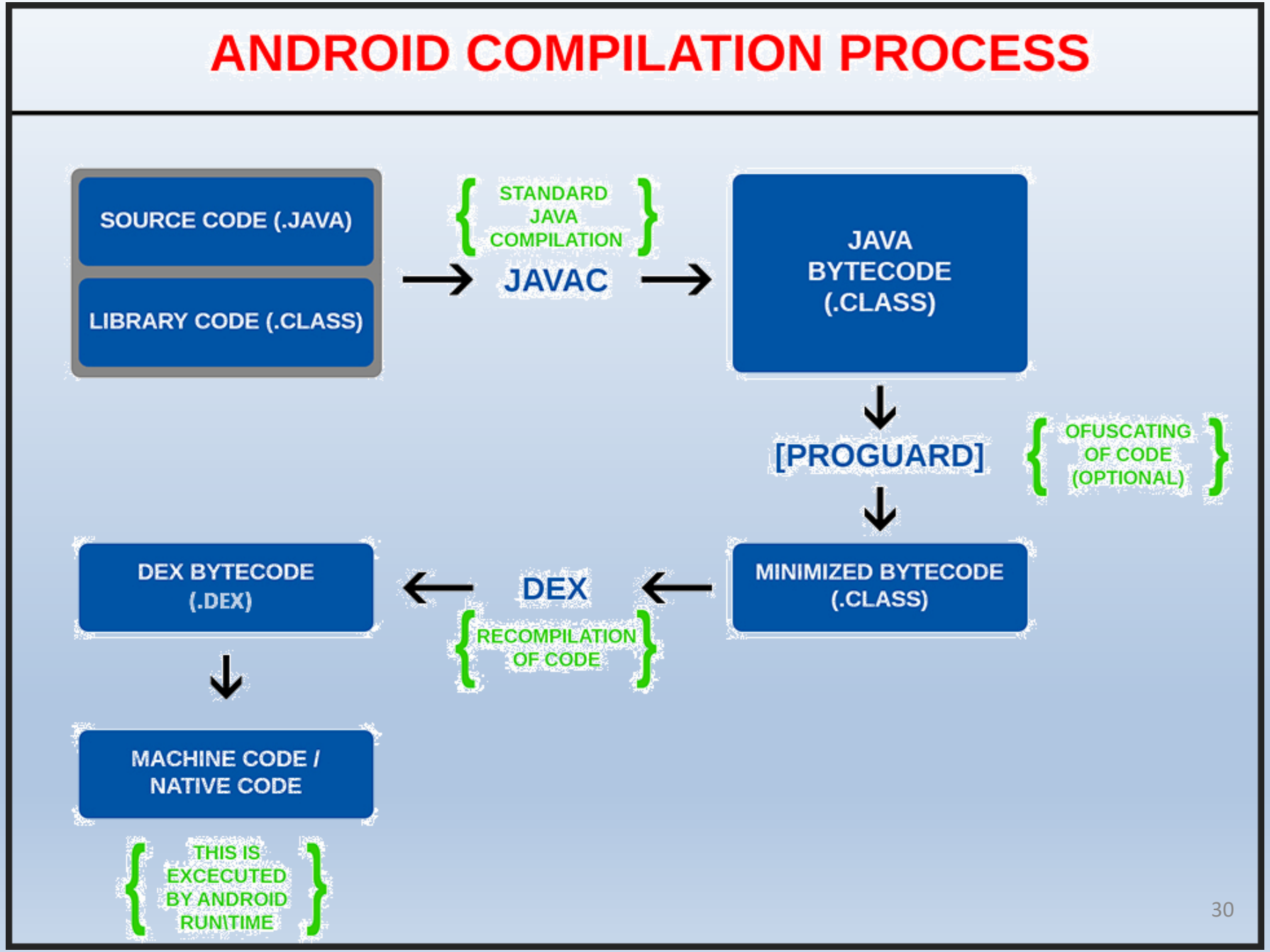
- **IDE basée sur Intelli J**

- **Développée par Google**
  - Vrai support sur le long terme
  - Dernières technologies
  - Parfaite intégration des outils SDK et AVD
  - Gestion complète d'un projet

- **Contient un éditeur de mise en page avec prévisualisation**

Android Studio




ANDROID COMPILATION PROCESS
SOURCE CODE (.JAVA)
LIBRARY CODE (.CLASS)
STANDARD JAVA COMPILATION
JAVAC
JAVA BYTECODE (.CLASS)
[PROGUARD]
OFUSCATING OF CODE (OPTIONAL)
MINIMIZED BYTECODE (.CLASS)
DEX
RECOMPILATION OF CODE
DEX BYTECODE (.DEX)
MACHINE CODE / NATIVE CODE
THIS IS EXCECUTED BY ANDROID RUN\TIME

Architecture d'un projet

- Le manifeste
- Le code Java
- Les ressources

❑ Le Manifeste

- **Il contient une partie de la configuration de votre projet**
  - ❑ Déclare ce que l'application contient (activités, services, …)
  - ❑ Précise comment ces composants sont liés à Android (que fait-on apparaitre dans le menu, …)
  - ❑ Précise les permissions de l'application (accès au réseaux, localisation, caméra, …)

❑ **Point de départ de toute application Android**

❑ Exemple

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".SeconActivity"></activity>
        <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>
```

Le code Java

- Permet de démarrer l'activité (l'application Android)
- Contient toutes les classes et la logique de votre application

Le code Java, exemple

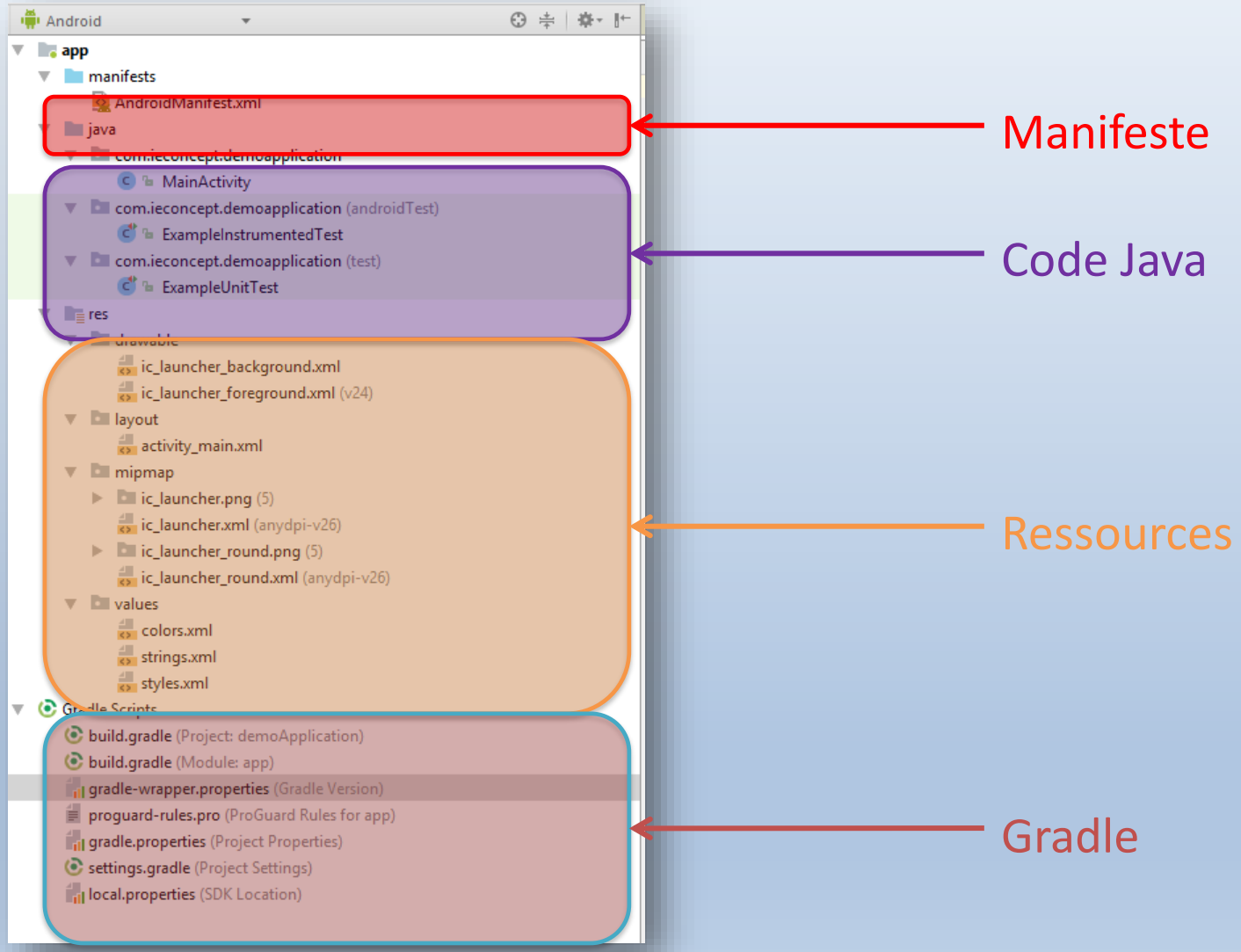
activity_main.xml | MainActivity.java | View.java | activity_secon.xml | SeconActivity.java | AndroidManifest.xml

```
1  package com.example.loginapp;
2
3  import ...
11
12 public class MainActivity extends AppCompatActivity {
13     private EditText Name;
14     private EditText Password;
15     private TextView Info;
16     private Button Login;
17     private int counter=5;
18
19
20     @Override
21     protected void onCreate(Bundle savedInstanceState) {...}
36     private void validate(String userName, String userPassword) {
37
38         if((userName.equals( "Admin" )) && (userPassword.equals( "1234" ))){...}  else {
42             counter--;
43             Info.setText(userName + " Number of Attempts Left: " + counter );
44             if (counter == 0) {...}
47         }
48         }
49
50 }
51
```

Les ressources

- **Contient toutes les ressources utiles à votre application**
  - « drawable » contient les images (JPG, PNG, …)
  - « layout » les descriptions XML de l'IHM (les layouts)
  - « menu » les descriptions XML des menus
  - « raw » les fichiers généraux (binaire BDD, CSV, …)
  - « values » les messages pour internalisation, les dimensions, les styles, …
  - « mipmap » icones de l'application

Représentation dans Android Studio



Sécurité dans le système Android

Chaque application est exécutée dans une sandbox (bac à sable) :

- Android est un système Linux multi utilisateur
- Chaque application est un utilisateur différent
- Chaque fichier de l'application n'est accessible que par elle
- Par défaut, chaque processus possède sa machine virtuelle
- Chaque application a son propre processus Linux

Sécurité des Mobiles: Objectifs

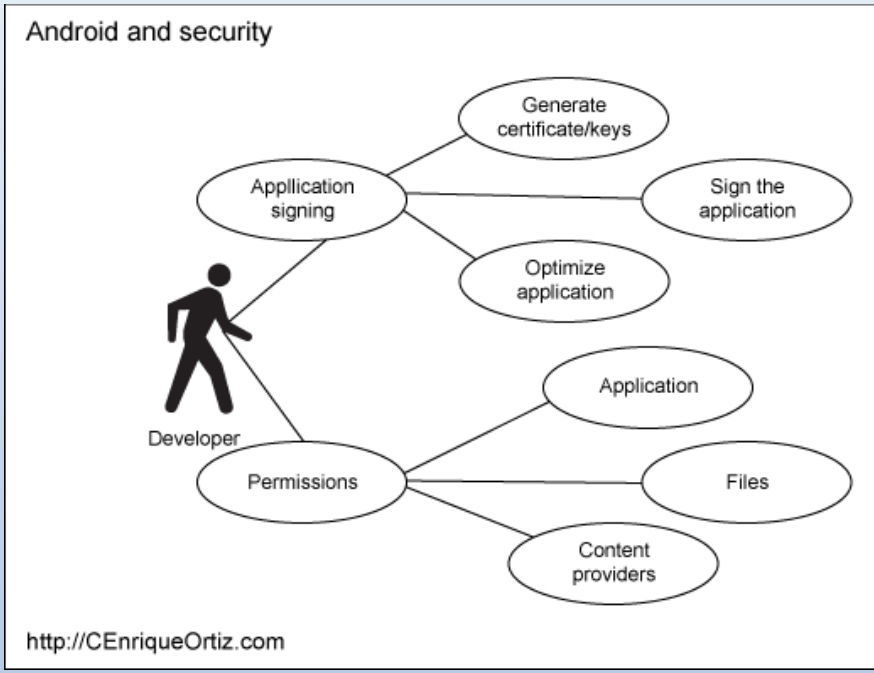
- Isolation/protection des Applications
  - Sandbox
  - Protection des données
- Protection des ressources embarquées
  - Accès au réseau cellulaire, répertoire d'adresses, GPS, caméra, comptes de réseaux sociaux,…
- Protection des données
  - Chiffrement des fichiers

Chiffrement du système de fichiers

- Les données utilisateurs (stockées dans la partition données utilisateur) peuvent être chiffrées selon le procédé dmcrypt (importés de linux)
  - Chiffrement AES en mode CBC, avec IV
  - La clé maitre est nommé DEK
  - La clé maitre est chiffrée avec une clé SEK, liée au mot de passe
- Chaque secteur (512 octets) est chiffré en mode AES128_cbc avec
  - IV= ESSIV (Encrypted Salt Sector Initialization Vector)
    - SALT=sha256(DEK)
    - ESSIV=AES256_ecb(key=SALT,sector_number)
  - AES128 key= DEK

Android: Sécurité

- La sécurité android repose sur trois piliers
  - Les Sandboxes Unix, associés aux processus
  - Les permissions
  - La signature des applications



Understanding security on Android

Enhance application security with sandboxes, application signing, and permissions

C. Enrique Ortiz, Developer and author, About Mobility Weblog

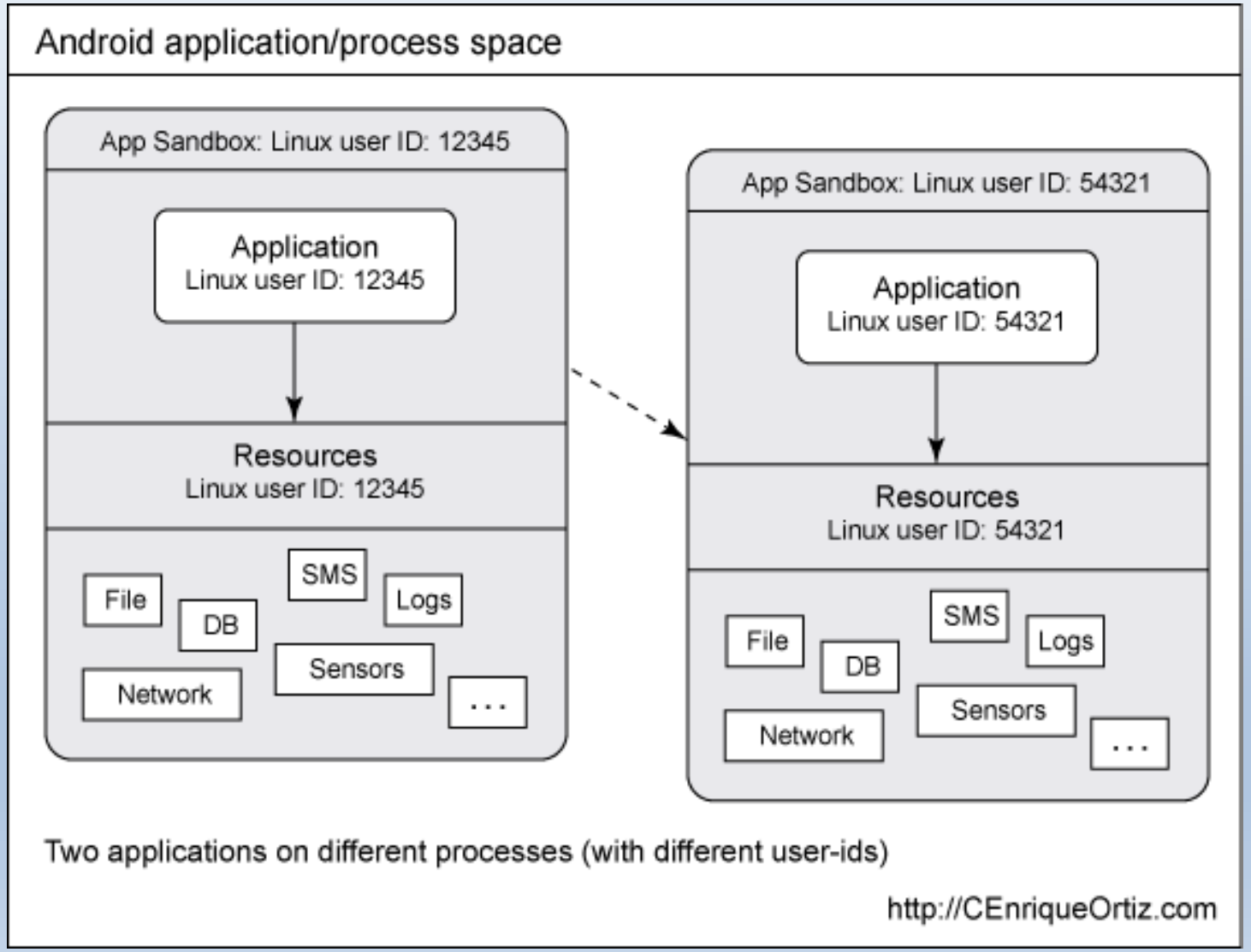
Tutorial: Building Secure Android Applications

http://siis.cse.psu.edu/slides/android-sec-tutorial.pdf

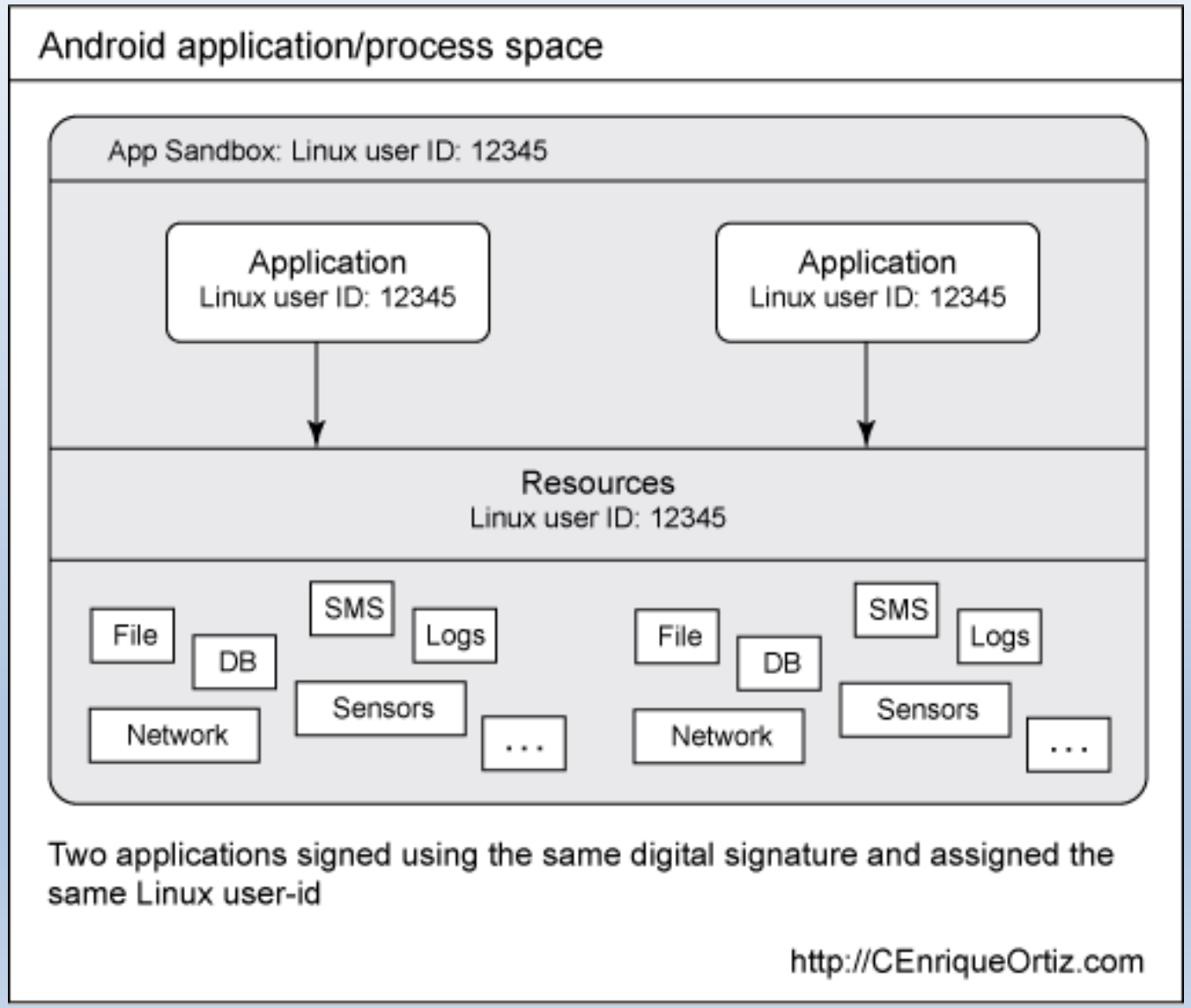
Application et User-Id

- Pour le noyau Linux associé à Android, une application possède un user-id, un id de groupe et un id de groupe secondaire.
  - Chaque application possède des droits d'accès en lecture, écriture et exécution.
  - C'est une politique de contrôle d'accès discrétionnaire (DAC).
- Le système Unix possède un compte root dont user-id est 0, et qui possède tous les droits.
- Chaque Application possède sa propre instance de Dalvik Virtual Machine (DVM)
- Deux applications signées par une même clé privée peuvent (optionnellement) utiliser un même UserID et donc partager un Sandbox identique.

Sandbox et Application User-id



Partage de processus



Permissions et Filtrage des Intents

- Le fichier Manifest.xml définit la structure d'une application, c'est-à-dire la liste de ses composants
  - Il contient également les demandes de permission d'accès aux ressources du mobile
  - Il décrit également les Intents traitées par l'application
- C'est le gestionnaire de la politique de sécurité de l'application

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    android:versionCode="16" android:versionName="1.1.12" package="org.chemlab.dealdroidapp"
    android:sharedUserId="androware.test">

    <uses-sdk android:minSdkVersion="4" android:targetSdkVersion="4"/>

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
    <uses-permission android:name="android.permission.VIBRATE" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <application android:icon="@drawable/icon" android:label="@string/app_name"
        android:process="androware.test">

        <!-- The app's UI is just a preferences page right now -->
        <activity android:name=".Preferences" android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
```

Signature des Applications

- Chaque application (.apk) doit être signée
- Sous Eclipse on utilise les outils
  - Keytool pour la génération de clés RSA dans un keystore
  - Jarsigner pour la signature d'une application
- Un certificat peut être auto-signé

Business Model des applications Android

- Le modèle des applications Android "gratuites" est la collecte d'information destinée à la diffusion de publicités ciblées.
- La collecte d'information et son exportation implique l'autorisation par l'utilisateur d'un certain nombre de permissions.
- Cependant des techniques de collecte dites "Zéro Permission" sont possibles.

La protection juridique d'une application mobile

- En 2017, plus de 175 milliards d'applications mobiles ont été téléchargées dans le monde.
- Face à ce marché en pleine expansion, de plus en plus de sociétés développent leurs propres applications mobiles.
- Ces dernières peuvent être gratuites ou payantes.
- Il est donc essentiel de savoir comment protéger, juridiquement ce type de création.

1. éléments constituant l'application mobile

Une application mobile peut être composée :

- d'éléments logiciels ;
- d'une ou plusieurs bases de données ;
- d'un nom et d'un logo ;
- d'une interface graphique et de contenu multimédia ;
- de musique.

La plupart de ces éléments, peuvent bénéficier de la protection du droit d'auteur.

** Certains éléments sont protégés par des régimes de protection spécifiques comme le droit *sui generis* du producteur de base de données.

2.Le régime de protection des éléments composant une application mobile

A. La protection du logiciel

- Pour être considéré comme original et bénéficier de la protection du droit d'auteur, un logiciel doit résulter d'un effort personnalisé portant la marque de l'apport intellectuel de son auteur.
- Pour bénéficier de la protection par le droit d'auteur, le logiciel doit également être matérialisé, c'est-à-dire suffisamment élaboré. Cette condition exclut la protection des idées, concepts et algorithmes qui ne peuvent pas faire l'objet d'une appropriation. Il en va de même pour les fonctionnalités et le langage de programmation qui ne sont pas non plus protégeables par le droit d'auteur.
- Les éléments d'un logiciel susceptibles d'être protégés par le droit d'auteur sont le code source et le matériel de conception préparatoire.

B. La protection de la base de données

- Une base de données est susceptible de faire l'objet d'une double protection :
  - l'architecture de la base (contenant) peut être protégée par le droit d'auteur si elle est originale ;
  - le contenu de la base et les investissements substantiels engagés pour constituer, vérifier présenter la base peuvent être protégés par le droit *sui generis* des producteurs de bases de.

- Pour bénéficier de la protection par le droit *sui generis*, le producteur d'une base de données doit justifier de :
  - la réalisation d'un investissement financier, matériel ou humain ;
  - qui doit être substantiel ;
  - pour la constitution, la vérification ou la présentation du contenu de la base de données.

Dans le cadre de cette protection, le producteur de la base peut interdire l'extraction ou la réutilisation de tout ou partie de la base sous peine de sanctions.

C. La protection du nom et du logo

La dénomination et le logo d'une application mobile peuvent être protégés par le droit d'auteur s'ils sont originaux. Ils peuvent également être déposés à titre de marque auprès de l'INPI s'ils ont vocation à être exploités commercialement afin de distinguer les produits ou services du déposant de ceux de ses concurrents.

1. La protection des autres éléments

L'interface graphique, les contenus multimédias et la musique de l'application mobile sont aussi protégeables par le droit d'auteur sous réserve d'originalité.

Est-ce que les logiciels sont brevetables ?

- En général, l'auteur d'un logiciel peut obtenir la protection de son code source selon le droit d'auteur, qui appartient au droit de la propriété intellectuelle, et non au droit de la propriété industrielle (qui concerne les brevets).
- Néanmoins, de très nombreux brevets sont délivrés tous les ans sur des inventions dans le domaine de l'informatique, et une nouvelle classe a même été créée en 2017 pour les inventions qui concernent l'intelligence artificielle.
- En effet, les conseils en propriété industriels ont depuis longtemps trouvé des parades, en accords avec les offices de brevets, afin d'obtenir pour leurs clients une protection sur ces inventions digitales.

Comment breveter un logiciel ?

- Le brevet est un titre de propriété industrielle qui permet la protection d'un **produit**, d'une **utilisation**, d'**un procédé**, une **solution technique** qui doit, entre autres, être inventive et non évidente pour un homme du métier.
- C'est pourquoi un programme d'ordinateur peut tout à fait faire l'objet d'un brevet, par exemple sous la forme d'un **procédé mis en œuvre par un ordinateur**. Ce n'est donc pas le code source du programme lui-même qui est protégé par le brevet, mais ses **fonctions** dans une solution plus globale.

Disposer d'un brevet pour une solution qui utilise une application mobile

- En janvier 2019, l'office des brevets des États-Unis (USPTO) a rendu publique une demande de brevet déposée par la société Uber.
- L'acteur mondial du VTC souhaite en effet breveter l'usage de l'intelligence artificielle, depuis son application mobile, pour détecter les passagers potentiellement ivres.
- Le croisement des données issues des comportements du client au moment de la commande (façon de taper sur les touches, orthographe douteuse, vitesse de marche…) permettrait alors d'anticiper les comportements propres à une personne en état d'ivresse. Tout cela serait traité en temps réel pour alerter les chauffeurs.

Cet exemple nous montre que **le brevet s'applique aux fonctions et l'utilisation du logiciel**. Le brevet permet donc de protéger la technique et non pas le programme informatique.

Source : https://yesmypatent.com/blog/fiches-pratiques-5/post/deposer-un-brevet-pour-une-application-un-logiciel-un-programme-54

Ressources & Références

- https://developer.android.com/about/dashboards/index.html
- https://developer.android.com/guide/platform/index.html
- https://developer.android.com/studio/index.html
- https://developer.android.com/
- https://perso.telecom-paristech.fr/urien/secmob2017.pdf
- https://www.bestcours.com/programmation/android/580-programmation-mobile-avec-android-pdf.html
- https://perso.univ-rennes1.fr/pierre.nerzic/Android/poly.pdf
- http://developer.android.com/guide/components/fundamentals.html
- https://github.com/viaforensics/android-encryption/blob/master/decrypt.py
- http://siis.cse.psu.edu/slides/android-sec-tutorial.pdf
- David Ehringer, The_Dalvik_Virtual_Machine.pdf
- https://www.wipo.int/export/sites/www/ip-development/fr/agenda/pdf/ip_and_mobile_applications_study.pdf
- https://www.app.asso.fr/centre-information/base-de-connaissances/les-grands-themes/applications-mobiles/focus-la-protection-juridique-dune-application-mobile

- https://inno3.fr/sites/default/files/2020-01/B.JEAN-OPEN%20SOURCE%20%20ET%20BREVETABILITE%20DES%20INVENTIONS%281%29.pdf
- https://yesmypatent.com/blog/fiches-pratiques-5/post/deposer-un-brevet-pour-une-application-un-logiciel-un-programme-54