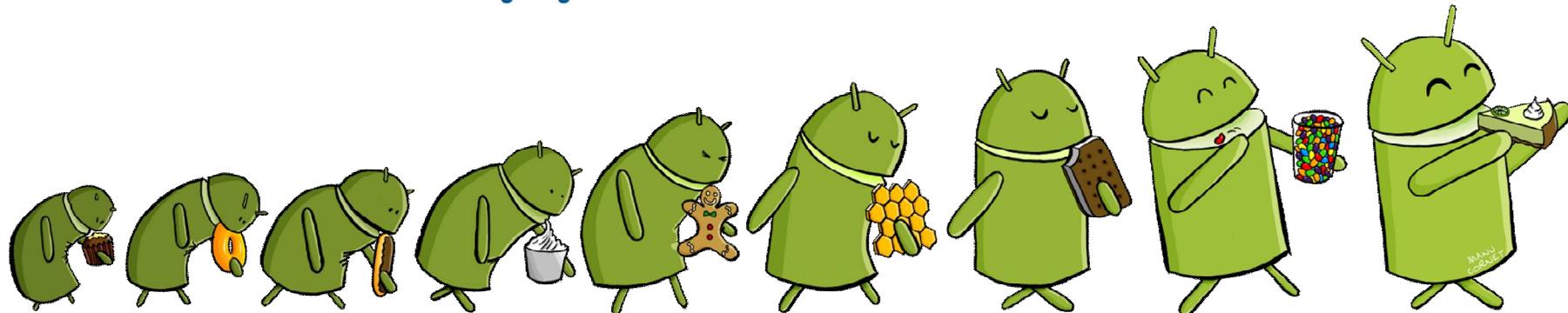


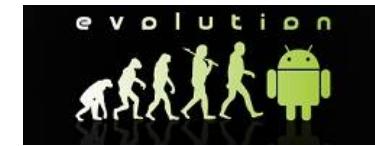
Introduction au développement Android



Janvier 2017

lionel.buathier@univ-lyon1.fr

Sommaire



- ▶ **1. Introduction**
- ▶ 2. Android de A à Z
- ▶ 3. La diversité des appareils Android
- ▶ 4. Les outils de développement
- ▶ 5. Architecture d'un projet Android
- ▶ 6. Les fondamentaux d'une application Android
- ▶ 7. Conclusion
- ▶ 8. Bibliographie



Introduction

► Les téléphones

- deviennent de plus en plus performants
- permettent de faire de nombreuses choses
- même de téléphoner^^
- sont de plus en plus nombreux et diversifiés

► Parallèlement, les OS mobiles évoluent vers plus

- de fonctionnalités,
- de fluidité,
- d'ergonomie.



Les types de développement mobile

- ▶ Développement d'applications natives sur les différents systèmes d'exploitation :
 - Android / iOS / Windows Phone
- ▶ Développement de sites Web "Responsive Design"
 - l'affichage s'adapte à la taille de l'écran
- ▶ Développement d'applications en HTML5 pour les différents systèmes d'exploitation s'exécutant sur des navigateurs basés sur Webkit



Fonctionnalités en vogue

- ▶ Géo-localisation de l'utilisateur ou de différents points d'intérêt
- ▶ Partage d'informations avec les utilisateurs à proximité
- ▶ Intégration avec les plateformes sociales
- ▶ Reconnaissance des images prises avec la caméra
- ▶ Réalité augmentée
- ▶ Micro-transactions (NFC)

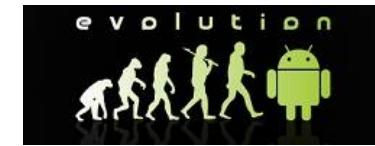


Un marché qui a ses propres enjeux

- ▶ **Courbe d'apprentissage importante** pour tous les rôles : architectes organiques et fonctionnels, architectes d'information, analystes, développeurs, etc.
- ▶ **Multiplication des appareils à supporter** tant pour le développement que pour les essais
- ▶ **Changements importants dans les systèmes** d'exploitation d'une version à l'autre qui peuvent créer des problèmes dans les applications
- ▶ **La qualité de nos applications est jugée rapidement par les utilisateurs**, une petite erreur peut être fatale



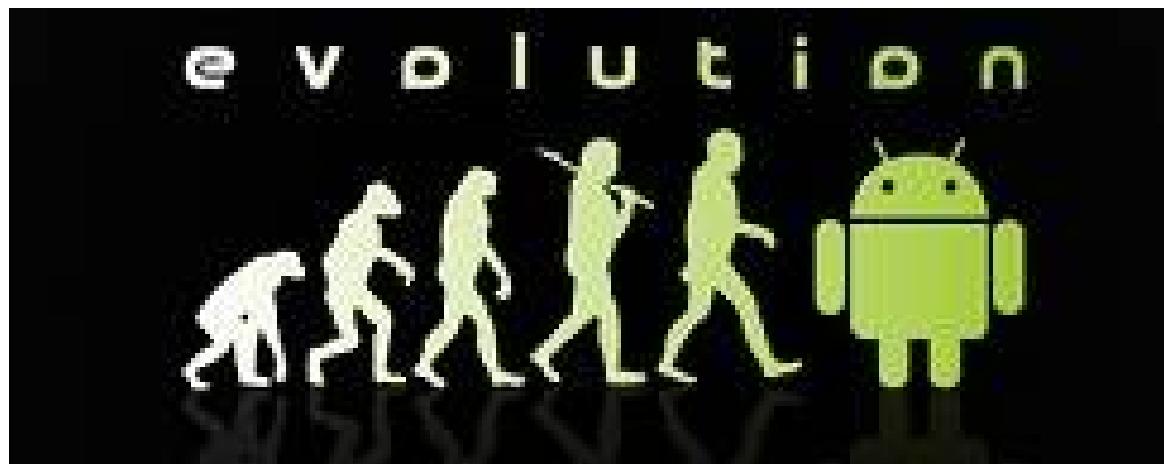
Sommaire



- ▶ 1. Introduction
- ▶ **2. Android de A à Z**
- ▶ 3. La diversité des appareils Android
- ▶ 4. Les outils de développement
- ▶ 5. Architecture d'un projet Android
- ▶ 6. Les fondamentaux d'une application Android
- ▶ 7. Conclusion
- ▶ 8. Bibliographie



Une petite histoire d'Android



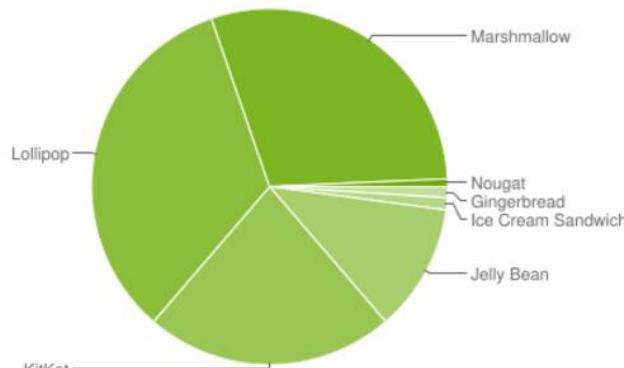
http://www.presse-citron.net/wordpress_prod/wp-content/uploads/2014/11/android-infographie.png

<http://i2.wp.com/www.androidetvous.com/wp-content/uploads/2014/09/android-infographie-1.png>



Répartition des différentes API (01/2017)

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	1.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.1%
4.1.x	Jelly Bean	16	4.0%
4.2.x		17	5.9%
4.3		18	1.7%
4.4	KitKat	19	22.6%
5.0	Lollipop	21	10.1%
5.1		22	23.3%
6.0	Marshmallow	23	29.6%
7.0	Nougat	24	0.5%
7.1		25	0.2%



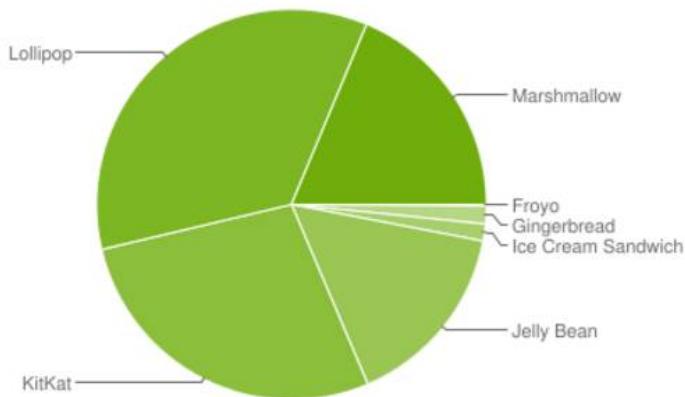
Data collected during a 7-day period ending on January 9, 2017.

- ▶ Les API < 2.2 n'apparaissent plus car ne sont plus compatibles avec Google Play.
- ▶ <http://developer.android.com/about/dashboards/index.html#Platform>
- ▶ <http://developer.android.com/about/dashboards/index.html>



Répartition des différentes API (09/2016)

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	1.5%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.4%
4.1.x	Jelly Bean	16	5.6%
4.2.x		17	7.7%
4.3		18	2.3%
4.4	KitKat	19	27.7%
5.0	Lollipop	21	13.1%
5.1		22	21.9%
6.0	Marshmallow	23	18.7%



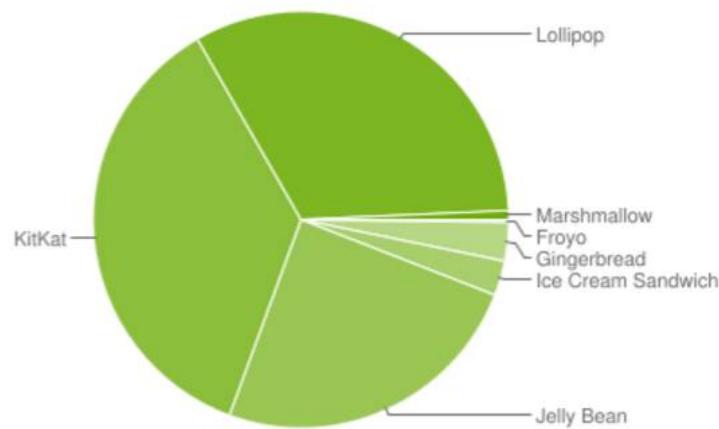
Data collected during a 7-day period ending on September 5, 2016.

- ▶ Les API < 2.2 n'apparaissent plus car ne sont plus compatibles avec Google Play.



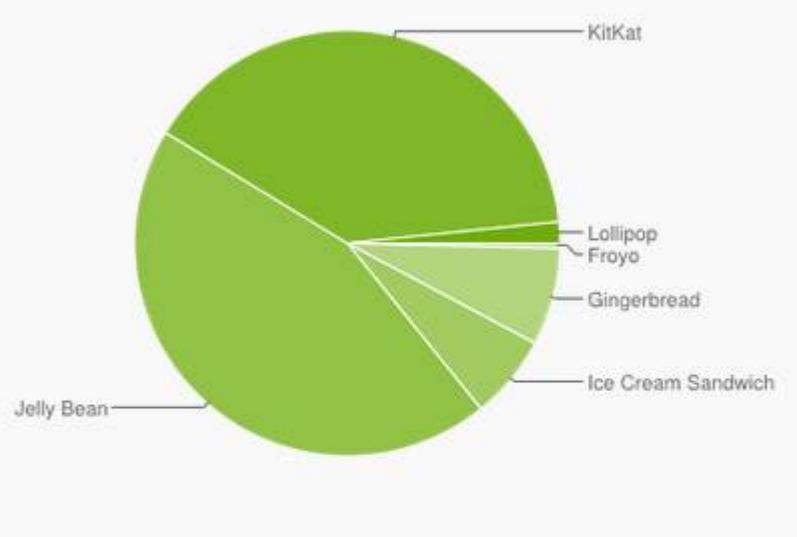
Répartition des différentes API (01/2016)

Version	Codename	API	Distribution
2.2	Froyo	8	0.2%
2.3.3 - 2.3.7	Gingerbread	10	3.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	2.7%
4.1.x	Jelly Bean	16	9.0%
4.2.x		17	12.2%
4.3		18	3.5%
4.4	KitKat	19	36.1%
5.0	Lollipop	21	16.9%
5.1		22	15.7%
6.0	Marshmallow	23	0.7%



Répartition des différentes API (02/2015)

Version	Codename	API	Distribution
2.2	Froyo	8	0.4%
2.3.3 - 2.3.7	Gingerbread	10	7.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	6.4%
4.1.x	Jelly Bean	16	18.4%
4.2.x	Jelly Bean	17	19.8%
4.3		18	6.3%
4.4	KitKat	19	39.7%
5.0	Lollipop	21	1.6%



Data collected during a 7-day period ending on February 2, 2015.

Any versions with less than 0.1% distribution are not shown.



Les types d'écrans tailles et densités

- ▶ Pour caractériser la partie affichage d'un appareil, le système Android définit un couple taille / densité d'écran :

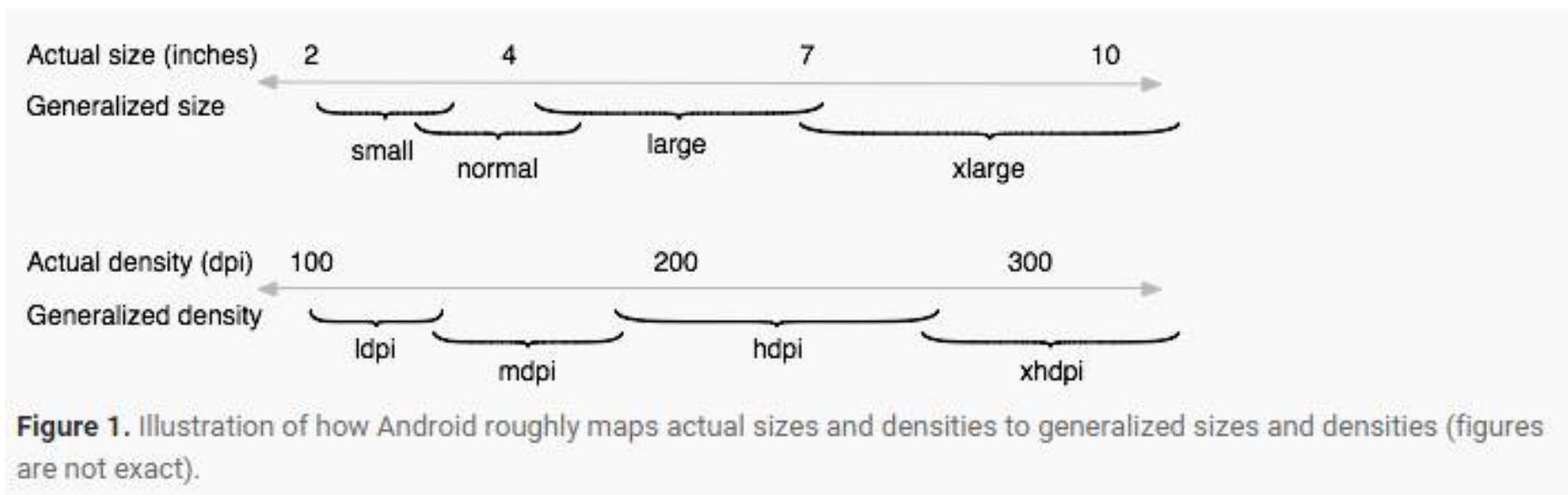


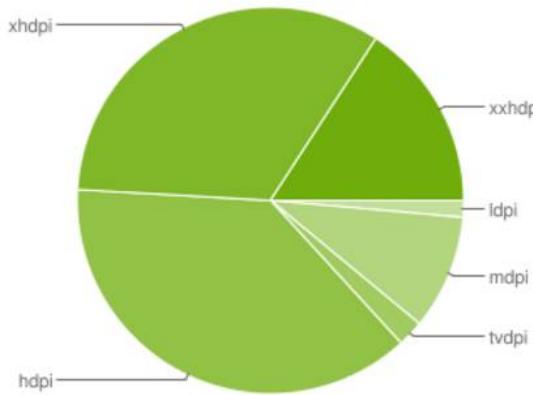
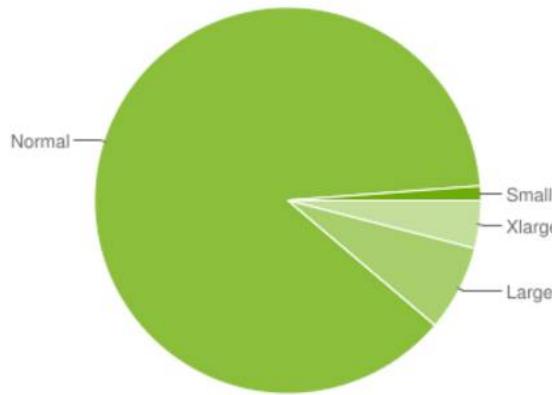
Figure 1. Illustration of how Android roughly maps actual sizes and densities to generalized sizes and densities (figures are not exact).

- ▶ http://developer.android.com/guide/practices/screens_support.html#range
- ▶ http://developer.android.com/guide/practices/screens_support.html



Répartition des types d'écrans (01/2017)

	ldpi	mdpi	tvdpi	hdpi	xhdpi	xxhdpi	Total
Small	1.2%						1.2%
Normal		2.6%	0.2%	36.8%	32.2%	15.8%	87.6%
Large	0.2%	4.1%	2.0%	0.4%	0.5%		7.2%
Xlarge		2.9%		0.5%	0.6%		4.0%
Total	1.4%	9.6%	2.2%	37.7%	33.3%	15.8%	



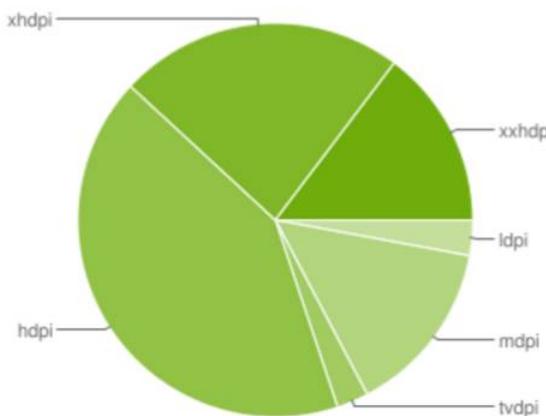
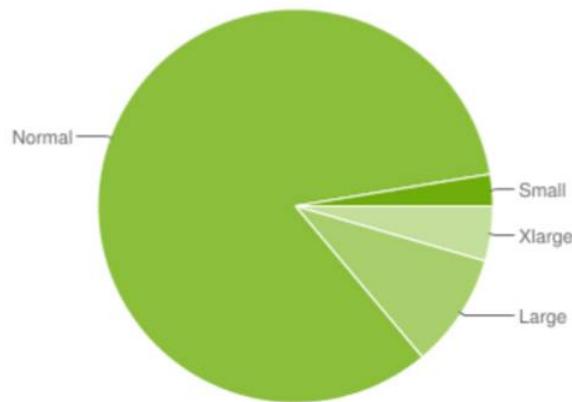
Data collected during a 7-day period ending on January 9, 2017.

- ▶ <http://developer.android.com/about/dashboards/index.html#Screens>
- ▶ http://developer.android.com/guide/practices/screens_support.html#terms



Répartition des types d'écrans (01/2016)

	ldpi	mdpi	tvdpi	hdpi	xhdpi	xxhdpi	Total
Small	2.6%						2.6%
Normal		5.5%	0.1%	41.1%	22.1%	14.7%	83.5%
Large	0.3%	5.4%	2.5%	0.6%	0.6%		9.4%
Xlarge		3.5%		0.3%	0.7%		4.5%
Total	2.9%	14.4%	2.6%	42.0%	23.4%	14.7%	



Data collected during a 7-day period ending on January 4, 2016.

Any screen configurations with less than 0.1% distribution are not shown.



Généralités sur Android



- OS embarqué moderne dédié aux appareils mobiles
- Projet porté par l'Open Handset Alliance, mais piloté par Google
- OS open Source sous licence Apache et distribué gratuitement
- Fonctionne sur les architectures différentes : ARM, Atom, MIPS(TV)
- Exploite le langage java, mais les lib systèmes sont en C
- Une plateforme open Source pour le développement mobile



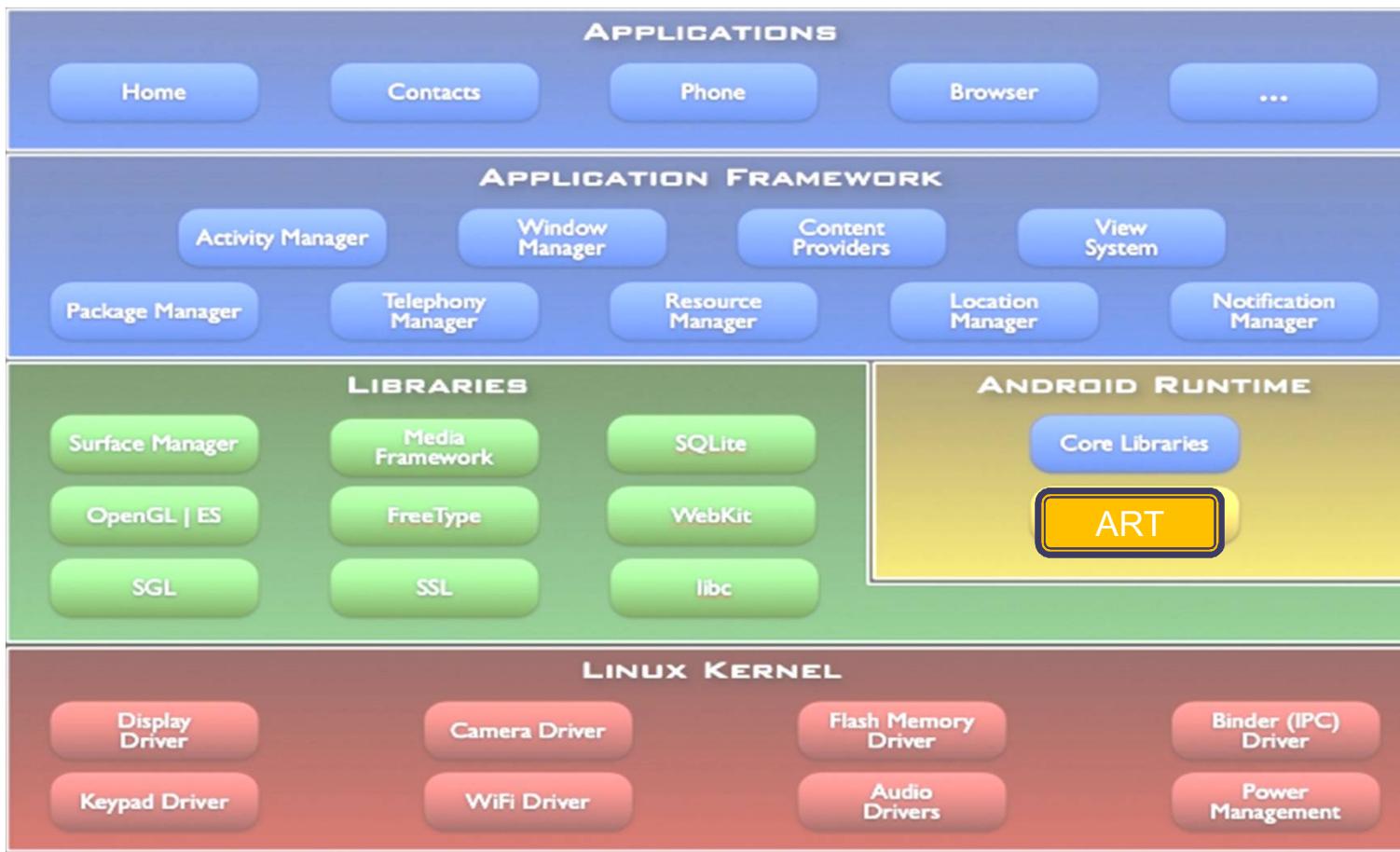
Architecture du système Android

► Android intègre :

- un Framework d'Application
- une machine virtuelle ART (Darvik jusqu'à Android 4.4)
- le Navigateur Intégré (Web Kit)
- une Optimisation Graphique
- le moteur de base de données SQLite
- le support des medias
- la téléphonie GSM (Edge , 3G/4G)
- les Connexions Bluetooth et WiFi
- Un APN, un GPS, un Compas et un accéléromètre
- (un environnement de développement évolué)



Architecture du système Android



ART Runtime (depuis Android 5)

▶ Compilation anticipée

- ART introduit le concept de compilation anticipée (ahead-of-time (AOT), qui permet d'améliorer les performances.
- Lors de l'installation , ART compile l'application en utilisant l'outil embarqué **dex2oat**. Celui-ci accepte en entrée les fichiers DEX et génère une application exécutable spécifiquement pour l'appareil.
- Dalvik était basé sur la compilation lors du lancement (Just In Time JIT compilation).

▶ Garbage collector plus optimisé

- Le Garbage Collector peut influer sur les performance d'une application , la fluidité de l'affichage, etc.

<https://www.infinum.co/the-capsized-eight/articles/art-vs-dalvik-introducing-the-new-android-runtime-in-kitkat>



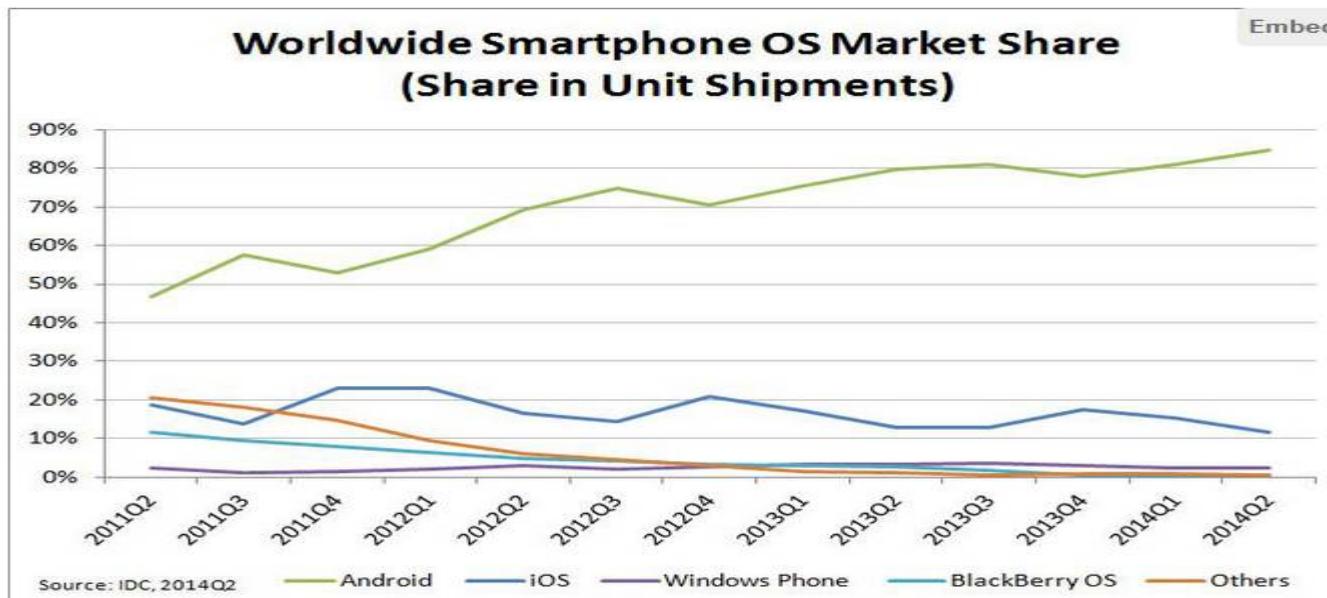
Répartition des OS mobiles (08/2016)

Operating System	2Q16	2Q16	2Q15	2Q15
	Units	Market Share (%)	Units	Market Share (%)
Android	296,912.8	86.2	271,647.0	82.2
iOS	44,395.0	12.9	48,085.5	14.6
Windows	1,971.0	0.6	8,198.2	2.5
Blackberry	400.4	0.1	1,153.2	0.3
Others	680.6	0.2	1,229.0	0.4
Total	344,359.7	100.0	330,312.9	100.0

Source: Gartner (Aout 2016)



Evolution de la répartition des OS mobiles



Source: IDC, 2014 Q2



Parts de marché des fabricants de Smartphones

Company	2Q16 Units	2Q16 Market Share (%)	2Q15 Units	2Q15 Market Share (%)
Samsung	76,743.5	22.3	72,072.5	21.8
Apple	44,395.0	12.9	48,085.5	14.6
Huawei	30,670.7	8.9	26,454.4	8.0
Oppo	18,489.6	5.4	8,073.8	2.4
Xiaomi	15,530.7	4.5	15,464.5	4.7
Others	158,530.3	46.0	160,162.1	48.5
Total	344,359.7	100.0	330,312.9	100.0

Source: Gartner (Aout 2016)



Répartition des revenus iOS / Android



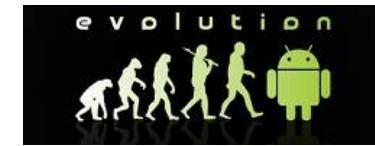
Source App Annie



Evolution des revenus iOS dans le monde



Sommaire



- ▶ 1. Introduction
- ▶ 2. Android de A à Z
- ▶ **3. La diversité des appareils Android**
- ▶ 4. Les outils de développement
- ▶ 5. Architecture d'un projet Android
- ▶ 6. Les fondamentaux d'une application Android
- ▶ 7. Conclusion
- ▶ 8. Bibliographie



Appareils Android



- ▶ Android a été conçu pour les SmartPhones et les tablettes.
- ▶ Son aspect OpenSource a permis de le porter pour d'autres types d'appareils électroniques...



26

Appareils Android (suite)

Android Wear



Google Glass



Chrome Cast & Android TV



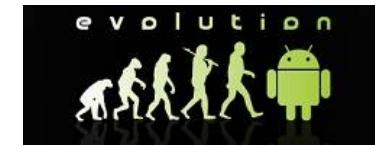
Android Auto



NetBook, eBooks, etc.



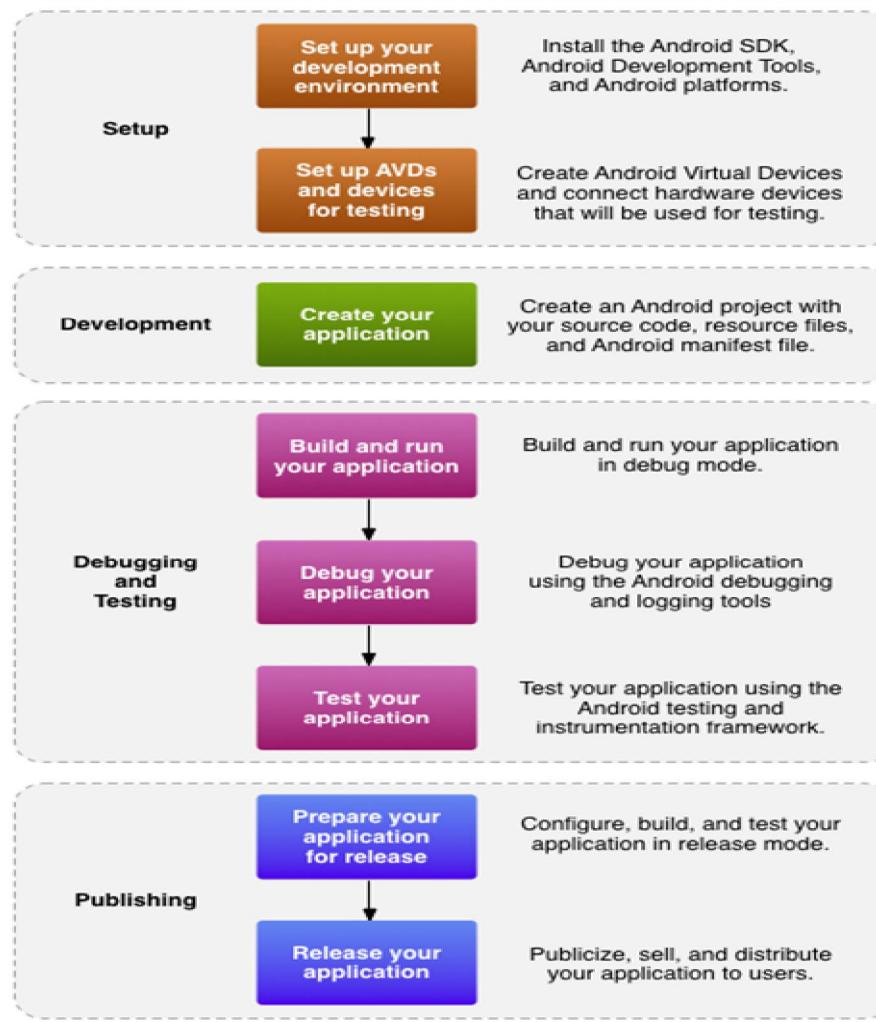
Sommaire



- ▶ 1. Introduction
- ▶ 2. Android de A à Z
- ▶ 3. La diversité des appareils Android
- ▶ **4. Les outils de développement**
- ▶ 5. Architecture d'un projet Android
- ▶ 6. Les fondamentaux d'une application Android
- ▶ 7. Conclusion
- ▶ 8. Bibliographie



Processus de développement pour Android



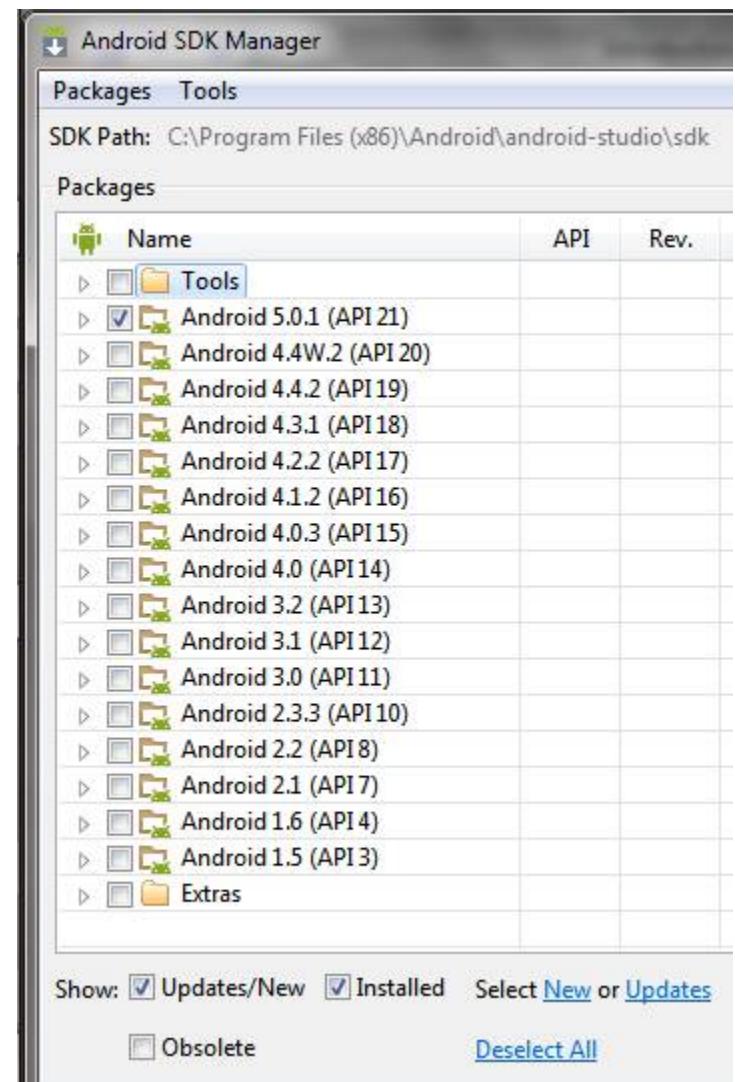
Les outils pour le développement Android

- ▶ Principaux outils utilisés lors du développement sur la plateforme via le SDK d'Android fourni par Google:
 - Android Studio (ou Eclipse avec le plugin ADT)
 - SDK et Device Manager
 - Émulateur Android (AVD)
 - Débogueur



SDK et Device Manager

- ▶ Permet de télécharger les différents versions du système d'exploitation et des APIs de google
- ▶ Permet également de créer, modifier et démarrer les appareils virtuels



Android Studio

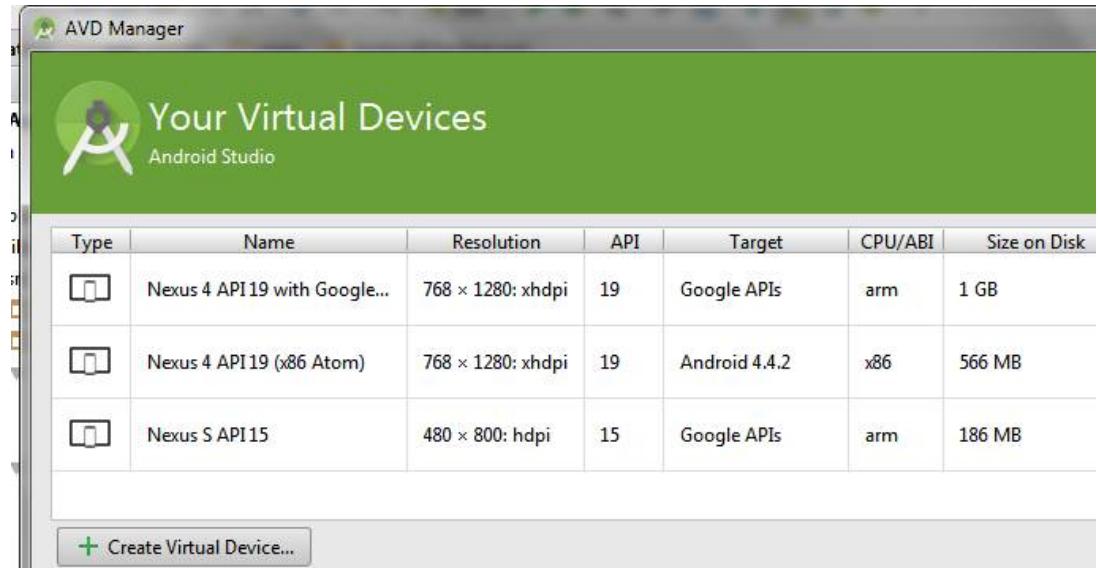
- ▶ IDE officiel pour le développement d'applications Android
(Le plugin ADT d'Eclipse n'est plus maintenu)
- ▶ Based on [IntelliJ IDEA](#)
- ▶ Flexible Gradle-based build system
- ▶ Build variants and multiple apk file generation
- ▶ Code templates to help you build common app features
- ▶ Rich layout editor with support for drag and drop theme editing
- ▶ Lint tools to catch performance, usability, version compatibility, and other problems
- ▶ ProGuard and app-signing capabilities
- ▶ Built-in support for [Google Cloud Platform](#), making it easy to integrate Google Cloud Messaging and App Engine

<http://developer.android.com/tools/studio/index.html>



AVD Manager

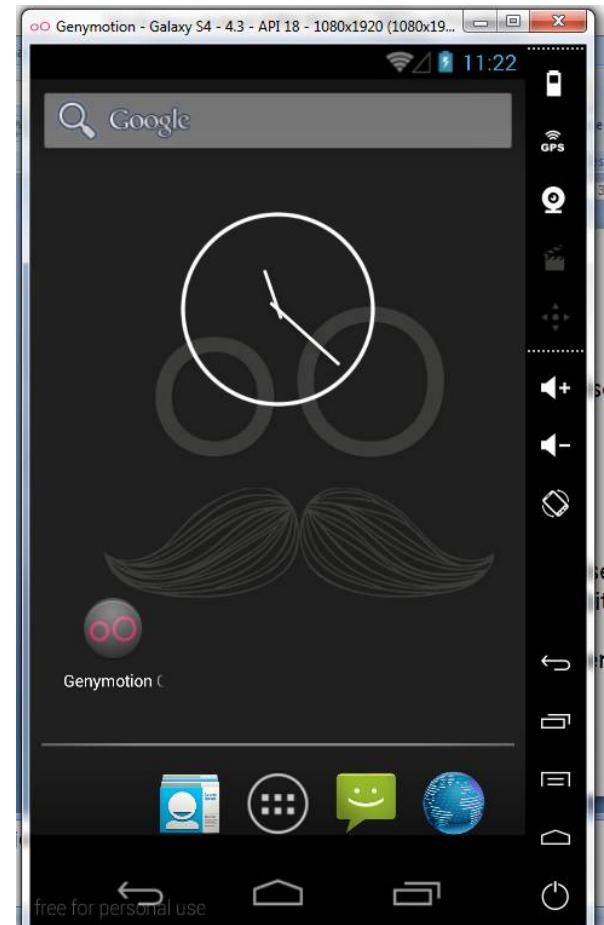
- ▶ Permet de sélectionner les attributs principaux de l'appareil :
 - Identifiant unique
 - Version du système d'exploitation
 - Taille de la carte SD
 - Skin ou résolution d'écran
 - Paramètres de configuration technologique (support caméra, quantité de mémoire, présence du GPS, etc.)



Émulateur Android

- ▶ Quelques raccourcis importants :
 - Faire pivoter l'écran : CTRL-F12
 - Permuter l'activation de la simulation du réseau cellulaire : F8
 - Permuter le profilage de code : F9

- ▶ Quelques paramètres de démarrage intéressants :
 - -netdelay <delay> : simuler de la latence réseau
 - -netspeed <speed> : simuler un certain débit réseau
 - -cpudelay <delay> : simuler un ralentissement du processeur



Débogueur

- ▶ DDMS : Dalvik (l'ancienne JVM Android) Debug Monitor Server
- ▶ Sur Android chaque application tourne comme une nouvelle instance de la machine virtuelle Dalvik / ou ART, avec son propre port pour le débogueur
- ▶ Pour lancer le debugger : 
- ▶ Attacher le debugger au process à debugger : 

<https://developer.android.com/studio/debug/index.html>



Système de Log

- ▶ les messages systèmes sont affichés dans le LogCat lors du débogage de l'application.
- ▶ la classe Log permet d'écrire des message de log dans votre code.
- ▶ Exemple :
 - `Log.d(TAG, "passage dans onCreate()");`

<https://developer.android.com/tools/debugging/debugging-studio.html#systemLog>



Bien gérer la multitude des appareils

- ▶ Suivre l'évolution des versions d'OS

<http://developer.android.com/resources/dashboard/platform-versions.html>

- ▶ Suivre l'évolution des tailles et résolutions d'écrans

<http://developer.android.com/about/dashboards/index.html#Screens>



Bien gérer la multitude des appareils

- ▶ Tester sur les appareils eux-mêmes dans de vraies situations :
 - Afin de tester les usages véritables de l'appareil, rotations aléatoires, résolutions d'écrans différents, etc.
 - Et bien sûr, l'expérience avec l'écran tactile et les boutons de l'appareil
- ▶ Vérifier la compatibilité avec les différentes versions d'API
- ▶ Vérifier la compatibilité avec les différentes tailles et résolutions d'écrans



Le processus de publication

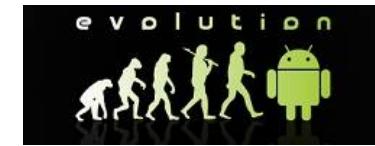
- ▶ Exportation de l'application avec l'environnement de développement dans un fichier .apk
- ▶ Signature de l'application avec keytool et jarsigner
- ▶ S'inscrire pour publier l'application sur le Google PLay (25\$ à vie)
- ▶ Soumettre l'application en la décrivant correctement

▶ Processus d'approbation ?

- ▶ On parle plutôt d'une gestion à postériori !
- ▶ Les frais de transaction prélevés sur Google PLay sont équivalents à 30 % du prix d'achat de l'application.



Sommaire



- ▶ 1. Introduction
- ▶ 2. Android de A à Z
- ▶ 3. La diversité des appareils Android
- ▶ 4. Les outils de développement
- ▶ **5. Structure d'un projet Android**
- ▶ 6. Les fondamentaux d'une application Android
- ▶ 7. Conclusion
- ▶ 8. Bibliographie



Structure d'un projet (Android Studio)

- ▶ Android studio permet de créer des projets multiplateforme (Mobile / TV / Glass / Wear)
- ▶ Chaque plateforme peut définir plusieurs dossiers sources correspondant à différentes versions d'exécutables (ex: gratuite / payante) -> flavors
- ▶ Android studio s'appuie sur le système de compilation 'Gradle' qui permet de gérer cette multiplicité avec des fichiers de scripts (build.gradle) hiérarchiques.

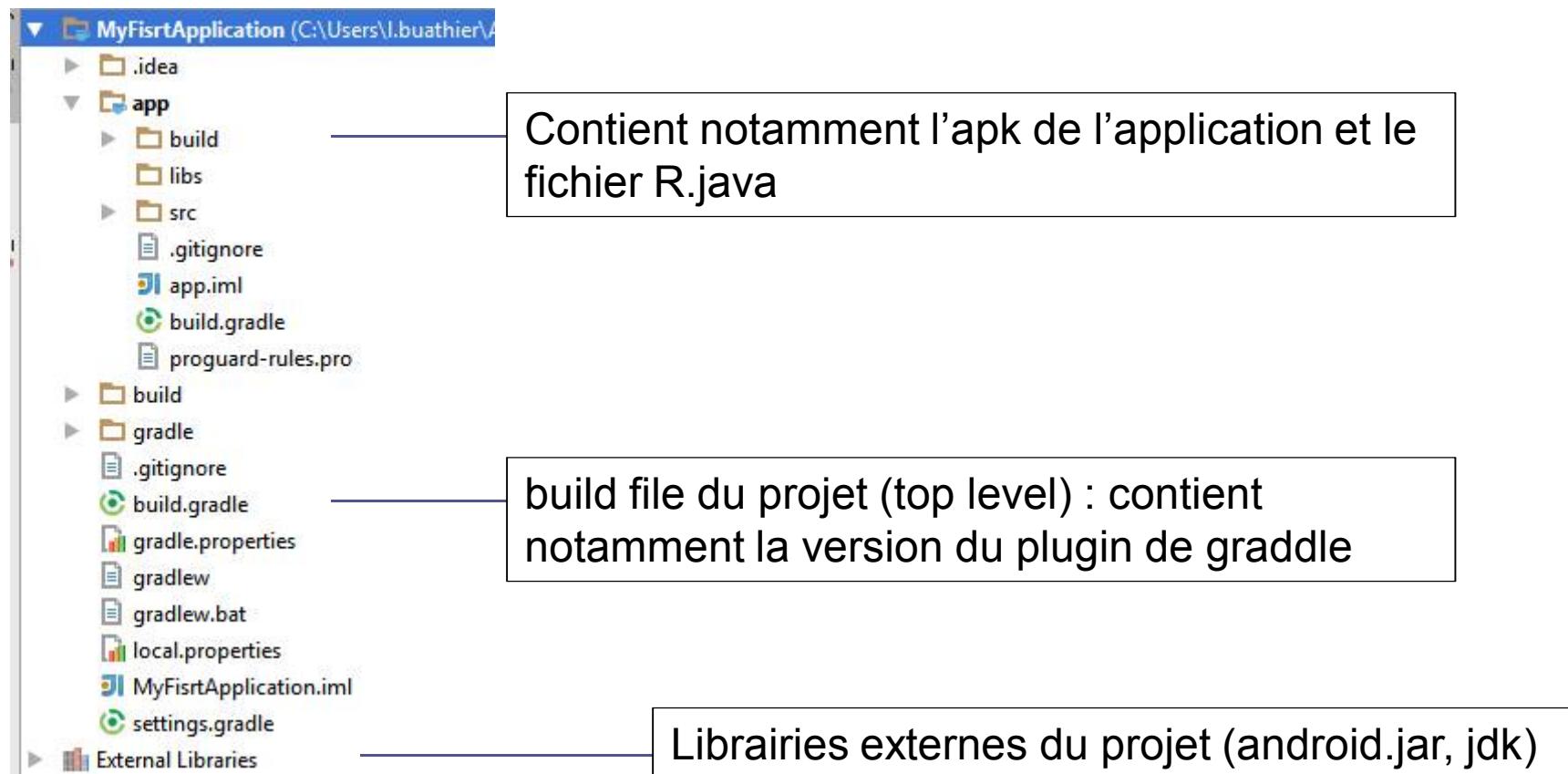
<http://developer.android.com/sdk/installing/studio-build.html>

<http://developer.android.com/tools/building/configuring-gradle.html>

<http://developer.android.com/tools/building/plugin-for-gradle.html>



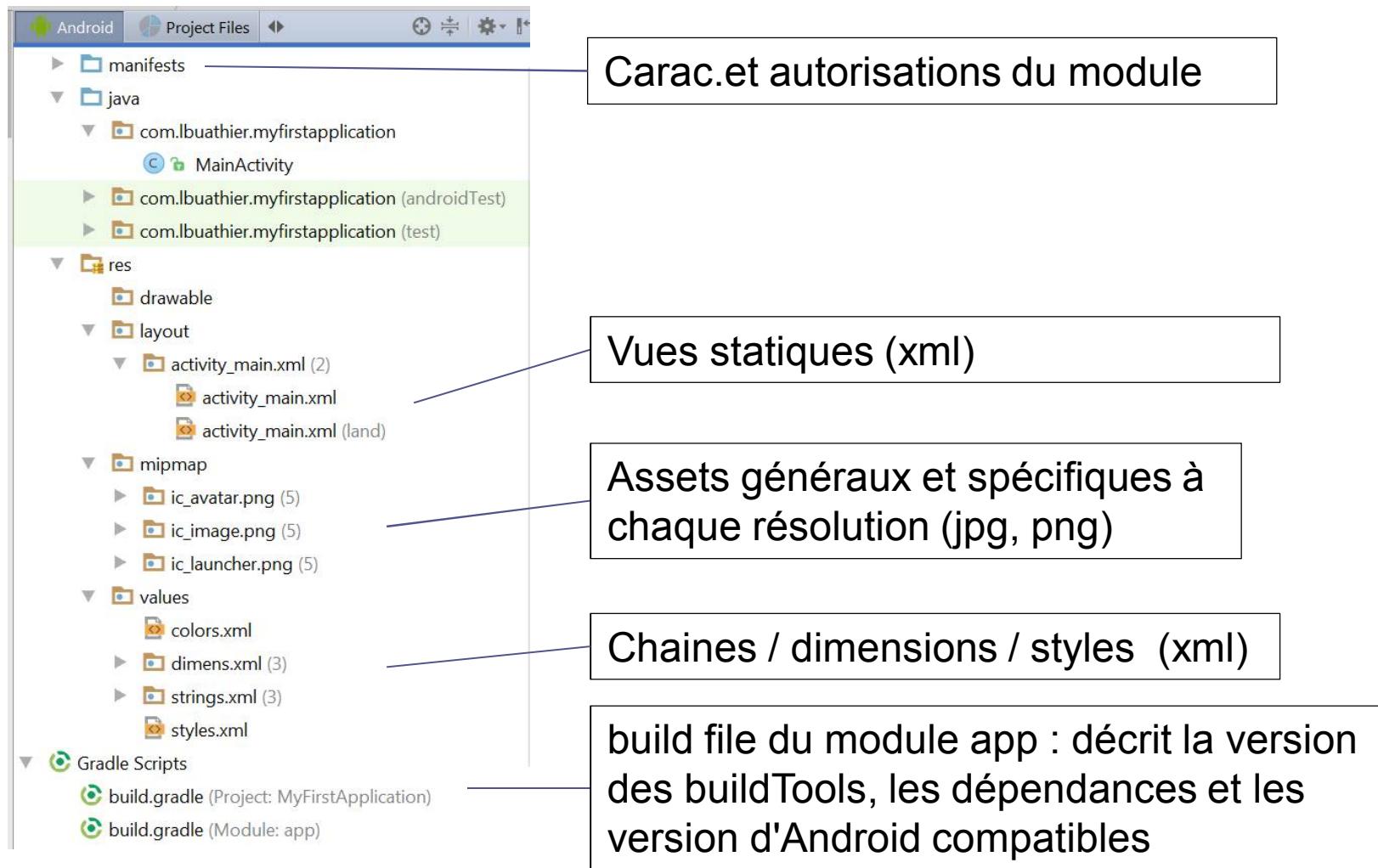
Structure d'un projet (Android Studio)



<https://developer.android.com/tools/studio/index.html>

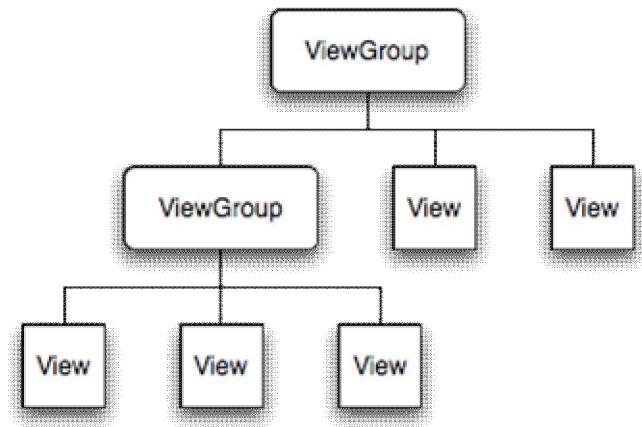


Structure d'un projet (Android Studio)



Layouts

- ▶ Les layouts définissent la structure des différents écrans
- ▶ Ils sont composés sous la forme de fichiers XML définissant le positionnement ou le séquencement des différents éléments
- ▶ Ils sont définis sous la forme d'un arbre de vues et de groupes de vues permettant le maximum de réutilisation
- ▶ Les layout peuvent être redéfinis par exemples pour le mode landscape



Layouts – un exemple

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <Button
        android:id="@+id/buttonValidation"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="@string/buttonValid" />
</LinearLayout>
```



Resources

- ▶ Les ressources représentent les éléments textuels, graphiques et de styles nécessaires à l'affichage et au fonctionnement d'une application :
 - ▶ Animation
 - ▶ Color
 - ▶ Drawable
 - ▶ Layout
 - ▶ Menu
 - ▶ String
 - ▶ Style
 - ▶ Values



Resources – i18n et résolutions multiples

- ▶ Android supporte un ensemble de convention de nommage permettant de gérer les requis d'internationalisation et d'adaptation aux différents écrans des appareils
- ▶ Les conventions de nommage utilisent les codes iso (i18n) pour les différents pays
- ▶ Ainsi il s'agit de définir par exemple les répertoires suivants pour différencier les images en anglais et français :
 - ▶ drawable
 - ▶ drawable-fr
- ▶ Même approche pour les différents formats d'écrans
 - ▶ drawable-ldpi
 - ▶ drawable-mdpi
 - ▶ drawable-hdpi



AndroidManifest.xml

- ▶ Définit l'architecture d'un module de l'application ainsi que les différents éléments de configuration nécessaires à son exécution :
 - ▶ Titre de l'application
 - ▶ Icône de l'application
 - ▶ La liste des activités de l'application
 - ▶ L'activité principale lancée au démarrage
 - ▶ La liste des **permissions** nécessaires à l'application pour son exécution



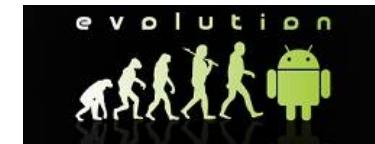
AndroidManifest.xml – un exemple

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.iut.example">
    <uses-permission android:name="android.permission.INTERNET"/>
    <application android:icon="@drawable/icon"
        android:label=" Example" android:theme="@style/Theme.Holo">
        <activity android:name=".ExampleActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
        <activity android:name=".InformationListActivity"/>
        <activity android:name=".InformationActivity"/>
    </application>
    ...

```



Sommaire



- ▶ 1. Introduction
- ▶ 2. Android de A à Z
- ▶ 3. La diversité des appareils Android
- ▶ 4. Les outils de développement
- ▶ 5. Architecture d'un projet Android
- ▶ **6. Les fondamentaux d'une application Android**
- ▶ 7. Conclusion
- ▶ 8. Bibliographie



Architecture d'une application Android

- ▶ Une application est constituée :
 - **d'activités** (ou de fragments) qui gèrent les vues
 - **d'intentions** (intent) à destination de composants internes ou externes
 - de **fournisseurs de contenus** (content provider) pour le partage d'information interne ou externe à l'application
 - de **services** (qui tournent en tache de fond)
 - de **récepteurs de services et d'intentions** (broadcast receiver)

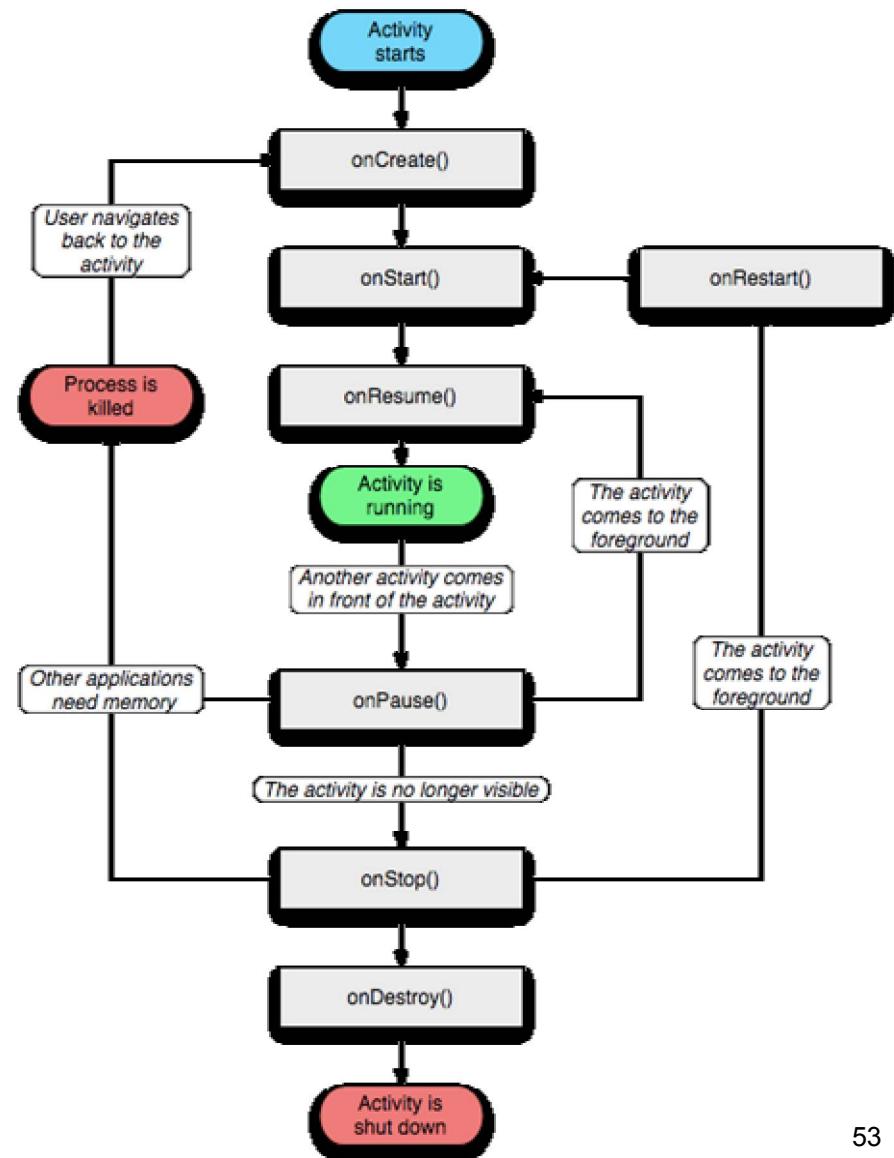


Activity

- ▶ Les activités sont au cœur même des applications Android
- ▶ Il s'agit de la représentation d'une interface utilisateur ayant un but précis pour l'utilisateur
- ▶ On peut le comparer à la partie contrôleur d'une application MVC
- ▶ Le contenu de l'écran supportant une activité est défini par les vues tandis que le traitement sous-jacent est supporté par le modèle, à la manière du MVC



Activity – Le cycle de vie



Activity - Exemple

```
public class Exemple extends Activity {  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.example_main);  
  
        Button myButton = (Button) findViewById(R.id.Button);  
        myButton.setOnClickListener(new Button.OnClickListener()  
        () {  
            public void onClick(View arg0) {  
                Log.i("Exemple", "clic sur bouton1");  
            }  
        });  
    }  
}
```



Services

- ▶ Les services sont des tâches qui tournent en arrière plan sans écran pour les supporter. Exemples :
 - ▶ jouer de la musique
 - ▶ Effectuer des traitements de données ⇒ affichage lorsque disponibles
- ▶ Le service expose une interface
 - ▶ commandes pour la mise en pause, l'arrêt ou la reprise de la lecture pour le lecteur de musique



Intent

- ▶ Permet de démarrer une nouvelle instance d'une activité ou d'un service et de lui passer les paramètres nécessaires à son exécution
- ▶ Il s'agit donc d'un objet que l'on passe à la méthode de démarrage d'une activité

```
Intent intent = new Intent(getApplicationContext(),  
                           InformationActivity.class);  
  
intent.putExtra("info", info);  
startActivity(intent);
```



Views

- ▶ Une View représente un composant graphique réutilisable à l'intérieur d'un ou plusieurs écrans d'une application
- ▶ Les vues sont par la suite agrégées afin de créer des écrans via le fichier de positionnement (« layout »)
- ▶ Un ensemble complet de vues est fourni par l'API Android
 - ▶ Button, Checkbox, EditText, DatePicker, etc. pour les formulaires
 - ▶ ImageView
 - ▶ ListView, ExpandableListView
 - ▶ ProgressBar, SeekBar, MediaController
 - ▶ VideoView
 - ▶ WebView
 - ▶ etc.



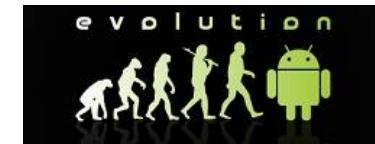
Adapters

- ▶ Les Adapters servent à alimenter les vues en données
- ▶ Certaines vues fournies avec l'API Android requièrent l'utilisation d'un adapter, comme par ex. une ListView.

```
public class InformationAdapter extends BaseCursorAdapter {  
  
    @Override  
    public void bindView(View view, Context context, Cursor cursor)  
    {  
        TextView tt = (TextView) view.findViewById(R.id.name);  
        if (tt != null) {  
            tt.setText(cursor.getString(cursor.getColumnIndex("name")));  
        }  
    }  
}
```



Sommaire



- ▶ 1. Introduction
- ▶ 2. Android de A à Z
- ▶ 3. La diversité des appareils Android
- ▶ 4. Les outils de développement
- ▶ 5. Architecture d'un projet Android
- ▶ 6. Les fondamentaux d'une application Android
- ▶ **7. Conclusion**
- ▶ 8. Bibliographie

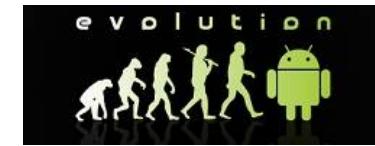


Conclusion

- ▶ Le développement sur plateforme mobile représente un enjeu de plus en plus important pour les sociétés informatiques.
- ▶ Les plateformes et outils de développements sont en perpétuelle évolution et nécessitent une remise en question permanente.
- ▶ La particularité d'Android est la difficulté de gérer les différentes versions d'API et les différents écrans...
- ▶ Vous allez apprendre à maîtriser tous ces aspects en TP.



Sommaire



- ▶ 1. Introduction
- ▶ 2. Android de A à Z
- ▶ 3. La diversité des appareils Android
- ▶ 4. Les outils de développement
- ▶ 5. Architecture d'un projet Android
- ▶ 6. Les fondamentaux d'une application Android
- ▶ 7. Conclusion
- ▶ **8. Bibliographie**



Bibliographie

- ▶ Cours Android CNAM - Guillaume GENS
- ▶ Cours Android - Steve Tremblay - Nurun Montréal
- ▶ Développez.com
- ▶ L'art du développement Android – Mark Murphy
- ▶ Android 4 – Reto Meier
- ▶ Android - Apprenez à développer efficacement pour le leader des OS mobiles – Florent Garin
- ▶ Et de nombreux autres livres de la bibliothèque
 - Allez les consulter !

