heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Les plateformes mobiles

Fabien Dutoit

SYM – Systèmes mobiles

Utilisation

- *Des utilisations assez distinctes :*
  - *Systèmes embarqués dédiés*
  - *Smartphones (iOS, Android)*
  - *Internet des Objets*
- *La téléphonie est en stagnation, alors que les smartphones et plus particulièrement les applications mobiles sont en croissance*
- *Dans le cadre de ce cours, nous allons principalement nous intéresser aux smartphones*

Contexte

- *Contexte très volatil, plateformes très hétérogènes (HW et SW)*
- *Difficulté de concevoir des applications multi-plateformes*
- *Une spécification globale matérielle fait encore défaut*
- *Le smartphone pourrait devenir le périphérique d'authentification personnelle universel d'ici quelques années*

Les manques

- *Difficulté de stocker une identité de manière sûre dans le smartphone (UICC…)*
- *Pas de zone de stockage sécurisée (tamper proof)*
- *Pas de cartes SD authentifiées / marquées*
- *Pas de circuit dédié à l'authentification*

Les manques - un exemple

Difficulté de stocker une identité de manière sûre dans le smartphone:

- *Sunrise ID Checker*
  (annoncé le 14.09.2017 – 36 vues sur la vidéo de présentation le 31.08.2019)
- *Utilise les potentialités des smartphones pour signer un contrat sur son écran, les étapes:*

1. Scanner un code QR
2. Prendre une photo de la carte d'identité (r/v)
3. Se filmer le visage quelques secondes
4. Signer avec le doigt sur l'écran
5. Valider le contrat
6. Validation par un opérateur humain

src: https://www.xavierstuder.com/2017/09/15/sunrise-id-checker-contrat-numerique/

Problèmes communs

- *Petites dimensions, ressources restreintes, consommation réduite*
- *Fonctions de téléphonie sévèrement temps réel*
- *Système embarqué pouvant convoyer des données sensibles*
- *Manipulation à une seule main*

Systèmes d'exploitation mobiles

Actifs

Android - Google

iOS - Apple

Développement stoppé en 2017 - maintenance limitée

Windows 10

BlackBerry

Passage à Android

Abandonnés

Windows Phone

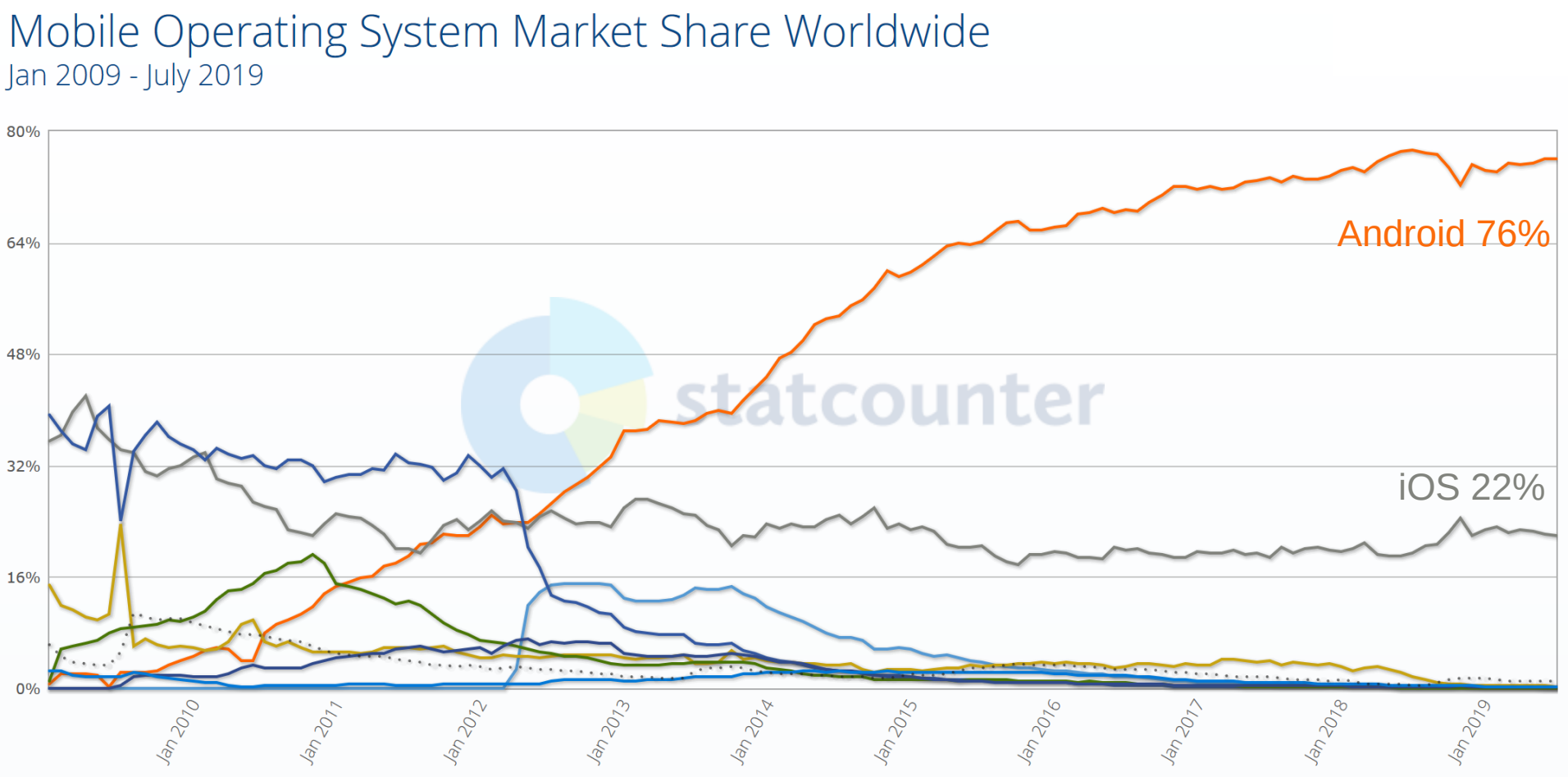
Firefox OS

webOS

symbian

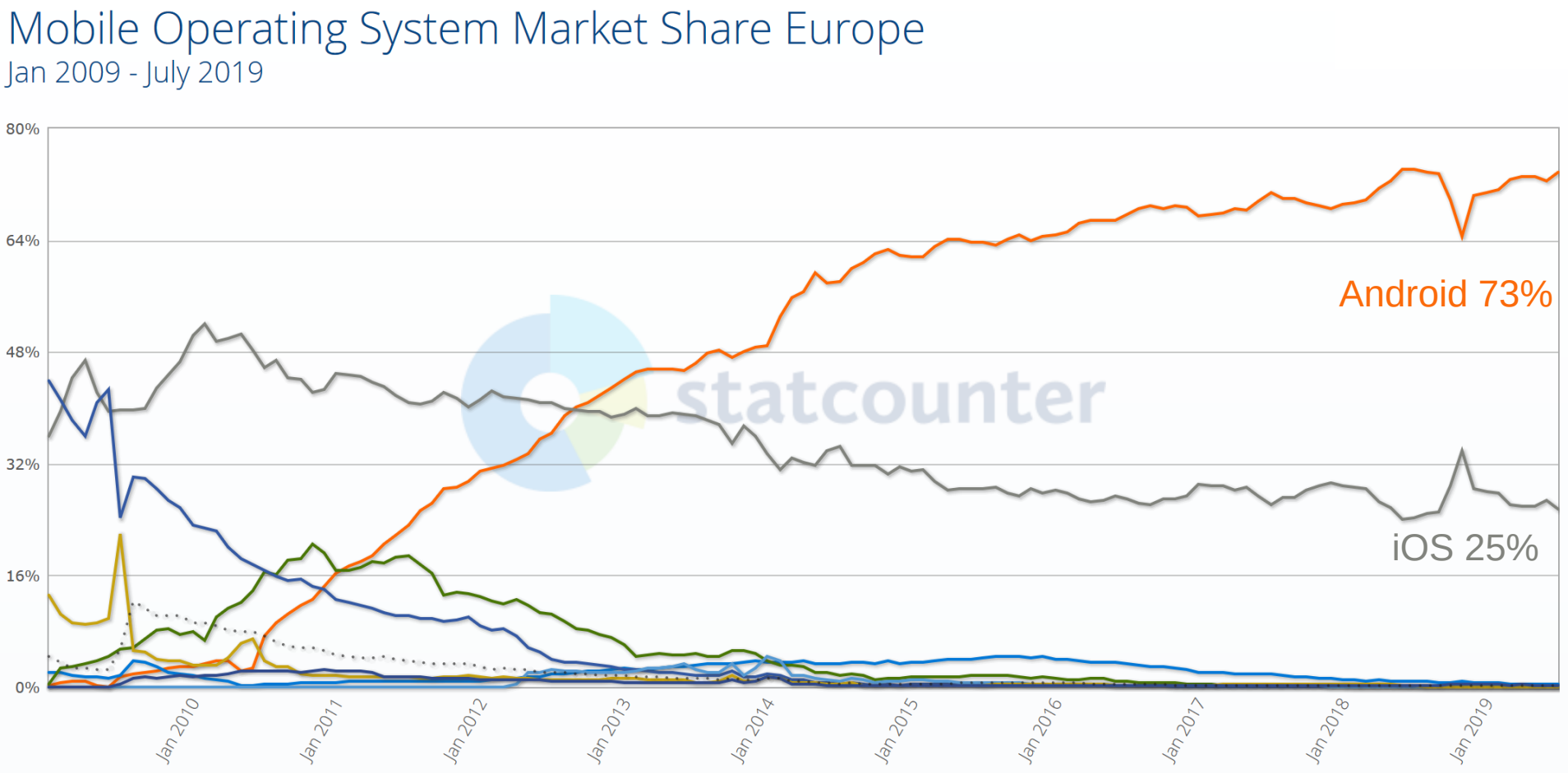
...

Statistiques - Monde



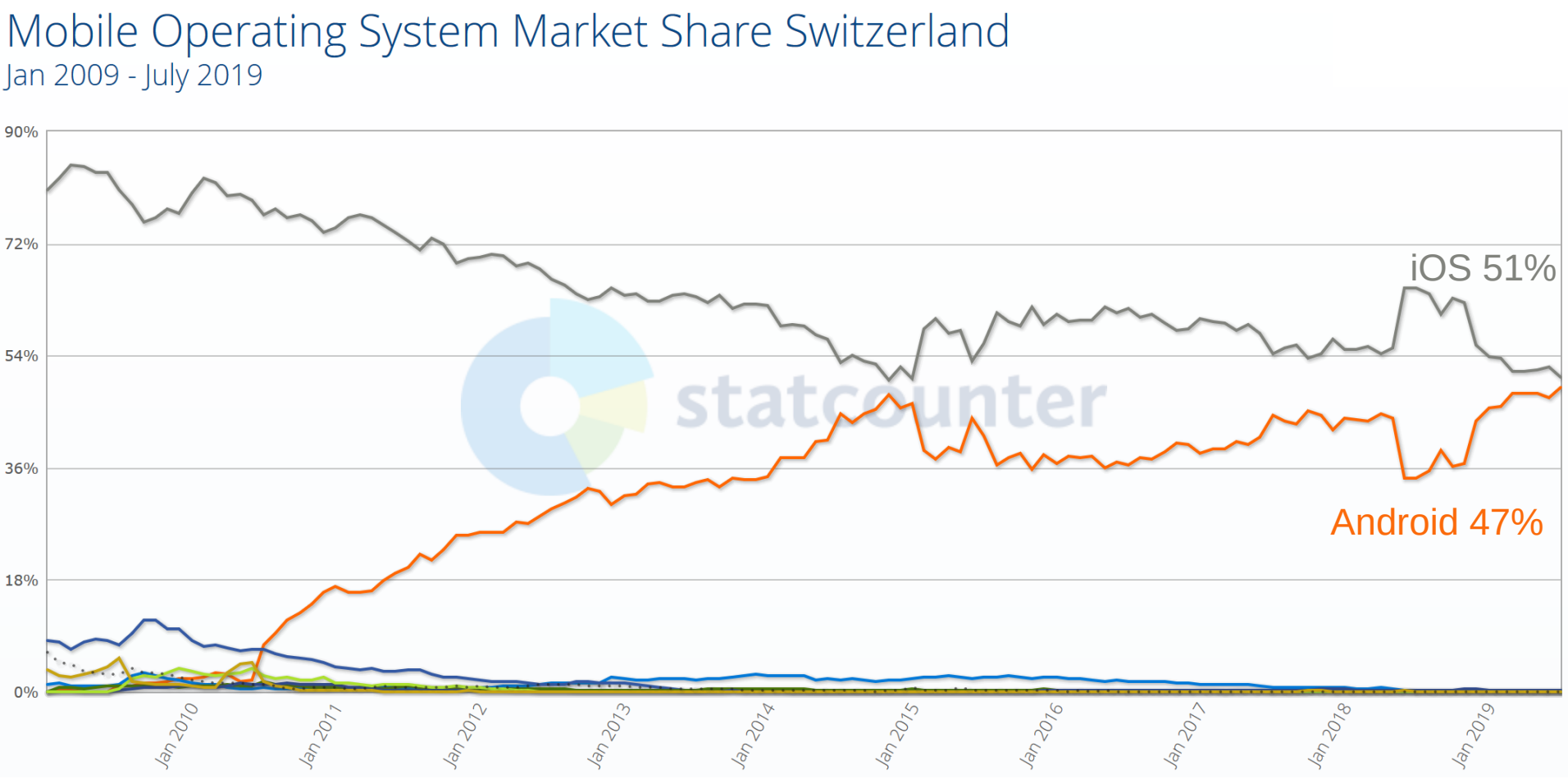
src: http://gs.statcounter.com/os-market-share/mobile

Statistiques - Europe



src: http://gs.statcounter.com/os-market-share/mobile

Statistiques - Suisse



src: http://gs.statcounter.com/os-market-share/mobile

Palm OS

- *Palm était active dans l'industrie des ordinateurs de poche (PDA), il a été racheté en 2010 par HP*
- *Venu tard à la téléphonie, Palm est un acteur mineur*
- *Programmable en C, C++ ou Java*
- *La base Linux en fait une base de développement assez favorable, mais consommant beaucoup*
- *Avenir incertain, mise en Open Source en novembre 2011*
- *Remplacé par webOS, sur lequel quelques téléphones et tablettes sortent jusqu'en 2011. Utilisé depuis par LG sur ses télévisions connectées*

J2ME

- *Plate-forme logicielle de Sun Microsystems, destinée à la programmation des systèmes embarqués*
- *Deux modèles principalement : CDC et CLDC*
- *Très peu d'implémentations CDC, CLDC est considéré comme très insuffisant*
- *Peu de possibilités de programmation à bas niveau, sinon par les JSR*

Symbian OS

- *Partenariat entre PSION, Nokia, Ericsson, Motorola et Matsushita, en 1996*
- *Nokia devient le seul propriétaire en 2008*
- *Sévère érosion des ventes depuis 2008*
- *Symbian OS vient de Epoc 32 (Psion) et est Open Source depuis 2010, Nokia continuera le développement d'une branche fermée*
- *Autonomies (batterie) record*
- *Avenir plombé depuis l'accord Microsoft-Nokia*

Nokia Symbian

Historiquement, le premier véritable "smartphone"
(Nokia N 95 – avril 2007)

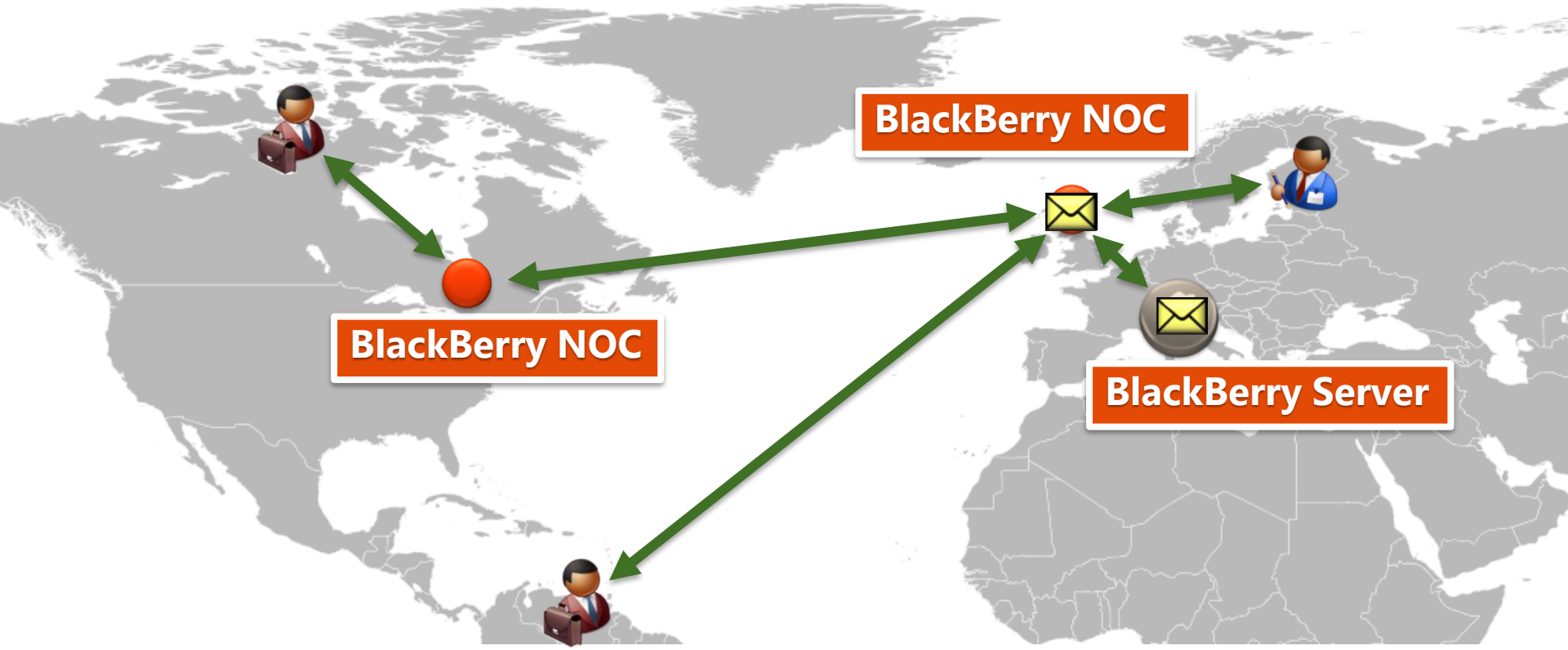
Nokia: what's next ?

- *L'accord d'exclusivité avec Microsoft s'est terminé fin 2015*
- *Nokia peut produire dès 2016 de nouveaux smartphones, pas forcément estampillés Microsoft*
- *Une douzaine de smartphones Android sont sortis depuis début 2017*
- *Une nouvelle version du 3310 est sortie en 2017*

RIM Blackberry

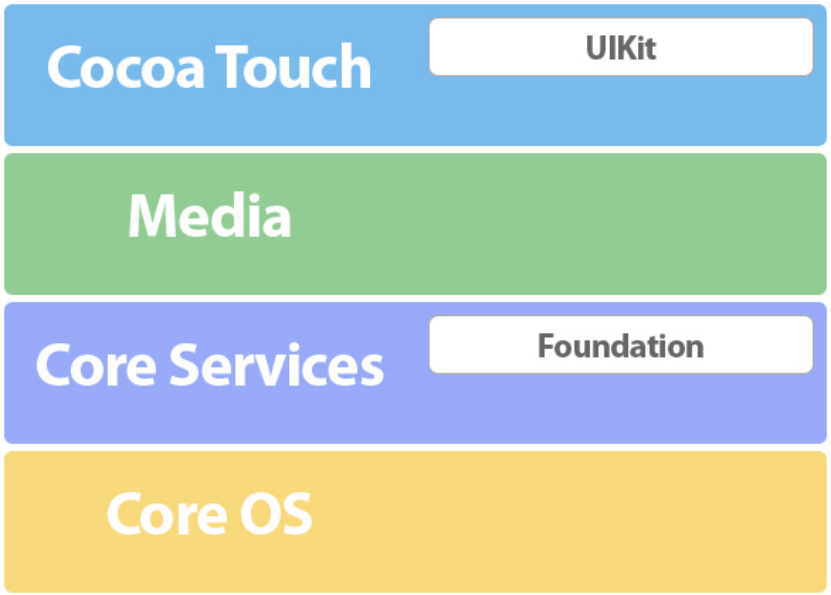
- *Fondé en 1999, premiers succès en 2003*
- *Protocole propriétaire pour la technologie push e-mail*
- *Chiffrement des données*
- *Bonne pénétration du marché business, en particulier aux Etats-Unis*
- *Programmation des applications en Java*
- *En difficultés depuis l'automne 2011 (panne générale du réseau)*
- *En mars 2017, sortie du BlackBerry Aurora tournant sur Android 7*
- *En 2018, sortie du KEY2, smartphone avec un clavier physique*

RIM, le problème du NOC



iOS - Apple

- *Première version en 2007*
- *Système d'exploitation mobile dérivé de macOS*
  - *Basé sur le noyau hybride XNU (Unix-like)*
  - *Portage de NextStep (d'où le langage Objective-C)*
- *Partage une architecture similaire à macOS:*



src: https://aruniphoneapplication.blogspot.com

iOS

- *Le développement pour iOS repose sur l'environnement XCode (IDE + LLVM toolchain + simulateur)*
- *Il existe des alternatives, par ex: AppCode ou Visual Studio*
  Mais la compilation sera obligatoirement réalisée par XCode et donc possible uniquement sur un mac
- *Nécessite une licence payante pour développer et distribuer*
  - *Une version gratuite existe, déploiements limités à 7 jours et ne permet pas d'utiliser la plupart des fonctionnalités iOS avancées (cloud, messagerie push, etc.)*

iOS

- *Objective-C à l'origine développé comme préprocesseur de C pour émuler Smalltalk*
- *Java s'est beaucoup inspiré de l'Objective-C*
- *« Message passing » à la Smalltalk*
- *Depuis 2014, utilisation du langage Swift*

iPhone - business model

- *Modèle vertical fidèle à la philosophie Apple: tout gérer de A à Z*
- *Plateforme matérielle de bonne facture, mais hyper-protégée et fermée*
- *Ceci a imposé un environnement assez fermé pour le développement « third party »*
- *En conséquence, la plate-forme est aussi moins évolutive de manière à en conserver le caractère exclusif*

iPhone - business model

- *En août 2018, Apple disposait d'une capitalisation record de plus de 1'000 milliards de dollars (!)*
  - *En billets de 1000$, cela ne fait jamais qu'une pile d'une hauteur de 106 kilomètres…*
  - *-33% début 2019*
- *La combinaison d'une offre très réduite (plateforme unique) avec un système très fermé permet de maximiser les bénéfices et assurer une marge importante d'env. 40-50%*

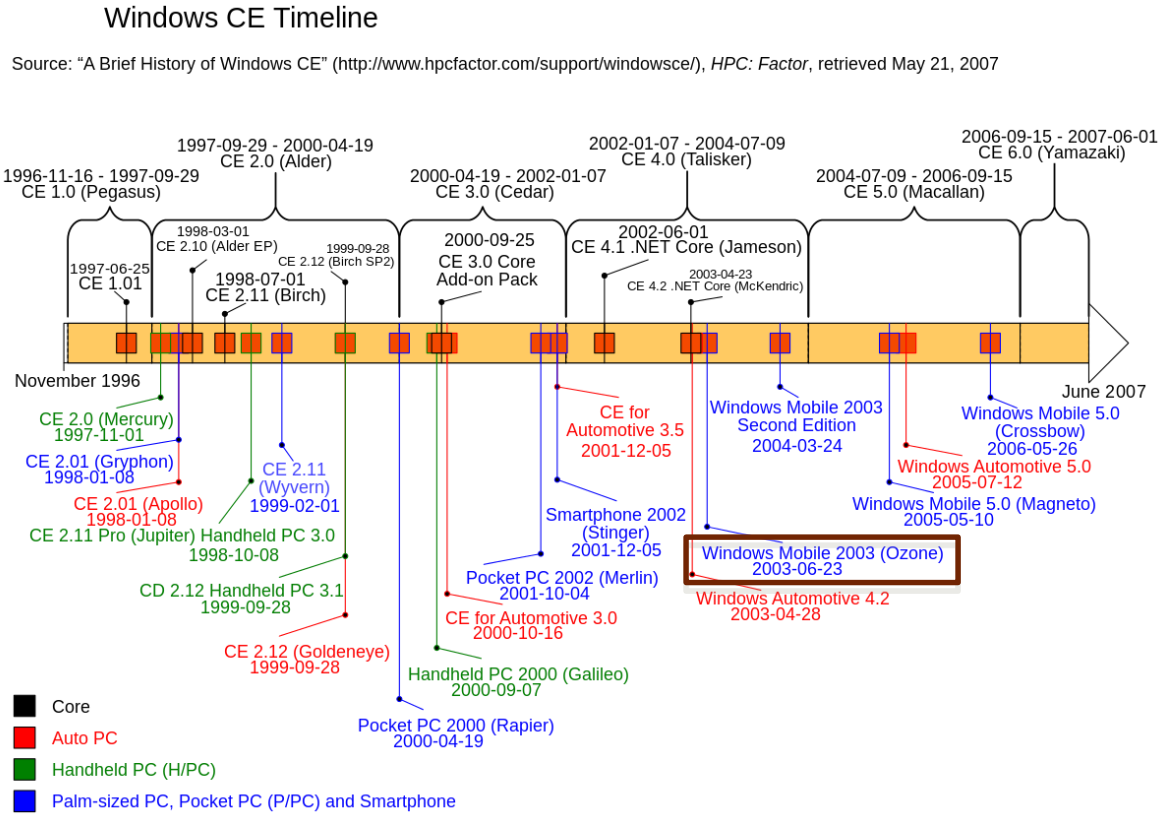
iPad, tablettes

- *Apple est le leader incontesté sur le marché des tablettes de grande dimension*
- *Depuis l'iPhone 6+, Apple occupe aussi le segment des «phablette», laissé jusqu'alors à la concurrence*
- *Les derniers iPhone ont déçus à leur sortie: pas de NFC ou alors du NFC sans API disponible, incompatibilités matérielles (chargeurs, carte SIM, prise jack, connecteur Lightning…), options logicielles discutables…*
- *Utilisation de iPad OS depuis 2019*

Windows

Trois séries des plateformes différentes:

- *Windows CE / Windows Mobile (entre 1996 et 2013)*
  Destiné initialement aux PC de poche (PDA et PocketPC)



src: wikipedia.org

Windows

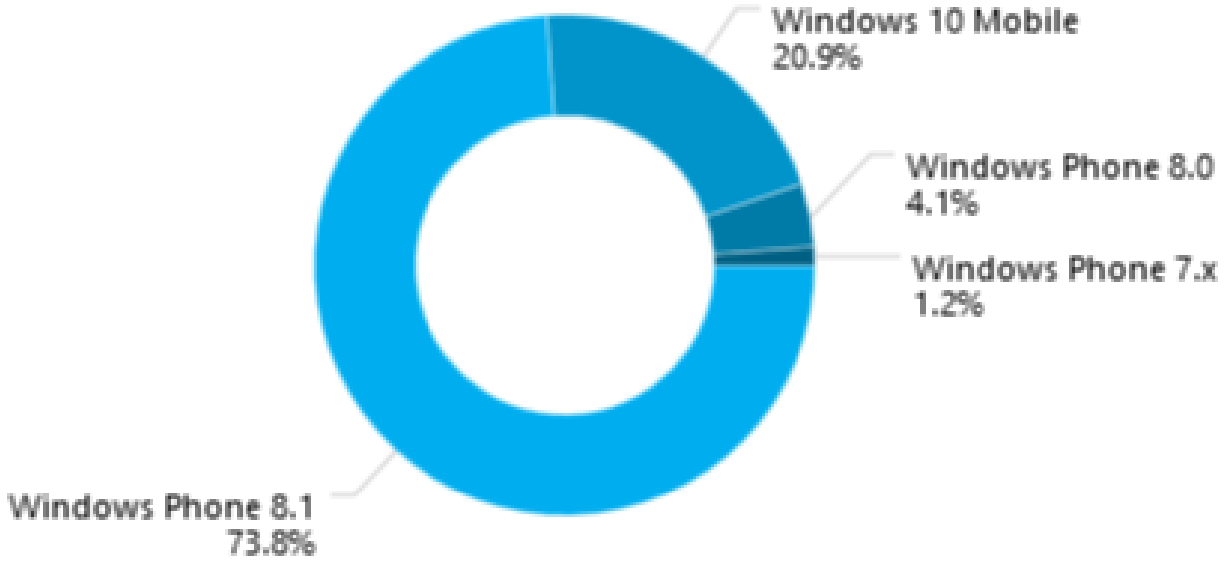
- *Windows Phone (entre 2010 et 2015)*
  - *Successeur de Windows Mobile et Microsoft Zune*
  - *Destiné à être en compétition avec iOS et Android*
  - *Plusieurs versions: Windows Phone 7, 8 et 8.1*
  - *Version utilisée lors du partenariat avec Nokia (en remplacement de Symbian)*
  - *Fin du support de l'OS en juillet 2017*
  - *Fermeture définitive du store planifiée pour 2019*

Windows

- *Windows 10 Mobile (depuis 2016)*
  - *Successeur de Windows Phone*
  - *Unification avec la version desktop*
  - *Depuis fin 2017, Microsoft ne développe plus activement cette plateforme. Uniquement des correctifs de bugs et de sécurité jusqu'à fin 2019*

Windows 10 Mobile

- *Microsoft a mis fin au support de Windows Phone 8 et 8.1 début juillet 2017*
- *Il reste uniquement Windows 10 Mobile, représentant uniquement 20% du parc de mobiles sous Windows*



src: http://blog.adduplex.com, Octobre 2017

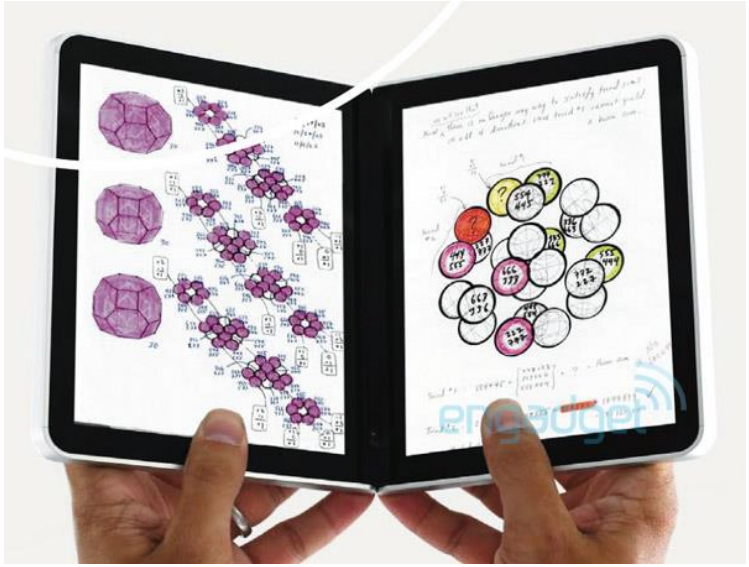
Windows 10 Mobile

- *Plateforme purement logicielle, mais relativement fermée*
- *Langage de programmation préféré C# : un Java avec des pointeurs… Basé sur .NET et XAMARIN*
- *Alternatives : C++, JavaScript*
- *En sérieuse perte de vitesse, Microsoft a stoppé le développement actif*

Windows - Microsoft Courier

Microsoft, pionnier des tablettes graphiques (Tablet PC sous XP) et des organiseurs tactiles, n'a pas cru dans le potentiel de ces "jouets" (ainsi qualifiés par l'équipe de développement d'Office)

Les stratèges de Redmont ont abandonné le développement peu avant qu'Apple ne sorte l'iPhone...



https://www.cnet.com/news/the-inside-story-of-how-microsoft-killed-its-courier-tablet/

Firefox OS

- *Système d'exploitation mobile libre destiné aux objets connectés développé par Mozilla et basé sur un noyau Linux*
- *Premier smartphone en 2013 (ZTE Open)*
- *Applications développées en HTML/CSS/JavaScript*
- *0.1% des smartphones vendus en Europe en 2014*
- *Des télévisions connectées tournent sur cet os par ex. Panasonic*
- *Dernière version stable en novembre 2015…*

Librem

- *Librem est une gamme d'ordinateurs portables utilisant du hardware et software 100% open-source*
- *Un smartphone est prévu, le Librem 5 basé sur PureOS (Debian)*
- *KDE adaptera l'interface graphique Plasma pour mobile*
- *Campagne de financement participatif réalisée en octobre 2017*
- *Sortie continuellement repoussée -> octobre 2019…*

Android

- *Lancé en juin 2007*
- *La plus grande croissance en 2010 dans le marché smartphone*
- *Open Handset Alliance : Google, mais aussi Motorola, HTC, Samsung, Sony(-Ericsson), HP, Dell, Huawei, Lenovo …*
- *Depuis 2011, Android est l'OS mobile le plus vendu, ~80% de part de marché en 2019*
- *Une mise à jour majeure par année*
  Actuellement, version 10.0 (Q) – début septembre 2019

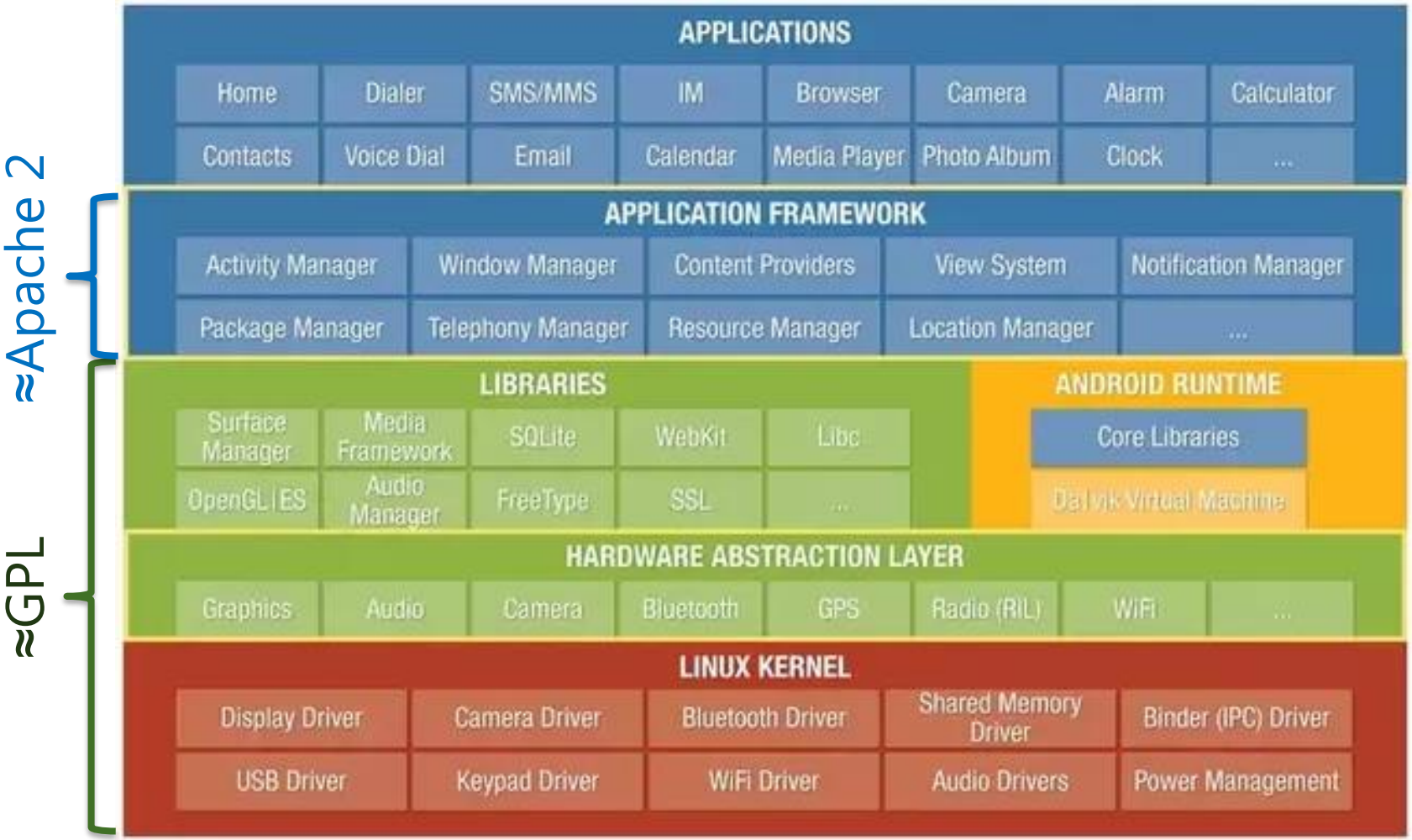
Android

- *Développé à l'origine par la start-up Android, rachetée depuis par Google*
- *Noyau Linux, machines virtuelles «maison» (Dalvik puis ART) pour abriter les applications et les isoler dans une sandbox*
- *Multi-tâche natif, open source, ce qui permet les déclinaisons personnalisées:*
  - *Surcouches: HTC Sense, équivalents chez Sony, LG ou Samsung*
  - *Forks: Kindle Fire avec Fire OS, Cyanogen, etc.*

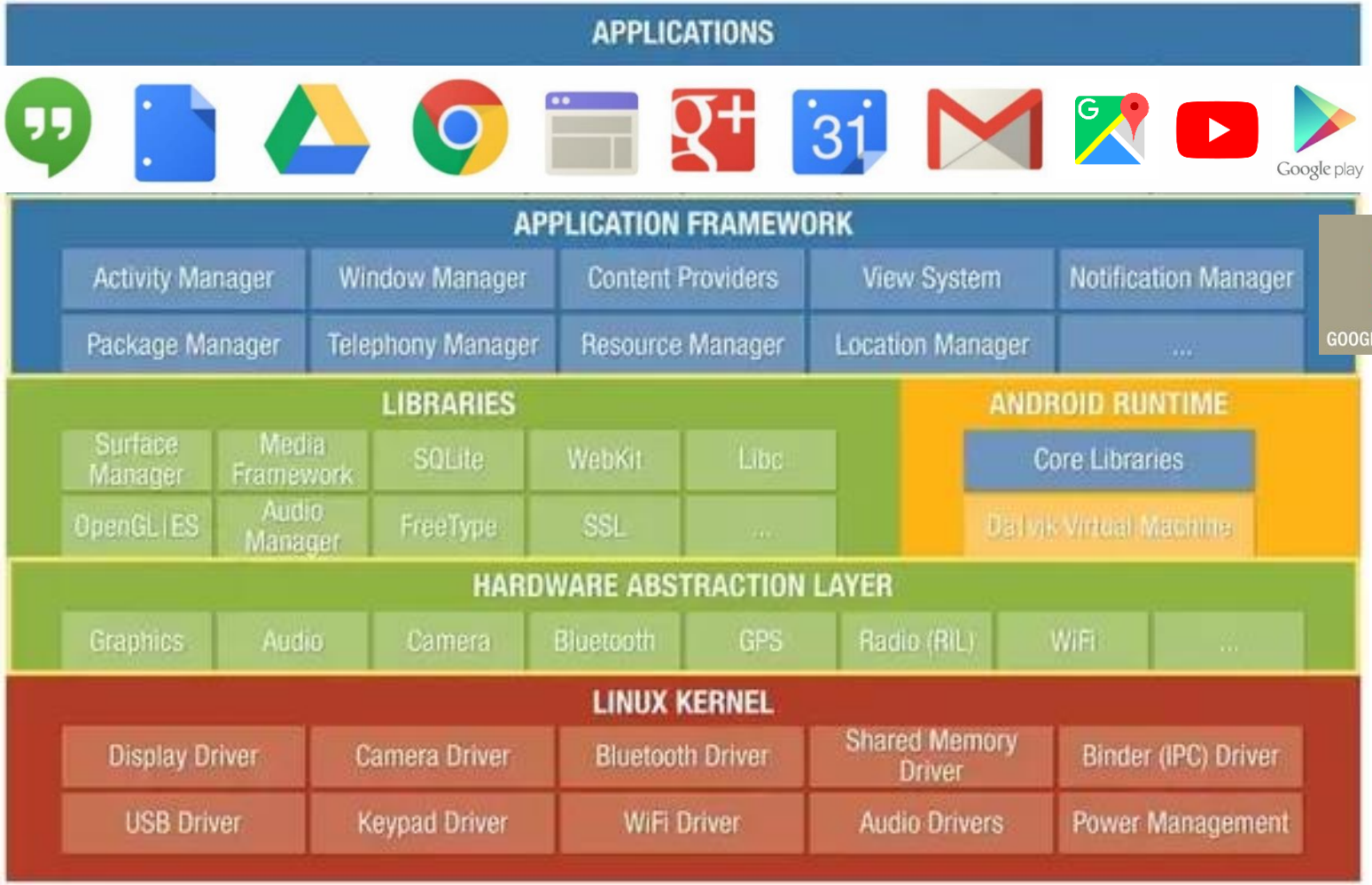
Android

- *La multiplicité des constructeurs et des architectures impose l'utilisation d'une machine virtuelle*
- *Dalvik dans sa dernière version était d'une efficacité surprenante – compilation JIT*
- *Dalvik est remplacé par une nouvelle VM (ART) à partir de 2014 – compilation de toutes les applications à l'installation*
- *L'implémentation Java sur Android est calquées sur le JDK*
  → Procès en cours avec Oracle depuis 2011
- *Depuis 2017, Kotlin est devenu le second langage supporté officiellement sur la plateforme Android*

Android – Architecture Android Open Source Project (AOSP)



Android – Architecture «GOOGLE»



Propriétaire

Android «GOOGLE» – Actualités

BUSINESS INSIDER FRANCE

TECH ÉCONOMIE POLITIQUE STRATÉGIE LIFESTYLE NEWSLETTERS

Trump announces ban on US government agencies doing business with Huawei will continue

Antonio Villas-Boas 9 Aug 2019, 14:51

Le Monde 30.08.2019

Huawei : les services Google seront totalement absents du prochain smartphone Mate 30

Le constructeur chinois va devoir se passer des services Google pour mobiles, y compris l'accès au Play Store, sur ses prochains modèles haut de gamme, dont la sortie est prévue dans les prochains mois.

forbes.com

148,227 views | Aug 27, 2019, 01:08am

New Huawei OS Shock: 'Confirmation' Of Russian Software For Mobile Devices

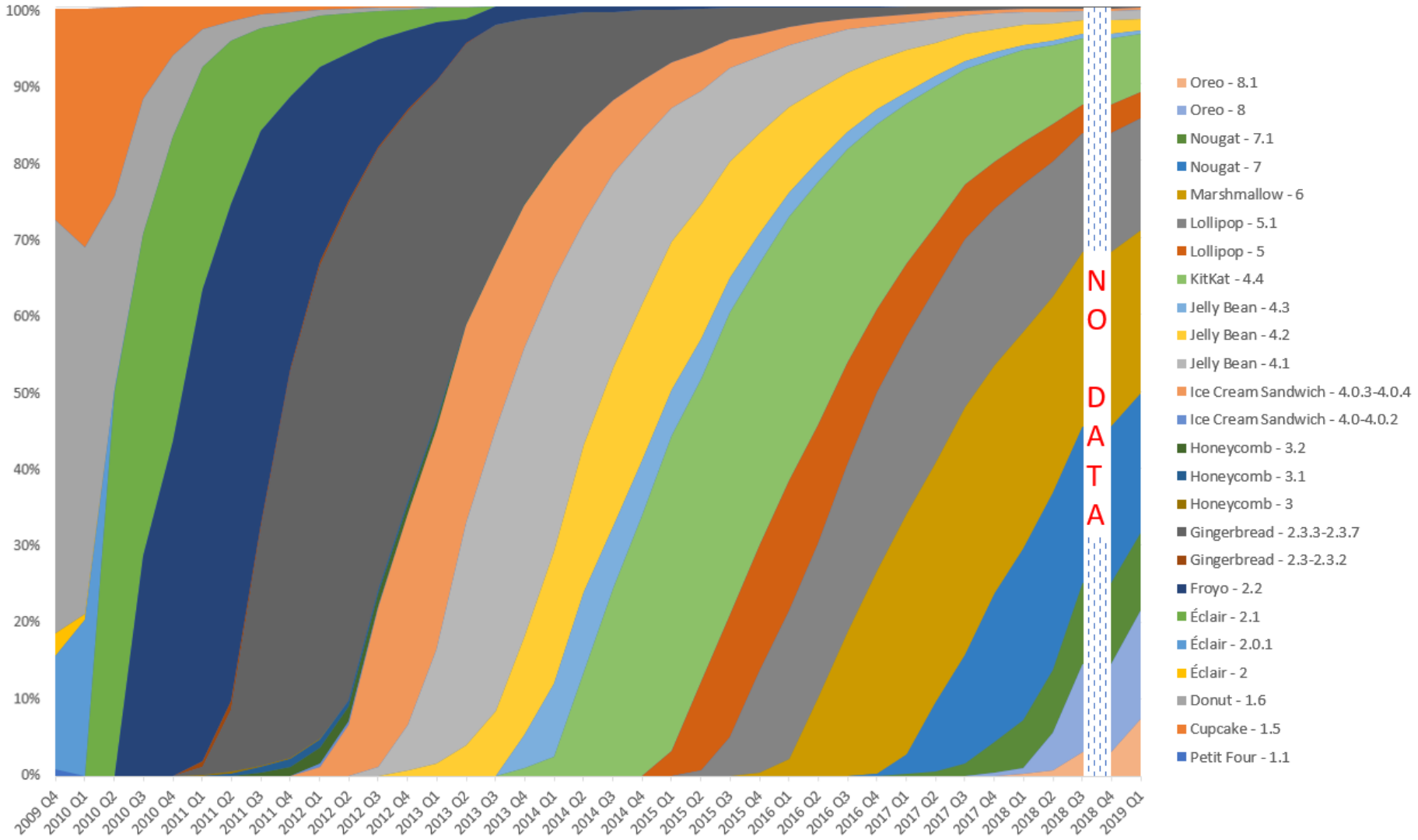
OS universel

Huawei dévoile Harmony OS, son alternative «light» à Android

Ven 09.08.2019 - 15:37
par Rodolphe Koller

ICTjournal

Android



Le parc de téléphones Android est très fragmenté

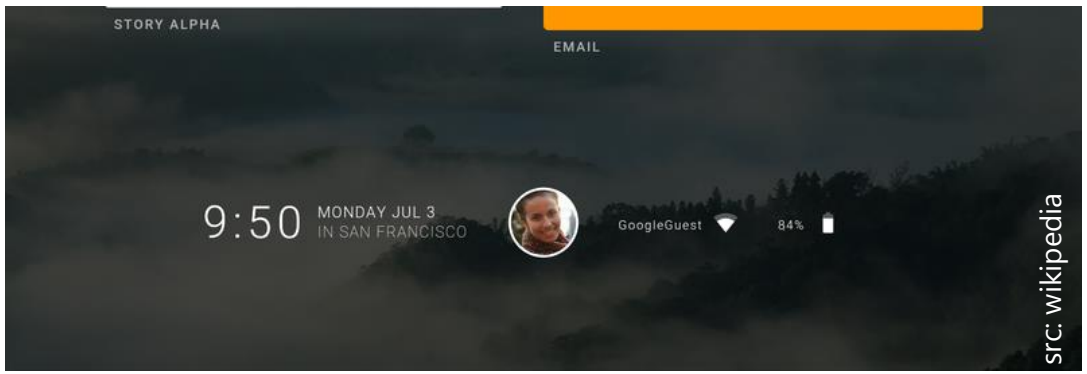
src: wikipedia, février 2019

Android – Kotlin

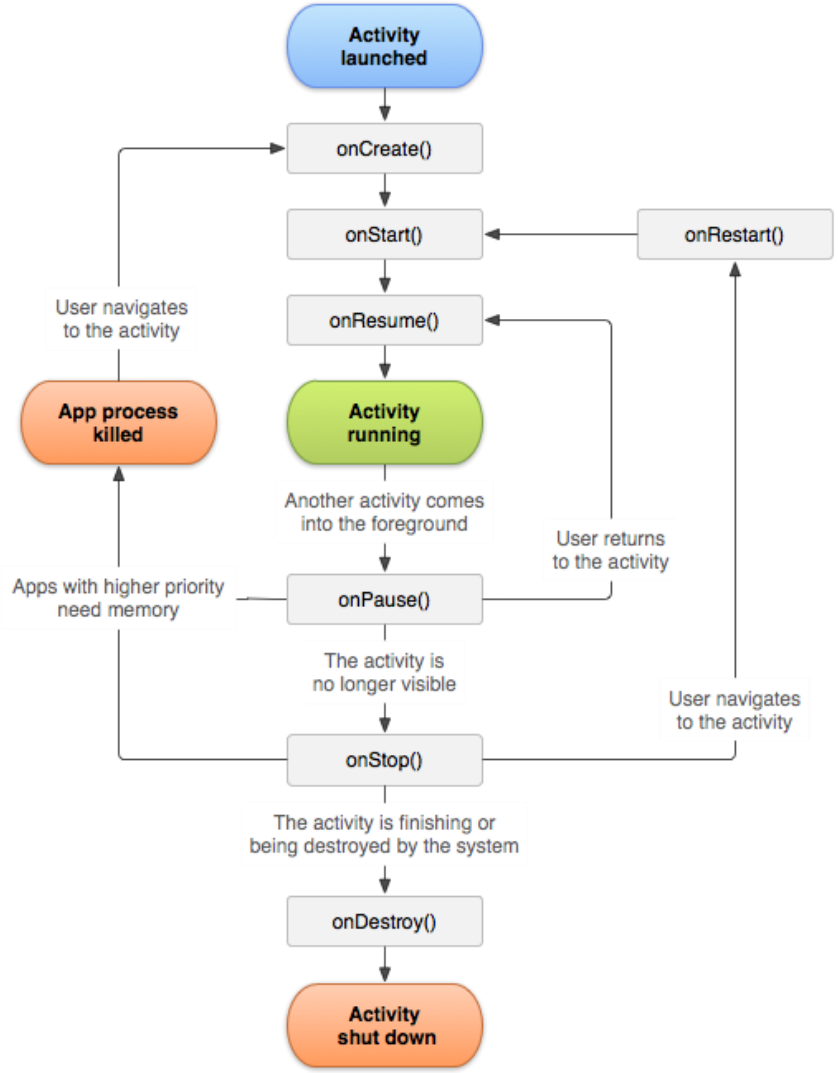
- *Langage de programmation orienté objet et fonctionnel. Développé principalement par JetBrains (Android Studio, IntelliJ, CLion)*
- *Peut être «compilé» en bytecode Java compatible avec la JVM ou en Javascript*
- *Outils permettant de convertir du code Java automatiquement*
- *Interopérabilité complète avec du code en Java*
- *Présente de très nombreuses similarités avec le langage Swift d'Apple*

Successeur d'Android ?

- *Google Fuchsia*
- *Abandon du noyau Linux pour un µ-kernel appelé Zircon (Magenta)*
- *Release initiale en août 2016*
- *Une première interface graphique est disponible depuis mai 2017*
- *Un site est apparu en juillet 2019 https://fuchsia.dev/, à suivre…*



Android – Les activités



Une ACTIVITE (Activity) est un élément applicatif visible. Une activité est initiée à l'aide d'une INTENTION (Intent) qui contient les paramètres significatifs pour cette application.
Une Activity peut au besoin être « tuée » par l'OS

Android – Les Intents

Il existe deux types d'Intents sur Android:

- *Explicites*
  Les intents explicites servent à démarrer une activité précise, faisant généralement partie de la même application

```
Intent intent = new Intent(this, MyActivity.class);
startActivity(intent);
```

- *Implicites*
  Les intents implicites servent à démarrer une action, généralement supportée par le système, sans préciser quelle activité devra être ouverte

```
Intent intent = new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse("http://www.heig-vd.ch"));
startActivity(intent);
```

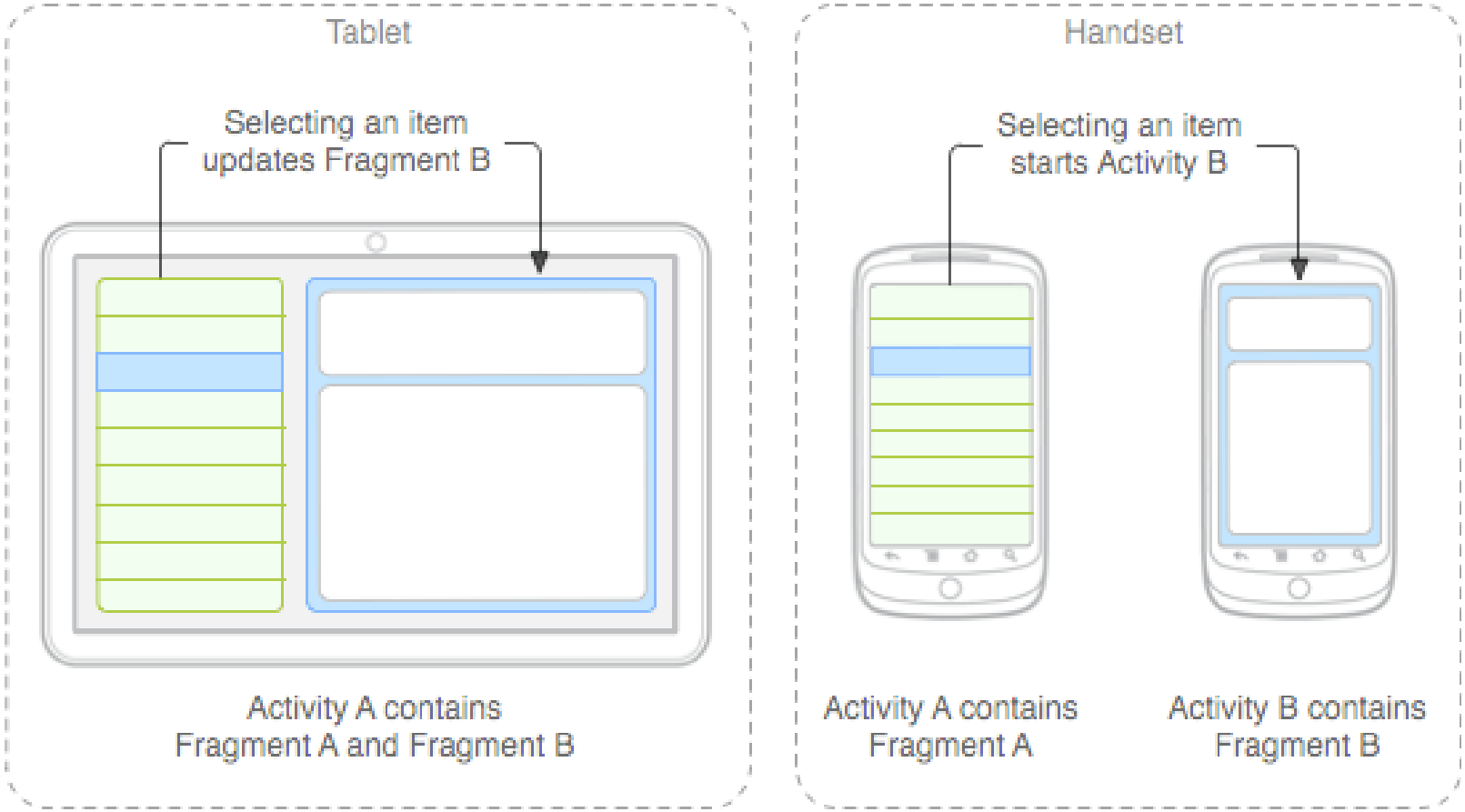
Android – Les Fragments

- Un fragment est une partie réutilisable d'interface graphique; il permet de mieux structurer une activité en la divisant en sous-ensembles réutilisables

- Les fragments sont souvent utilisés en conjonction avec des interfaces à onglets : chaque onglet est alors implémenté par un fragment faisant partie d'une activité regroupant tous les onglets

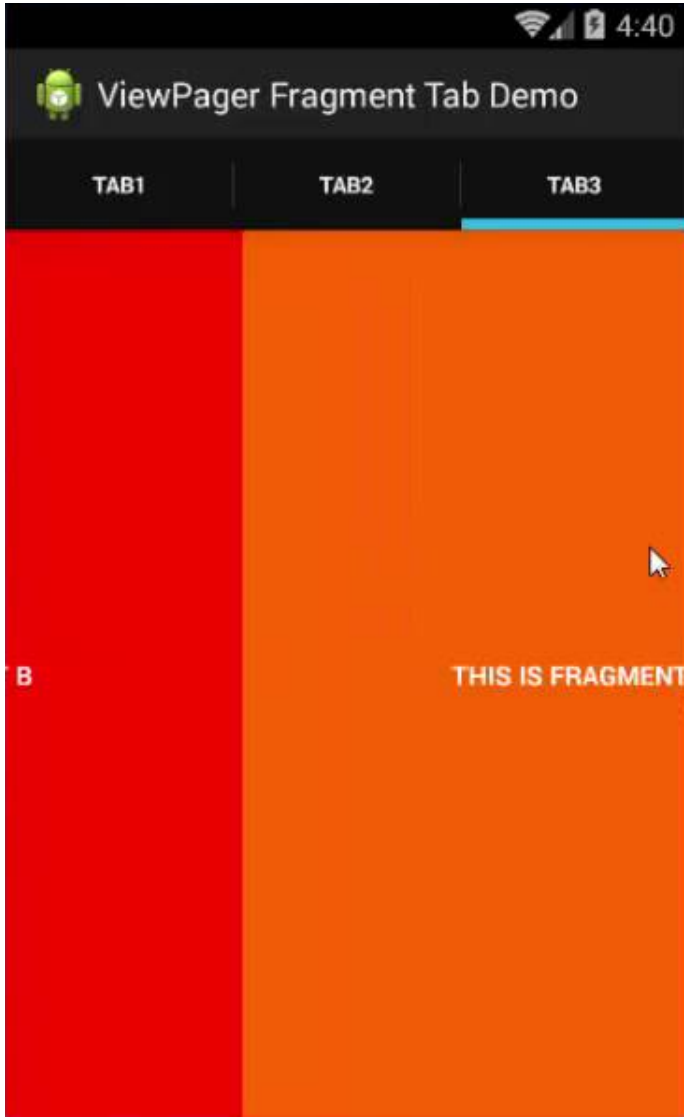
- Un fragment est instancié dans une activité

Android – Les Fragments

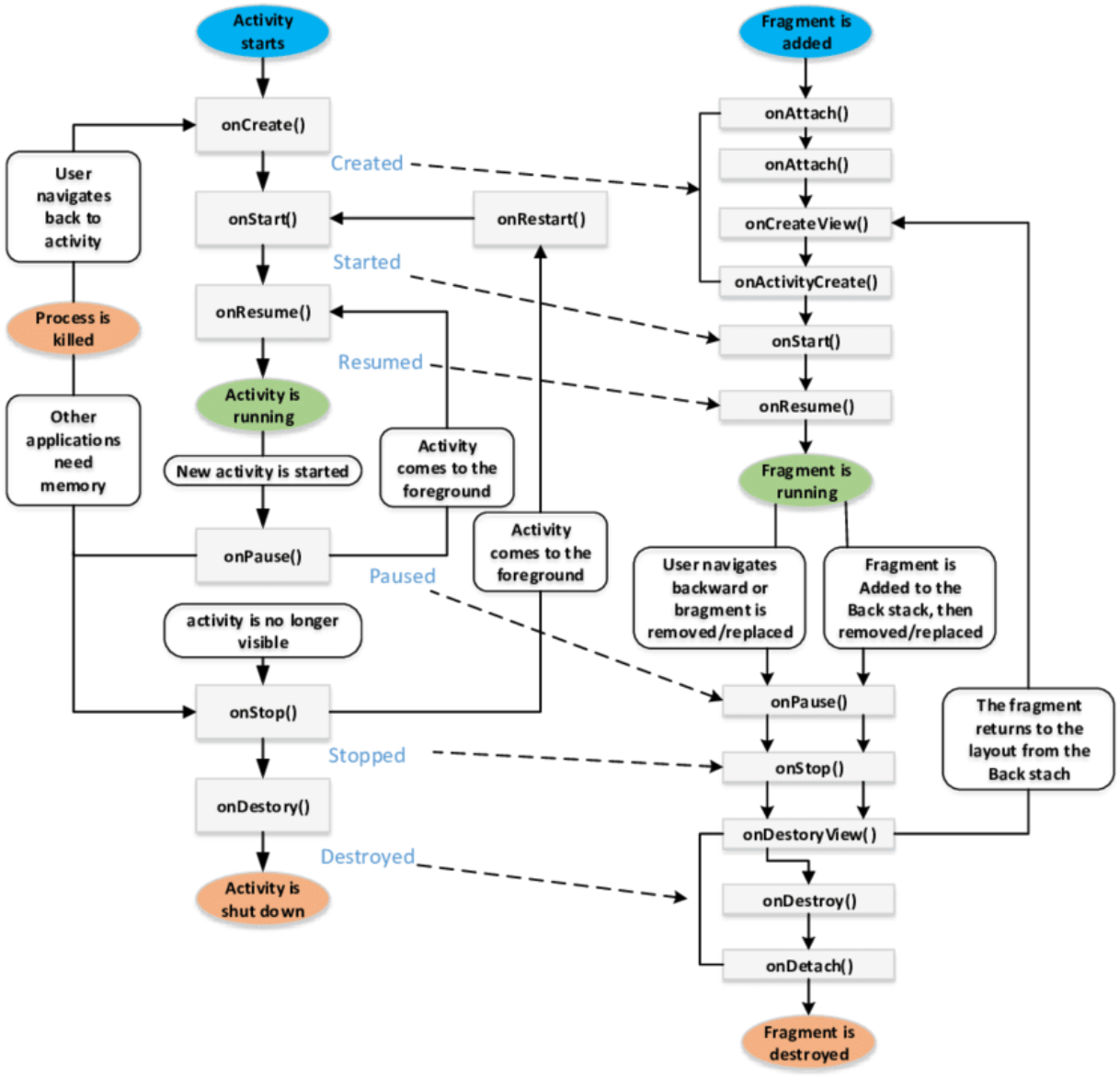


src: The Busy Coder's Guide to Android Development - Mark L. Murphy

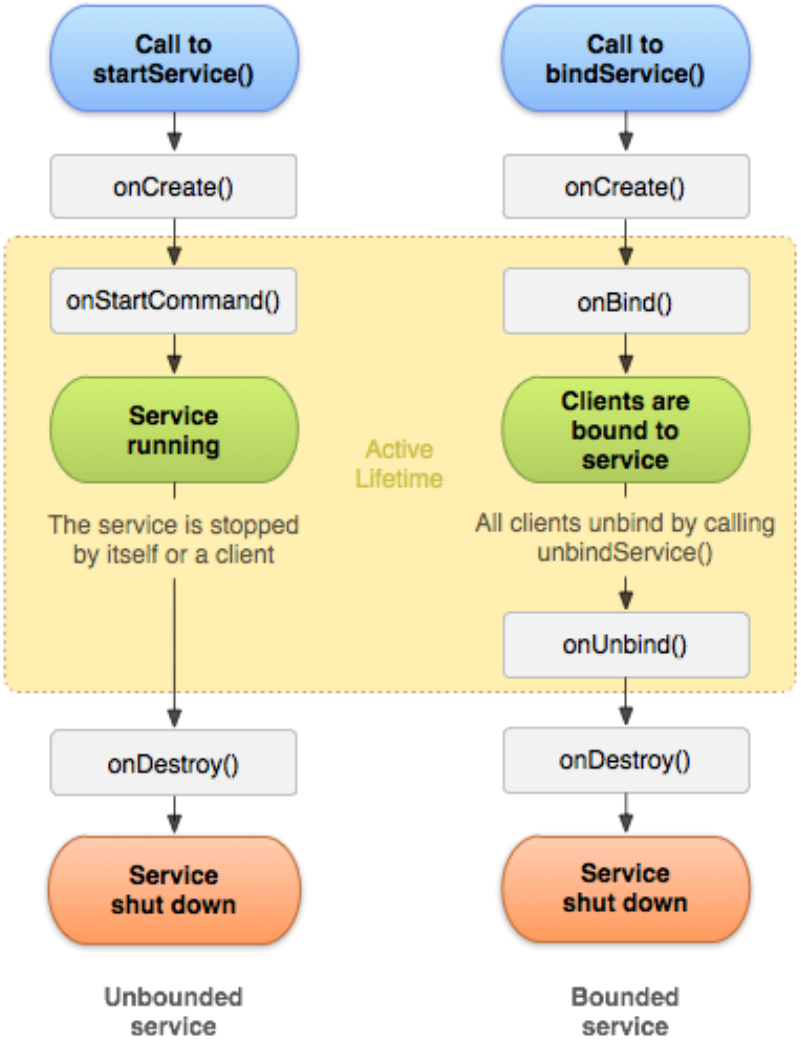
Android – Les Fragments



Android – Les Fragments



Android – Les Services



Un SERVICE est une tâche de fond, sans interactions avec l'écran ou l'utilisateur.
Typiquement lorsqu'un service veut signaler quelque chose à l'utilisateur, il émet une NOTIFICATION. Cette notification sera visible (audible, sensible) dans la barre de notifications de l'écran
Un SERVICE est exécuté dans le thread principal

Android – Les Services

A quoi servent-ils ?

- *A exécuter des opérations qui doivent continuer même si l'utilisateur quitte les activités de l'application (long téléchargement, lecture de musique, etc.)*

- *A maintenir certaines fonctionnalités quels que soient les aller-retours de l'utilisateur sur l'application (par ex.: maintenir la connexion sur un serveur e-mail)*

- *Lorsque l'on doit fournir une API à d'autres applications installées sur le smartphone*

Il existe à présent d'autres options offertes par le SDK permettant de se passer de l'utilisation de services, ceux-ci sont généralement compliqués à prendre en main

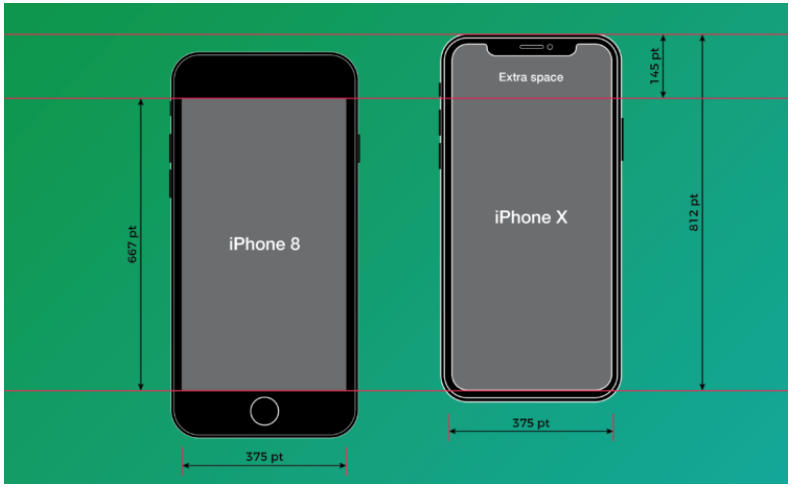
Android

- *Un système de développement complet est disponible sur Android Studio (IDE basé sur IntelliJ)*
- *Les écrans sont en principe décrits en XML*
- *Un fichier AndroidManifest.xml décrit l'application et définit ses propriétés et ses privilèges*
- *Du code C/C++ peut être intégré dans une application (Android NDK), la démarche est conseillée uniquement pour des taches demandant beaucoup de performances*

Android

- *Avoir des tailles d'écran variables implique certaines précautions lors du développement d'interfaces utilisateurs*

- *Les développeurs iOS en ont fait l'amer expérience lors des changements des tailles d'écrans des iPhones*



- *Dans le cadre d'Android, cette difficulté a été partiellement résolue en utilisant la notion de «Layout»*

Android – Les Layouts

Un layout (littéralement : disposition) définit une stratégie de placement des éléments de dialogue sur l'écran

La stratégie peut s'appliquer :

- *relativement à d'autres éléments de dialogue et aux bords de l'écran (RelativeLayout ou ConstraintLayout)*
- *simplement empilés à la suite les uns des autres (LinearLayout)*
- *de manière tabulaire (TableLayout)*
- *Etc.*

Android – Les Layouts

- *Les layouts peuvent bien sûr être combinés et imbriqués, c'est le rôle du fichier XML de description d'écran (mavue.xml)*
- *Certaines vues spécialisées implémentent leur propre layout (ListView, par exemple)*
- *Pour interagir programmatiquement avec les éléments d'interface, il faut «linker» le code avec la vue, en utilisant la référence sur l'élément R.id.monelement*

Android – Les Layouts

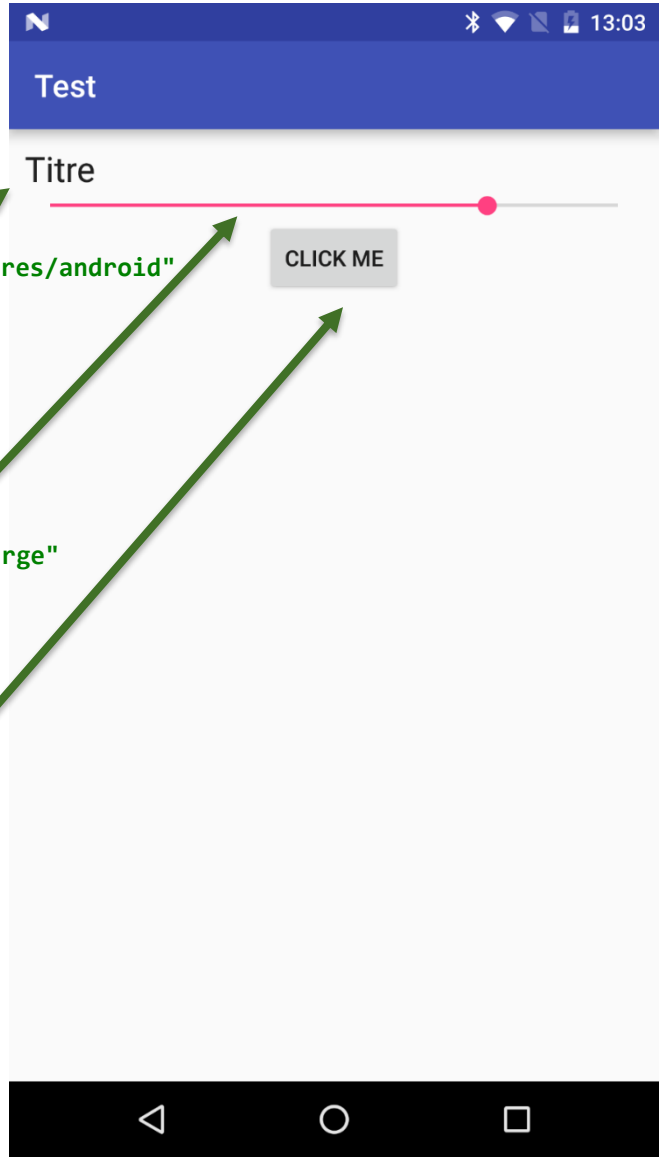
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="10dp">

    <TextView
        android:id="@+id/mon_titre"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="@string/title"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true" />

    <SeekBar
        android:id="@+id/ma_progressBar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/mon_titre"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true" />

    <Button
        android:id="@+id/mon_bouton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/clickme"
        android:layout_below="@+id/ma_progressBar"
        android:layout_centerHorizontal="true" />

</RelativeLayout>
```



Android – Les Layouts

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schema..."
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="10dp">

    <TextView
        android:id="@+id/mon_titre"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/title"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"/>

    <SeekBar
        android:id="@+id/ma_progressBar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/mon_titre"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"/>
```

...

- Layout «racine» de la vue
- Hauteur et largeur avec la valeur **match_parent**: le layout va occuper tout l'espace disponible à l'écran
- Padding à **10dp**, les éléments dans ce layout ne seront pas collés au bord de l'écran

Unités layout android:

px pixels 👎
in **mm** valeur physique 👎
pt points - 1/72 of an inch 👎
dp density-independant pixels
1 **dp** is 1 **px** sur un écran 160 dpi
sp scale-independent pixels

Android – Les Layouts

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schema..."
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="10dp">

    <TextView
        android:id="@+id/mon_titre"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/title"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"/>

    <SeekBar
        android:id="@+id/ma_progressBar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/mon_titre"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"/>

...
```

- Vue textuelle
- **@+id/mon_titre** est l'identifiant de la vue, il permettra de la linker dans le code
- La hauteur est indiquée avec la constante **wrap_content**, la vue prendra donc automatiquement la hauteur de son contenu. Taille de police plus grande, long texte sur un petit écran (plusieurs lignes), etc
- Le texte qui sera affiché dans la vue est la ressource **@string/title** du fichier *strings.xml*
- Cette vue est placée directement dans un **RelativeLayout**, on doit donc la positionner par rapport à celui-ci. On l'aligne en haut à gauche et à droite, la hauteur est automatique

Android – Les Layouts

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schema..."
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="10dp">

    <TextView
        android:id="@+id/mon_titre"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/title"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true" />

    <SeekBar
        android:id="@+id/ma_progressBar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/mon_titre"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true" />

...
```

- Curseur de défilement permettant de sélectionner une valeur numérique
- On retrouve quasiment les mêmes paramètres que pour la **`TextView`**
- La principale différence est que cette vue (≡ élément de vue) n'est pas aligné en haut du layout (elle recouvrirait la **`TextView`**) mais est placée dessous (**`below`**) la vue précédente référecée par son identifiant **`@+id/mon_titre`**

Android – L'interaction Layout / Activité

```
public class MonActivite extends Activity {

    private TextView    title       = null;
    private SeekBar     progress    = null;
    private Button      button      = null;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.mavue);

        //link to GUI
        this.title      = findViewById(R.id.mon_titre);
        this.progress   = findViewById(R.id.ma_progressBar);
        this.button     = findViewById(R.id.mon_bouton);

        //events
        this.button.setOnClickListener(new View.OnClickListener() {
            @Override public void onClick(View view) {
                Toast.makeText(MonActivite.this,
                        R.string.clickme_message,
                        Toast.LENGTH_LONG).show();
            }
        });
    }

}
```

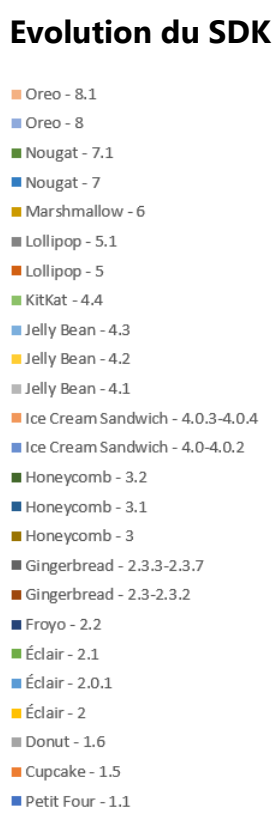
On choisit le layout

On lie les vues de la GUI au code afin de pouvoir interragir ensuite avec

On traite les événements UI

Programmer – AndroidX

- *Remplacement et amélioration de la Support Library fin 2018*
- *Intégration d'Android Jetpack*



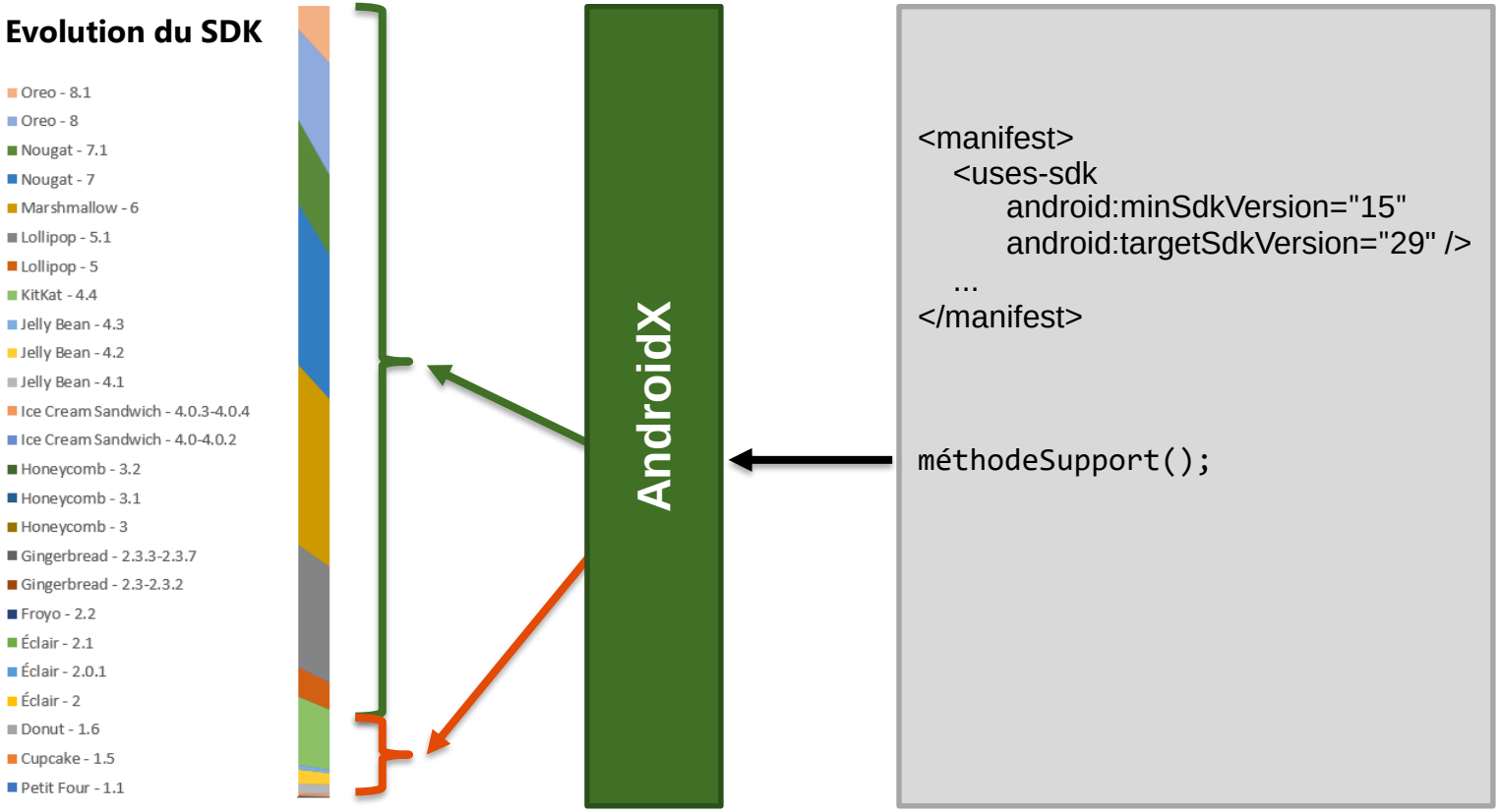

```
<manifest>
   <uses-sdk
        android:minSdkVersion="15"
        android:targetSdkVersion="29" />
   ...
</manifest>



if(Build.VERSION.SDK_INT >=
              Build.VERSION_CODES.LOLLIPOP) {
        nouvelleMéthodeSDK();
} else {
        méthodeSDK();
}
```

Programmer – AndroidX

- *Remplacement et amélioration de la Support Library fin 2018*
- *Intégration d'Android Jetpack*



Programmer – Android Jetpack

Foundation

Foundation components provide core system capabilities, Kotlin extensions and support for multidex and automated testing.

AppCompat

Degrade gracefully on older versions of Android

Android KTX

Write more concise, idiomatic Kotlin code

Multidex

Provide support for apps with multiple DEX files

Test

An Android testing framework for unit and runtime UI tests

Architecture

Architecture components have classes that help manage your UI component lifecycle, handle data persistence, and more.

Data Binding

Declaratively bind observable data to UI elements

Lifecycles

Manage your activity and fragment lifecycles

LiveData

Notify views when underlying database changes

Navigation

Handle everything needed for in-app navigation

Paging

Gradually load information on demand from your data source

Room

Fluent SQLite database access

ViewModel

Manage UI-related data in a lifecycle-conscious way

WorkManager

Manage your Android background jobs

Behavior

Behavior Components help you design robust, testable, and maintainable apps.

Download manager

Schedule and manage large downloads

Media & playback

Backwards compatible APIs for media playback and routing (including Google Cast)

Notifications

Provides a backwards-compatible notification API with support for Wear and Auto

Permissions

Compatibility APIs for checking and requesting app permissions

Sharing

Provides a share action suitable for an app's action bar

Slices

Create flexible UI elements that can display app data outside the app

UI

UI components make it easy for you to make your app not only easy, but delightful to use.

Animation & transitions

Move widgets and transition between screens

Auto

Components to help develop apps for Android Auto.

Emoji

Enable an up-to-date emoji font on older platforms

Fragment

A basic unit of composable UI

Layout

Lay out widgets using different algorithms

Palette

Pull useful information out of color palettes

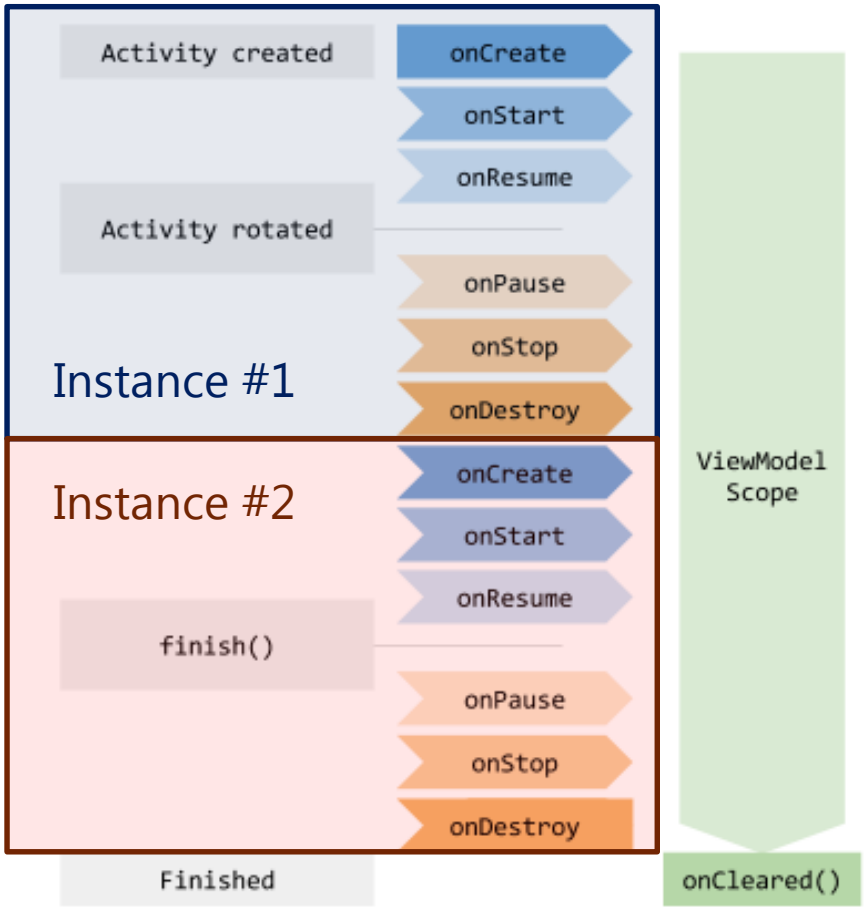
TV

Components to help develop apps for Android TV.

Wear OS by Google

Components to help develop apps for Wear.

Programmer – Android Jetpack – *ViewModel*



- *La classe ViewModel est conçue pour stocker et gérer les données relatives à l'interface utilisateur d'une manière tenant compte du cycle de vie.*
- *Elle permet aux données de survivre aux changements de configuration tels que les rotations d'écran.*

Programmer – Android Jetpack – *Room*

- *ORM pour SQLite*
- *Permet une gestion «haut niveau» de la base de données*
- *Simplifie la migration d'une version à une autre*

```java
@Dao
public interface ContactDAO {
    @Insert
    public void insert(Contact... contacts);

    @Update
    public void update(Contact... contacts);

    @Delete
    public void delete(Contact contact);

    @Query("SELECT * FROM contact")
    public List<Contact> getContacts();

    @Query("SELECT * FROM contact WHERE phoneNumber = :number")
    public Contact getContactWithId(String number);
}
```

Programmer – Android Jetpack – *LiveData*

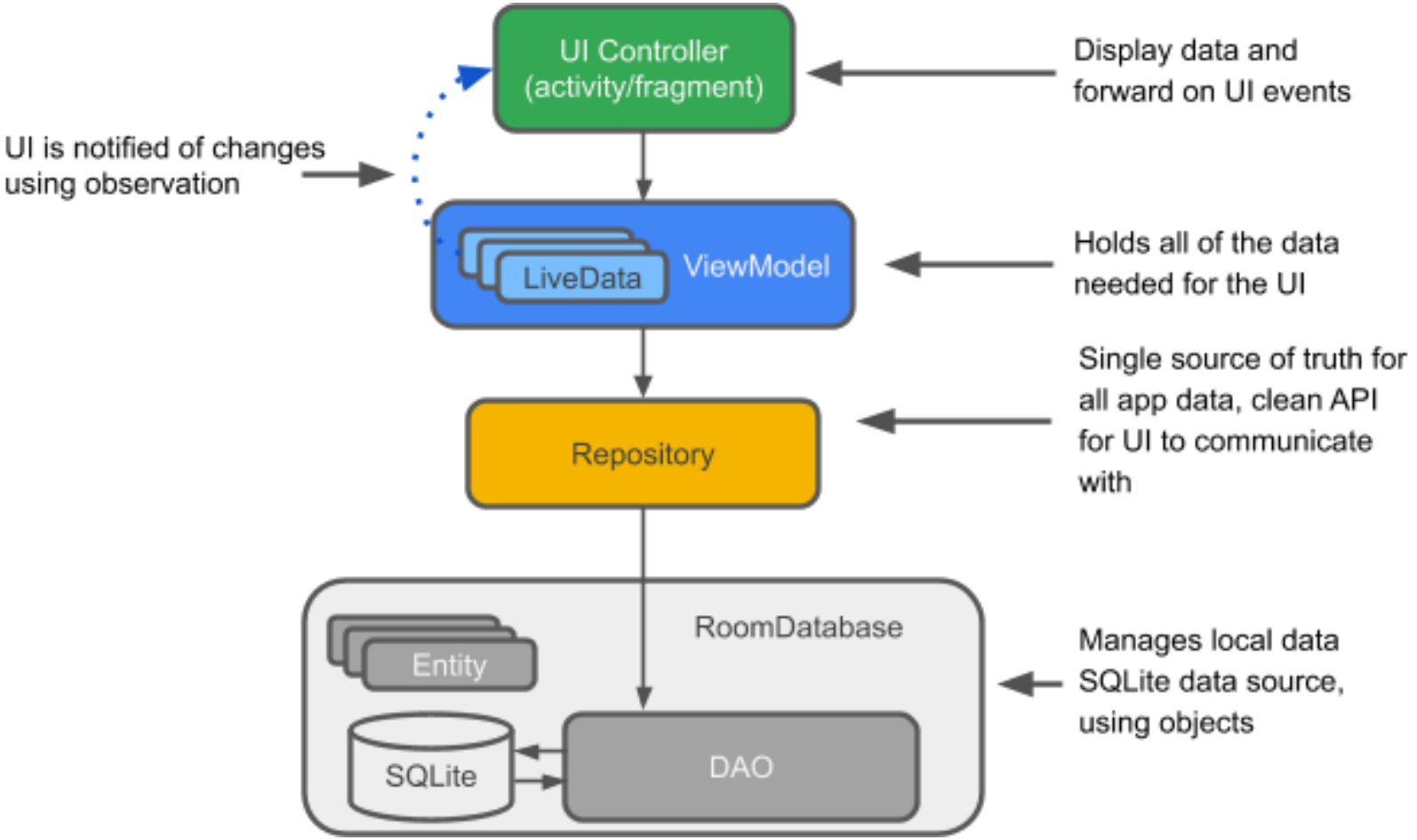
- *Permet d'encapsuler des données*
- *Et d'enregistrer des évènements qui seront automatiquement lancés lorsque la valeur change*
- *Tient compte du cycle de vie Android*

```
MutableLiveData<Integer> myLiveInt = new MutableLiveData<>();
myLiveInt.setValue(0);

findViewById(R.id.click_me_button).setOnClickListener((view) -> {
    myLiveInt.postValue(myLiveInt.getValue()+1);
});

myLiveInt.observe( owner: this, integer -> {
    Toast.makeText( context: this,  text: ""+ integer, Toast.LENGTH_SHORT).show();
});
```

Programmer – Android Jetpack – Série d'exercices 1

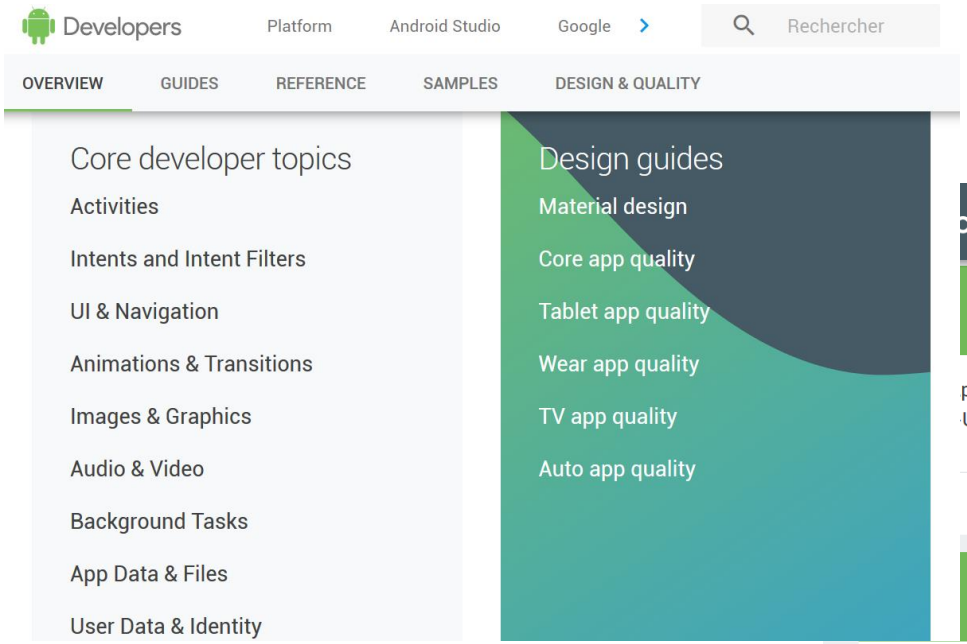


Programmer – Android

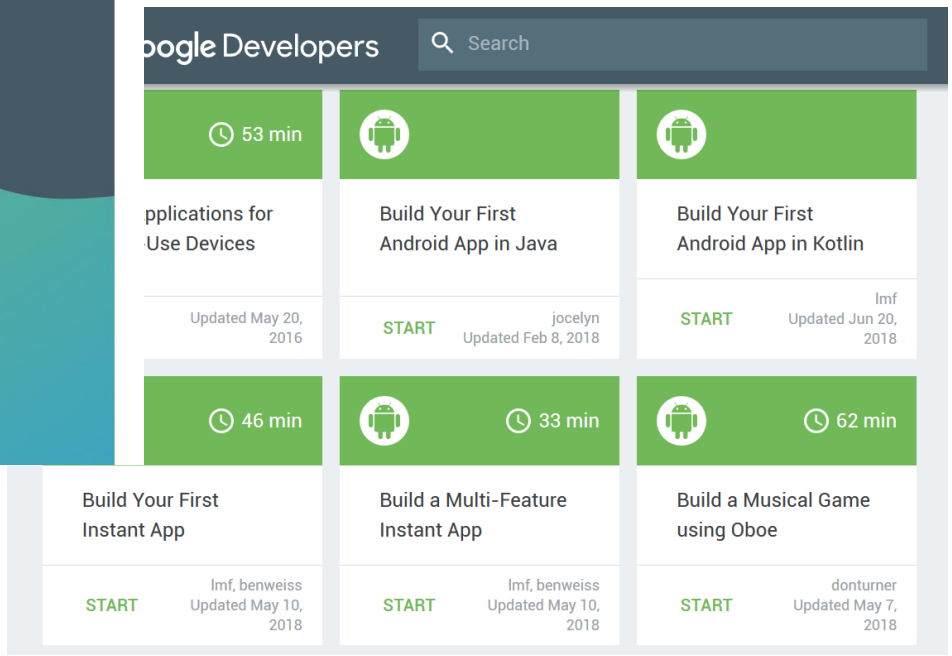
- *Le but de ce cours n'est pas de faire de vous des programmeurs Android chevronnés, mais quelques éléments sont indispensables*
- *Un grand principe de la programmation en environnement de grande mobilité est :*
  - ***Tout ce qui n'est pas directement lié à une interaction directe et immédiate avec l'utilisateur doit être délégué à une unité d'exécution (thread) séparée***

Programmer – Android

Quelques ressources complémentaires



https://developer.android.com/docs/



https://codelabs.developers.google.com/?cat=Android

Evolutions

- *Périphérie actuelle : Bluetooth, Bluetooth LE, NFC, HDMI, USB master, Chargeur sans-fil*
- *Nécessité de l'établissement d'une norme définissant un sous-ensemble minimal*
  - *Oppositions très fortes à cet établissement (Apple)*

Applications natives ?

- *Beaucoup de spécialistes pensent que l'avenir des terminaux portables va vers la notion de "thin client"*
- *Ces derniers ont de gros besoins en communication et interagissent plutôt mal avec leur environnement*
- *Cette prédiction ne se vérifiera probablement que pour les applications de pure consultation*

Applications natives ?

- *Les applications natives doivent être réécrites et maintenues pour toutes les plateformes*
- *Ce surcoût pourrait inciter certaines entreprises à imposer une plateforme spécifique pour leurs applications*
- *Ceci implique que les utilisateurs seraient amenés à utiliser la plateforme d'entreprise pour leur travail, indépendamment de ce qu'ils utilisent en privé*

Applications natives ?
Android Instant Apps

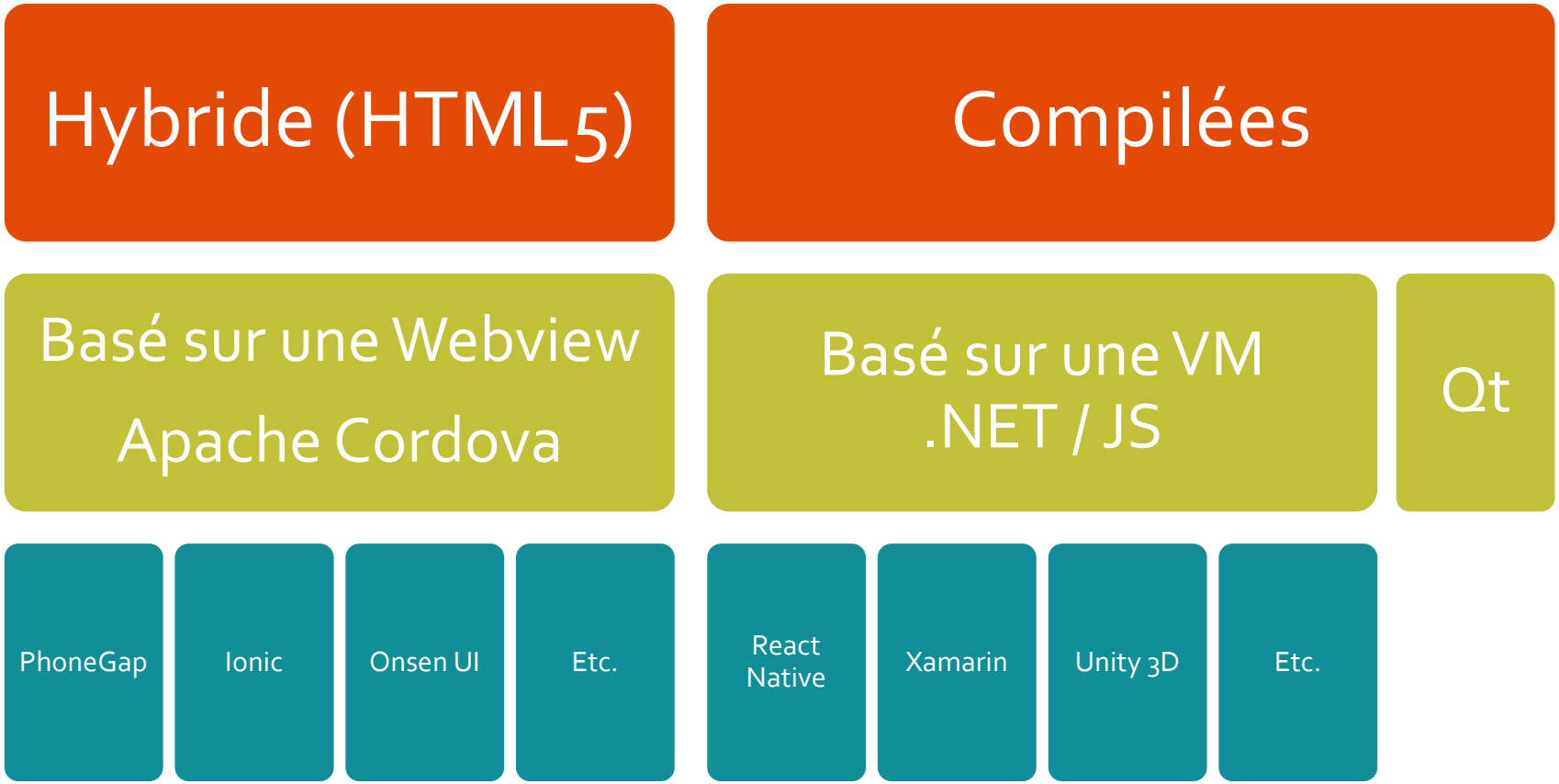
- *Disponibles depuis mai 2017, les cas d'utilisations ne sont pas encore totalement définis*
- *Il s'agit d'applications (ou sous-partie d'applications) natives ne nécessitant pas d'installation préalable. Elles se téléchargent et s'ouvrent automatiquement en cliquant par exemple sur un hyperlien ou en scannant un tag NFC*
- *Voir comment les aspects de sécurité seront traités…*

Applications natives ? Cross-plateformes

- *Technologies web (HTML/CSS/JS) packagées sous la forme d'applications natives*
  Très souvent basées sur Apache Cordova
  - *Ionic (AngularJS)*
  - *Adobe PhoneGap*

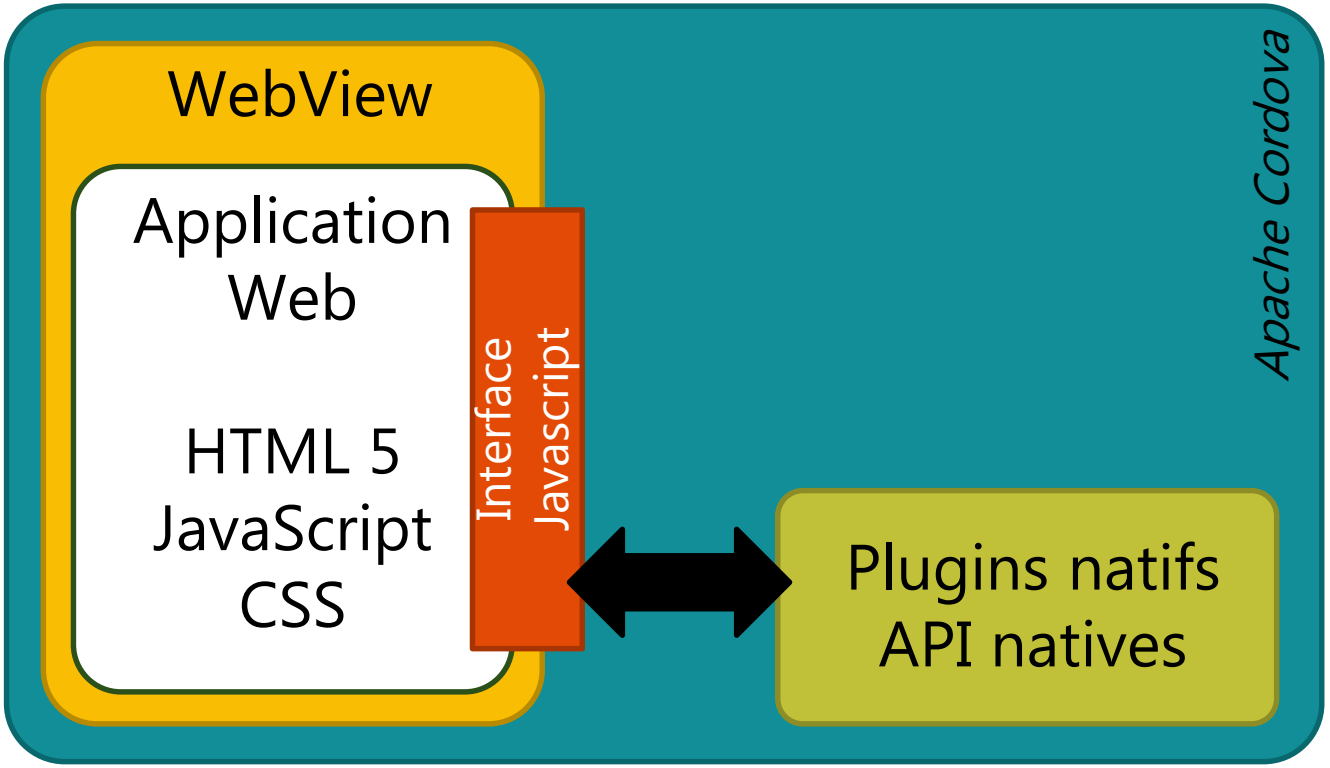
- *Génération de code «natif» Android/iOS à partir d'un code unique:*
  - *React Native (Facebook)*
  - *Xamarin (Microsoft)*
  - *Unity3D (Moteur de jeu vidéo)*
  - *Google Flutter*

Applications natives ? Cross-plateformes



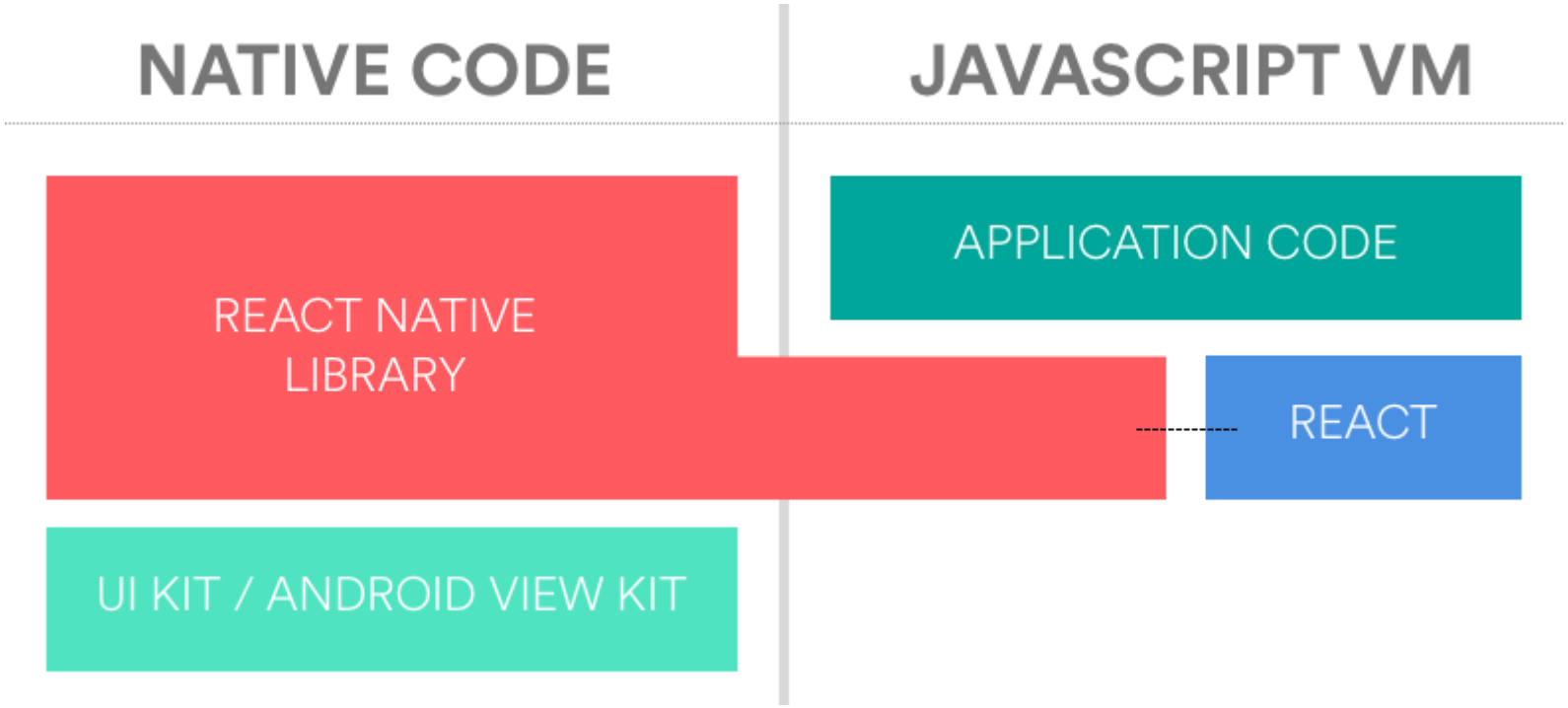
Applications natives ? Cross-plateformes

- Hybride (HTML5)

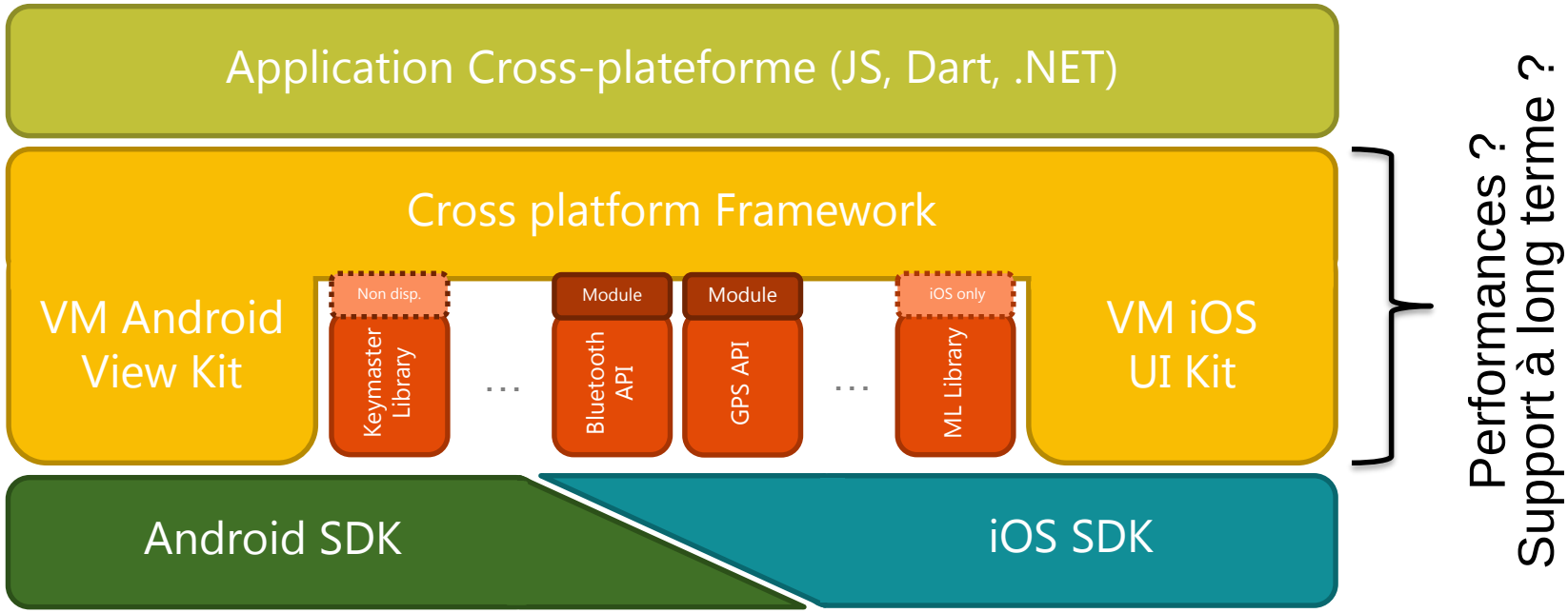


Applications natives ? Cross-plateformes

- Basé sur une VM, ex. React Native



Applications natives ? Cross-plateformes



Applications natives ? Cross-plateformes

- *Le choix entre un développement natif ou cross-plateforme va principalement dépendre des fonctionnalités de votre application*
- *Une application «simple» est certainement une bonne candidate à un développement cross-plateforme*
- *Si l'accès à des composants hardwares spécifiques (lecteur d'empreintes digitales, profiles bluetooth spécifiques, NFC, etc.) du smartphone est nécessaire, nous nous tournerons plus facilement vers un développement natif*

Applications natives ? Cross-plateformes

- *Aucune technologie cross-plateforme n'est parfaite, surtout si elles sont encore jeunes*
- *Evolution très rapide – par exemple env. 1 version/mois pour React Native*
  - *Ajout nouvelles fonctionnalités, nouveaux modules*
  - *Peut nécessiter de modifier le code existant*
- *Bien qu'open source, Facebook en assure le développement. Si Facebook lâche le développement cela peut s'arrêter très vite*
- *Exemple avec le framework Web Sproutcore, supporté initialement par Apple (MobileMe, iWork) a connu un développement très rapide, avant de se stabiliser et devenir un choix intéressant. Apple, lors du passage à iCloud a totalement abandonné ce framework*

Applications natives ? Cross-plateformes

Quelques retours de diplômants:

- *React-native*
  «Il aurait peut être fallu effectuer plus de tests en début de projet et ainsi choisir une autre technologie avec laquelle je n'aurai peut être pas eu de problème »

- *Flutter*
  « suffisamment documentés pour arriver rapidement et aisément à un résultat satisfaisant, et si c'était à refaire il figurerait à nouveau dans le projet »

Applications natives ? Cross-plateformes

- *Une possibilité très intéressante offerte par la plupart des frameworks basés sur JavaScript est la mise-à-jour « à chaud »*
- *L'application télécharge une nouvelle version du fichier JavaScript sans passer par une mise-à-jour complète*
- *Etonnamment Apple n'interdit pas cette pratique, mais…*

3.3.2 Except as set forth in the next paragraph, an Application may not download or install executable code. Interpreted code may be downloaded to an Application but only so long as such code: (a) does not change the primary purpose of the Application by providing features or functionality that are inconsistent with the intended and advertised purpose of the Application as submitted to the App Store, (b) does not create a store or storefront for other code or applications, and (c) does not bypass signing, sandbox, or other security features of the OS.

- (i) Your app should work on its own without requiring installation of another app to function.
- (ii) Make sure you include sufficient content in the binary for the app to function at launch.
- (iii) If your app needs to download additional resources, disclose the size of the download and prompt users before doing so. Existing apps must comply with this guideline in any update submitted after January 1, 2019.

Applications natives ? Cross-plateformes

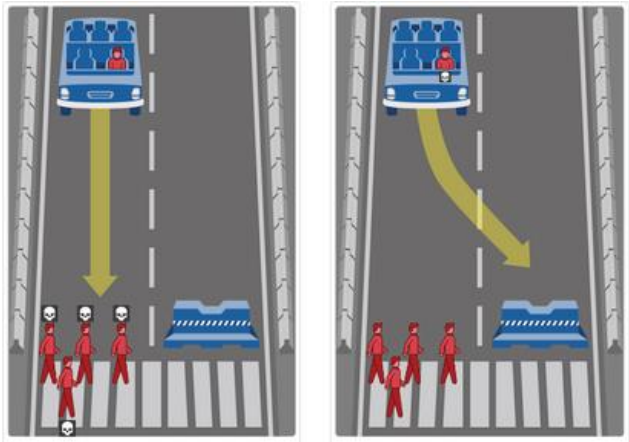
Dans tous les cas, votre application devra passer au moins une première fois la validation Apple:

4.2 Minimum Functionality

Your app should include features, content, and UI that elevate it beyond a repackaged website. If your app is not particularly useful, unique, or "app-like," it doesn't belong on the App Store. If your App doesn't provide some sort of lasting entertainment value, or is just plain creepy, it may not be accepted. Apps that are simply a song or movie should be submitted to the iTunes Store. Apps that are simply a book or game guide should be submitted to the iBooks Store.

Systèmes mobiles autonomes

L'évolution pour les prochaines années



- *Automobiles autonomes (problèmes plus juridiques que techniques)*
- *Prothèses autonomes à connexion neurale (chaises roulantes, exosquelettes, membres articulés : cas de Cathy Hutchinson, vidéos sur YouTube)*
- *«Killer robots» (Une commission aux Nations Unies)*