



Android



Jean-Claude Tarby
Laboratoire CRISTAL
Université de Lille





Android : le futur ?

- Android OK, mais pas que... il y a aussi...
- IOS...
- Ionic, Flutter et d'autres frameworks cross-plateformes
 - <https://flutter.io/>
 - Flutter is Google's mobile app SDK for crafting high-quality native interfaces on iOS and Android in record time
 - <https://ionicframework.com>
 - <https://phonegap.com/>
 - <https://www.appcelerator.com/Titanium/>
 - ...
- Question :
 - appli mobile ? ou assistant mobile avec IA ?
 - La question se pose de + en +...



Android

- But : découvrir la programmation Android (Java/Kotlin)
- Aucune connaissance préalable sauf Java
- Quel est votre niveau Android ?
 - Cf. Google doc :
 - FI :
 - <https://docs.google.com/spreadsheets/d/1abZTqXwl9HS1BY0B38jtUgiHDdvZM1KMoO-BP5owaqk/edit?usp=sharing>
 - 50% débutant / 50% expert
 - Vous confirmez votre choix ? (tuto ou appli)
 - FA :
 - https://docs.google.com/spreadsheets/d/1BvV8cZ9vMiOdAUststL4hyymkUMw_ICWEhdqMhNfqvE/edit#gid=0
 - Quasi 100% débutant
 - Vous confirmez votre choix ? (tuto ou appli)



Android

- **Evaluation**
 - Un **tutoriel** à réaliser :
 - Choix des sujets dans quelques minutes
 - Explication juste après
 - ... ou pour ceux qui connaissent déjà bien Android...
 - Une **petite application** à réaliser au plus tard dans la semaine qui suit le dernier cours
 - FI : 28 novembre 20h
 - FA : 13 décembre 20h
 - Sujet de l'application



Vous connaissez déjà Android ?

- Qu'en pensez-vous ?
 - Facile, simple, puissant... OU difficile, compliqué... ?
 - Pourquoi cet avis ?
 - Que pensez-vous par exemple :
 - des « fragments »,
 - de la gestion des BDD Sqlite
 - des layouts
 - des MVP/MVC/MVVM...
 - des bibliothèques type ButterKnife, Retrofit... ?



Avis personnel

- Android est très puissant
 - On peut faire tout ce qu'on veut (y compris attaquer le noyau)
 - Fonctionne sur un vaste ensemble de périphériques
 - de la montre connectée à la TV connectée, en passant par la voiture connectée, etc.
- Mais...
 - Très long à apprendre
 - Parfois très complexe (tordu ?)
 - En perpétuel changement
 - Chaque année une nouvelle version au moins...
 - Outils encore trop basiques.



VisUML

- Outils encore trop basiques ?
 - Une solution avec VisUML
 - **Démo...** (dont filtres, tags, séquences...)
 - D'où les sujets de PFE sur VisUML ☺
 - Cf. vidéo à
<https://www.youtube.com/watch?v=buyGojmbUpQ>



Procédure d'affectation des tutoriels

- Je vous choisis → vous choisissez votre sujet parmi la liste visible à
https://docs.google.com/spreadsheets/d/1ctug8w6BgfQyQ_LuGBMI3k2nQTQuWEPhyr3zUsxA-rw/edit?usp=sharing
 - Mais vous pouvez aussi me proposer des sujets (je les valide ou non)
- Début des tutoriels :
 - le ~~16 octobre~~ 23 octobre (FI), 2h de TPs
 - le 15 novembre ? (FA), 4h de TPs



Le tutoriel

(appelé TP pour faire + simple)

- **Le TP est un mini-tutoriel sur un sujet précis**
 - Pensez que vous parlez à des novices en Android/Kotlin !
- Au plus tard 72 heures avant le tutoriel, vous devez :
 - Déposer vos fichiers sur un git (gitlab de la fac ou équivalent ➔ accès facile)
 - Code source initial et Code source final, le tout bien commenté !
 - Un "support" de quelques pages (en pdf), cf. modèle
 - Envoyer un mail à toute la promo + moi (jean-claude.tarby@univ-lille.fr) + interprète (contact@via-interprete.fr) pour que tout le monde aie le temps de regarder le contenu
- Le jour J, vous faites le TP à toute la promo (et moi :-))
 - Durée : de 45 à 60 minutes (cf. planning envoyé bientôt)
 - on comprend le sujet traité, à quoi ça sert, comment on crée et gère les choses (de façon simple).
 - **Conseil : faites au plus simple pour le TP, 45 mns = très court !**
 - **Exemple : data en dur pour RecyclerView...**



Le tutoriel (appelé TP pour faire + simple)

- N'hésitez pas à donner des pointeurs (dans le support par exemple) pour des compléments d'infos.
- Les TPs de l'an passé sont disponibles à
https://drive.google.com/open?id=1pODWpzn_4ZDVxWwBtoh0tH38hoLkHSQ0
 - Ce ne sont pas tous de bons exemples à suivre...



Un conseil

Le code doit pouvoir être copié-collé depuis le PDF, donc pas une simple image !)

```
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    return when (item.itemId) {
        R.id.action_settings -> true
        else -> super.onOptionsItemSelected(item)
    }
}
```

- Pas ça !

```
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    return when (item.itemId) {
        R.id.action_settings -> true
        else -> super.onOptionsItemSelected(item)
    }
}
```

```
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    return when (item.itemId) {
        R.id.action_settings -> true
        else -> super.onOptionsItemSelected(item)
    }
}
```



Android, Google, San Francisco...





Avant tout...

- On charge les téléphones pour pouvoir les utiliser tout à l'heure...
 - Demander des chargeurs si possible
 - Dorénavant pensez à aller chercher votre téléphone/tablette avant la séance !!!
- Laisser les téléphones éteints pour le moment
 - Charge + rapide
 - On testera d'abord sur l'émulateur ☺ ☹



Nouveautés depuis 4.4/L

- New Android Runtime (ART)
 - Avant Dalvik, compilation à la volée
 - Maintenant ART avec optimisation du garbage collector, compilation en code machine natif lors de l'installation, support de débogage
- Nouvelles notifications
- Nouveaux composants graphiques
 - RecyclerView (mieux que ListView), CardView (vs. Fragment ?), touch feedback animations,...
- Modifications autour du multimédia
- Modifications autour du réseau
- ...
- Détails à <http://developer.android.com/preview/api-overview.html>



Nouveautés avec les versions 5...

- Nouvelles *nouvelles* notifications
- Nouvelles permissions (gestion par demande à utilisateur)
- Nouveau design (look and feel)
- Nouveau SDK et API, bien sûr...

- Android Studio 1.3 est sorti
- Android 6 arrive...



Android 6

- Permissions en live, etc.
 - <https://github.com/googlesamples/android-RuntimePermissions>
 - <https://developer.android.com/preview/features/runtime-permissions.html>
- Autres nouveautés...
 - Géoloc
 - <http://developer.android.com/training/location/receive-location-updates.html>
 - <http://developer.android.com/training/location/index.html>
 - Floating Action Button, SnackBar, CoordinatorLayout...
 - Cf. <http://android-developers.blogspot.fr/2015/05/android-design-support-library.html>
 - ...
- Google APIs
 - <https://developers.google.com/android/>



Android 7, 8, 9...

- Allez voir par vous-mêmes ☺
 - <https://developer.android.com/about/versions/nougat/index.html>



Nouveauté pour Android 8 Oréo

- Pour installer depuis sources inconnues (en TP !), (source = <https://android-developers.googleblog.com/2017/08/making-it-safer-to-get-apps-on-android-o.html>)
 - To take advantage of this new behavior, developers of apps that require the ability to download and install other apps via the Package Installer may need to make some changes. If an app uses a targetSdkLevel of 26 or above and prompts the user to install other apps, the manifest file needs to include the `REQUEST_INSTALL_PACKAGES` permission:
 - `<uses-permission android:name="android.permission.REQUEST_INSTALL_PACKAGES" />` Apps that haven't declared this permission cannot install other apps, a handy security protection for apps that have no intention of doing so. You can choose to pre-emptively direct your users to the Install unknown apps permission screen using the `ACTION_MANAGE_UNKNOWN_APP_SOURCES` Intent action. You can also query the state of this permission using the PackageManager `canRequestPackageInstalls()` API.
 - Remember that `Play policies` still apply to apps distributed on Google Play if those apps can install and update other apps. In the majority of cases, such behavior is inappropriate; you should instead provide a `deep link` to the app's listing on the Play Store.
 - Be sure to check out the updated `publishing guide` that provides more information about installing unknown apps, and stay tuned for more posts on security hardening in Android O.



Android 9 Pie...

- Jetpack...
 - on en parlera bientôt

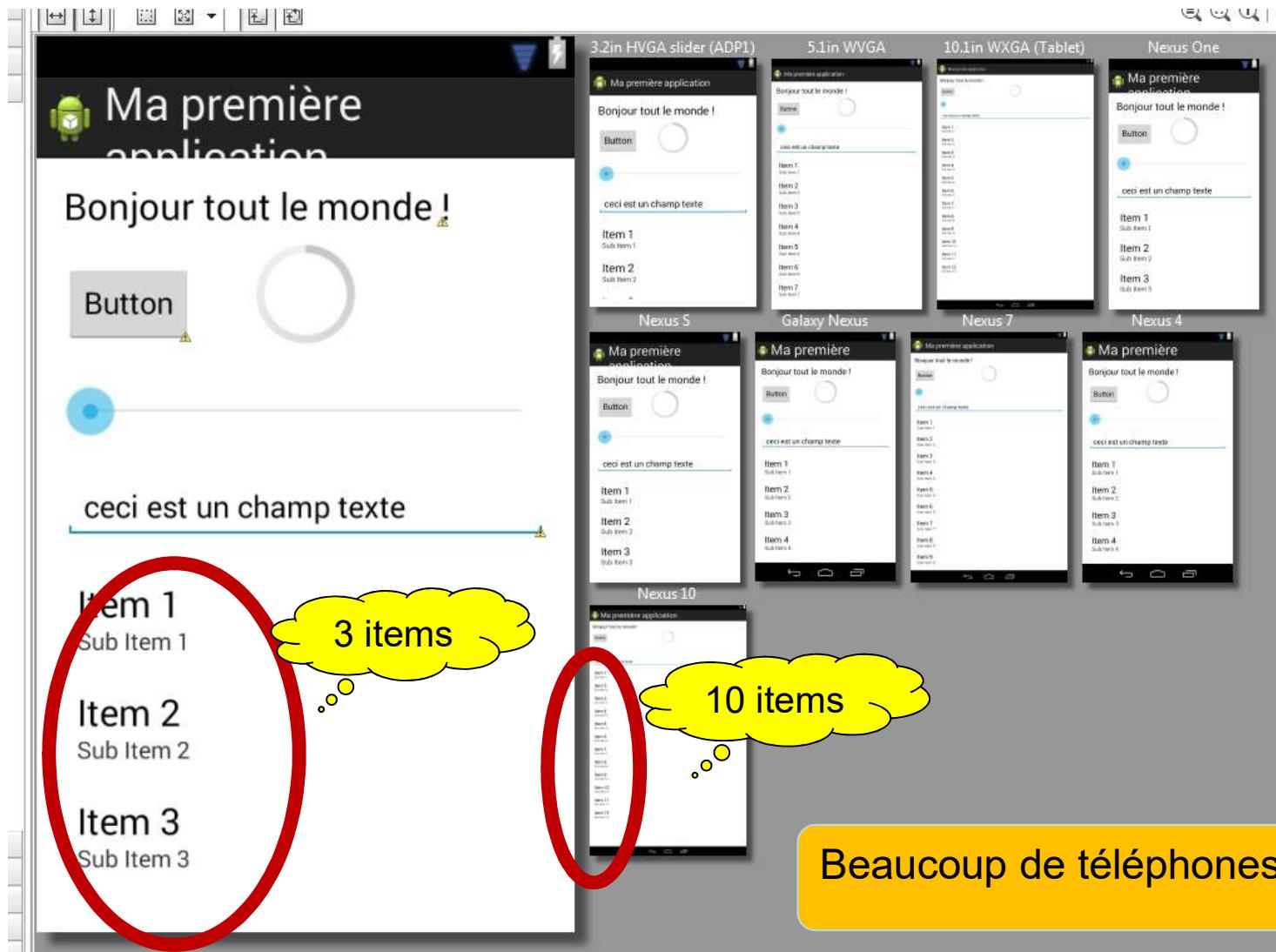




Ergonomie, UI Patterns, ...



Attention aux tailles des écrans !





Attention aux tailles des écrans !



Beaucoup de téléphones,
tablettes, ordinateurs...





Attention aux tailles des écrans !



Mais aussi...



Des montres et des bagues connectées





Des télévisions connectées





Des voitures connectées



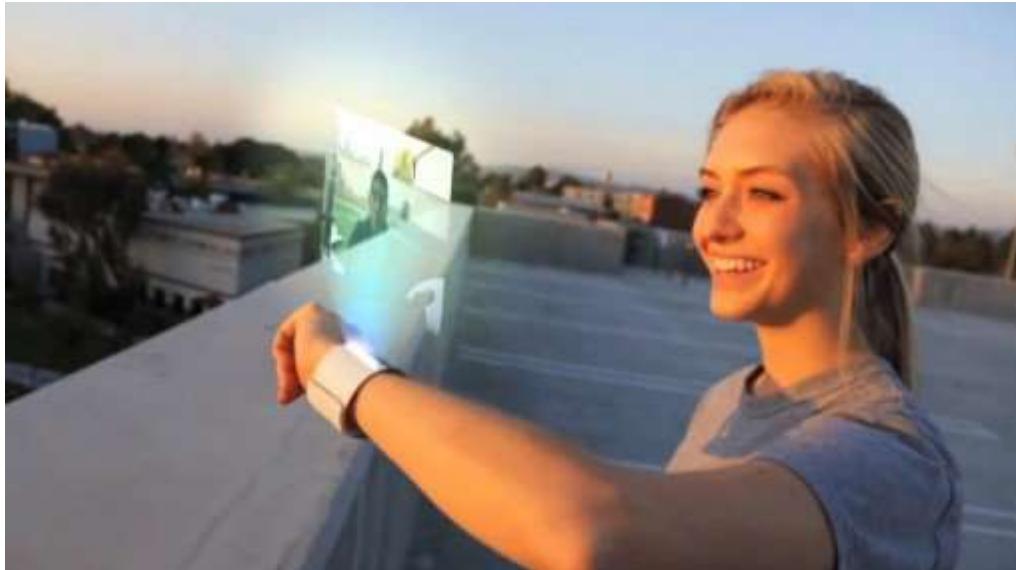


Des très grands écrans connectés

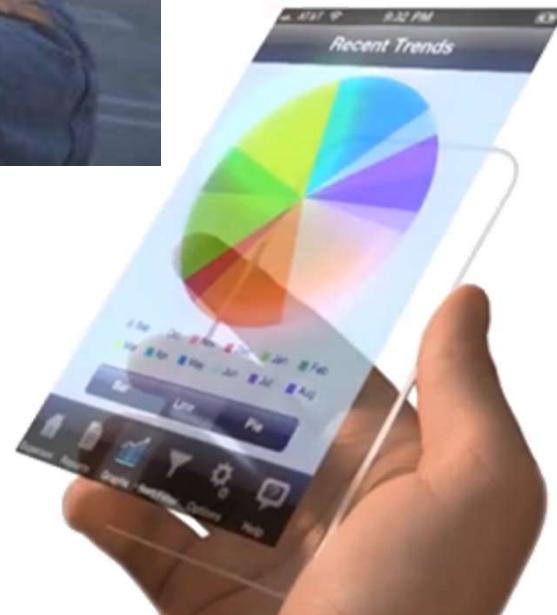




Des hologrammes connectés demain ?



Exemple à
<http://www.infohightech.com/un-casque-de-chantier-a-realite-augmentee/>





Des hologrammes connectés demain ?

Ca devient très compliqué !

de la « card » Google Glass à
900” !!!





Ressources Google pour le design !

- De + en + d'informations (good/bad news)
 - <https://design.google.com/> (site complet à part)
 - <http://developer.android.com/design/index.html>
- Des cours !
 - <http://developer.android.com/training/index.html>
- Material Design
 - <http://developer.android.com/design/material/index.html>
 - <http://www.google.com/design/spec/material-design/introduction.html>
- Icônes
 - <https://www.google.com/design/icons/index.html>
- Layouts
- Palettes de couleurs
- Polices de caractères
- ...
- <https://www.google.com/design/spec/resources>
- Cf. aussi <http://android-developers.blogspot.fr/2015/05/android-design-support-library.html>
- Voir aussi ce très bon site : <http://www.materialup.com/>



Design / Training / Samples ...

The screenshot shows a web browser displaying the Android developer samples page at developer.android.com/samples/index.html. A red circle highlights the top navigation bar, which includes links for Developers, Design, Develop (which is underlined in green), Distribute, Training, API Guides, Reference, Tools, Google Services, and Samples.

About the Samples

What's New

- Admin
- Background
- Connectivity
- Content
- Input
- Media
- Notification
- RenderScript
- Security
- Sensors
- System
- Testing
- UI

Samples

Welcome to code samples for Android developers. Here you can browse sample code and learn how to build different components for your applications. Use the categories on the left to browse the available samples.

Each sample is a fully functioning Android app. You can browse the resources, source files and see the overall project structure. You can copy and paste the code you need, and if you want to share a link to a specific line you can double-click it to get the URL.

Import Samples from GitHub

Android Studio provides easy access to import Android code samples from GitHub and is the recommended method to retrieve Android code samples.

To import a code sample into Android Studio:

- In the Android Studio menu, select **File > Import Sample** to open the Import Sample wizard.
- Select a sample to import and click **Next**.
- Specify the application name and project location if different from the displayed settings.
- Click **Finish**.

The sample project opens in a new Android Studio project.

Note: When starting Android Studio, you can also select **Import an Android code sample** in the Welcome to Android Studio wizard to import a sample project from GitHub as a new project.



Ressources Google (layout)

The screenshot shows a PDF document titled "Mobile Keylines and Spacing" from the "Material Design" guide. The document is displayed in Adobe Acrobat Pro, with the title page visible. The title page includes the text "Material Design" at the top, followed by "Mobile Keylines and Spacing" in large blue text, and "Last updated 23 October 2014" at the bottom. To the right of the title page, there is a sidebar with the following text and diagrams:

- "Use the layers palette to toggle keyline and spacing guides"
- A diagram showing a red square with a red border, labeled "Fixed increments determine widths of structural UI elements. For mobile we use a 56dp increment."
- A diagram showing a vertical red line with a red arrow pointing to the number "24", labeled "Vertical keylines determine placement of type and icons from edges of elements."
- A diagram showing a stack of three colored rectangles (green, blue, and pink) with a red arrow pointing to the blue rectangle, labeled "Vertical sizing determines the height of content blocks."
- A diagram showing two vertical red lines with a red arrow pointing to the gap between them, labeled "Horizontal margins determine spacing."



Qualité de vos applications

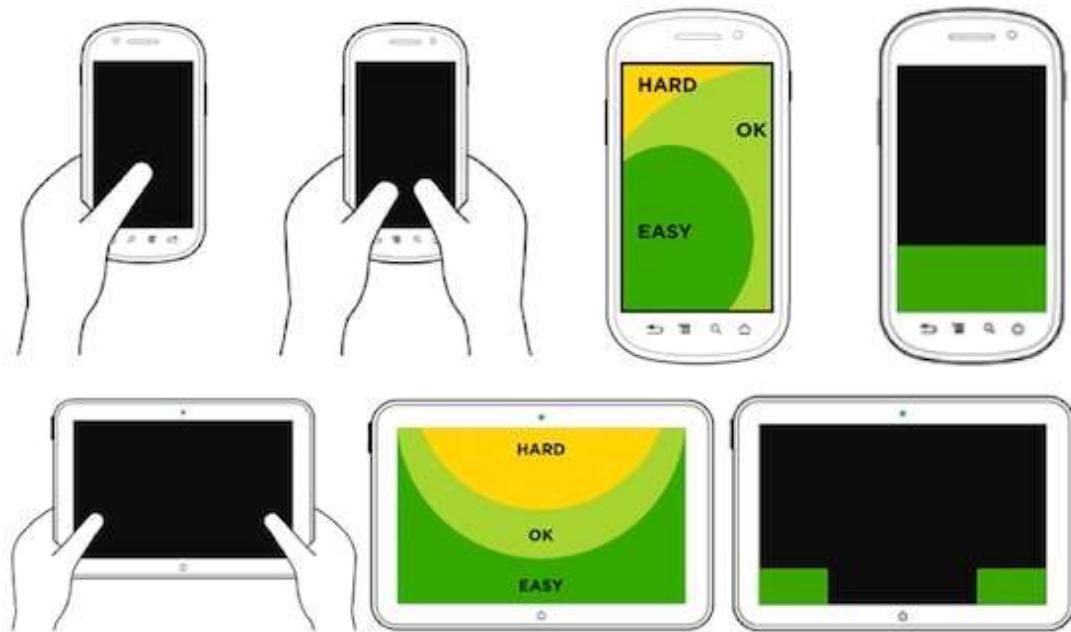
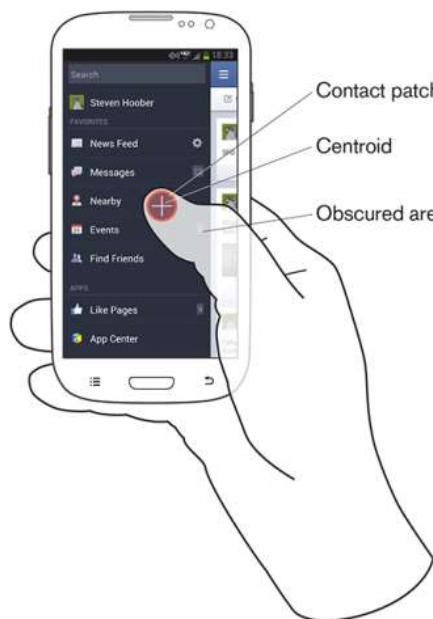
- De plus en plus de guidelines à respecter !
 - <http://developer.android.com/distribute/googleplay/quality/core.html>
 - À quand un outil pour le faire automatiquement ?
- Sources d'inspiration
 - <http://www.mobile-patterns.com/>
 - <http://www.androidpatterns.com/>
 - <http://androidpttrns.com/>
 - <http://pttrns.com/>
 - <http://www.android-app-patterns.com/>
 - <http://mycolorscreen.com/>





Quelques articles sur l'IHM mobile

- <http://www.simpleweb.fr/tag/tablette/> et <http://www.simpleweb.fr/> (mine d'infos)
 - <http://www.simpleweb.fr/2012/04/24/pour-bien-demarrer-dans-la-conception-dinterfaces-mobiles/>
 - <http://www.simpleweb.fr/2011/06/27/des-bibliotheques-de-composants-dinterfaces-mobiles/>
- <http://uxmag.com/articles/framework-for-designing-for-multiple-devices>





Aperçu de ce qu'on pourrait voir... ...si on avait 200h !

- Installation de l'environnement + 1ère application automatique
- Activités (StartActivity, Intents)
- Permissions
- IHM
 - layouts
 - widgets de base
 - Listview et Adapters
 - Spinner
 - Boite de dialogue...
 - Styles
 - Menus et ActionBar
- Data
 - BDD SQLite
 - Persistance des données (sauf SQLite)
 - Passage de données entre applications (hors fournisseurs de contenu/content provider)
 - préférences d'une application (sharedPreferences)
 - Stockage de texte simple dans un fichier
 - Fournisseur de contenus
- QRCode (lire et générer)
- Préférences (utilisateur et téléphone)
- C2DM (cloud to device messaging);
- Capteurs
 - GPS
 - Géoloc
 - Calcul itinéraire
 - Accéléromètre
 - Réseau
 - 3G, Wifi, Bluetooth...
 - Autres capteurs...
- AsyncTask
- Service
- Notification
- Threads
- Filtres
- Notification
- Appel/réception d'appel et de SMS
- Multimédia
 - son
 - vidéo
 - synthèse vocale
 - text-to-speech
- JSON et XML (parsing)
- Web services, connexion PHP /MySQL
- Publication sur le market + stats
- ...et encore plein d'autres choses...



Sommaire général

- Ce qu'on verra en 12 h de cours/TD/TP
 - Découverte de l'environnement Android Studio
 - Architecture générale et fonctionnement d'une application
 - Layout
 - Activité, Fragment
 - Ressources...
 - Mise en ligne dans Google Play
 - Découverte des fragments, de Room, de MVVM...
 - Et « le reste » avec 12 h pour vos mini-tutoriels



Quelques outils pour maquetter...

- Indigo Studio :
 - description succincte à <http://www.developpez.com/actu/51009/Infragistics-lance-Indigo-Studio-l-outil-gratuit-pour-la-conception-de-prototypes-d-IU-interactifs-pour-applications-Web-Desktop-et-mobile/>
 - outil dispo à <http://www.infragistics.com/products/indigo-studio>.
 - V2 payante (\$400 !), mais 30 jours gratuits
 - V1 gratuite : me contacter si nécessaire.
- Balsamiq (30 jours gratuits) : <http://balsamiq.com>
- Pencil (gratuit) : <http://pencil.evolus.vn/>
- un super petit outil pour montrer vos maquettes en contexte : <http://placeit.breezi.com/>
 - devenu payant ! ☺ ... mais « print screen » possible ☺
 - Quelques images encore gratuites
- <https://code.google.com/p/android-ui-utils/> et <http://www.journaldugeek.com/2011/11/28/android-design-preview-consultez-vos-maquettes-mobile/> : pour connecter Photoshop à vos mobiles
- <http://wireframesketcher.com/>
- <http://www.justinmind.com/prototyper/free-edition> <http://www.justinmind.com/> (a priori Très Bien)
- <http://www.build.me/> (a priori Très Bien)
- <http://fluidui.com/android/> (**adopté en 2013-2014 par vos prédecesseurs en e-services**)
- <http://www.invisionapp.com/> : gratuit pour un projet
- <http://www.justinmind.com/> : gratuit 30 jours
- <http://www.axure.com/>



(Très) Bonnes références (tutoriels, vidéos, forums)

- <http://developer.android.com/index.html>: LA référence
- <http://developer.android.com/training/index.html> : tutoriaux (mais des erreurs ! et pas toujours à jour !)
- <http://developer.android.com/guide/practices/index.html> : « best practices » officielles
- <http://www.google.com/design/spec/material-design/introduction.html> : la référence pour le Design Android
- <http://www.siteduzero.com> ==> <http://fr.openclassrooms.com/>: TB tutoriaux pour Android et iPhone
- <http://www.vogella.com/articles/Android/article.html> et <http://www.vogella.com/android.html>
 - dont ActionBar et Navigation Drawer (menu glissant) : <http://www.vogella.com/articles/ActionBar/article.html>
- <http://stackoverflow.com/>: forum pour Android et iPhone
- <http://android.developpez.com/cours/>
- <http://www.tutos-android.com/>
- <http://forum.frandroid.com>
 - <http://forum.frandroid.com/topic/16808-serie-de-tutoriauxdevelopper-sous-android/>
- <http://android-coding.blogspot.fr> : tutoriels moins courants
- <http://codentrick.com/> : tutoriels Android, Web
- <http://www.mysamplecode.com> : tutoriels Android et iOS
- <http://blog.restomaniak.com/restomaniak-sur-votre-mobile/> : tutoriels
- <http://www.tutomobile.fr/> : tutoriels pour Android et iPhone
- <http://www.univ-orleans.fr/lifo/Members/Jean-Francois.Lalande/enseignement/android/presentation-android.html> : cours pour se faire une idée générale d'Android
- Vidéos
 - <https://egghead.io/>
 - <http://thenewboston.org>
 - <http://www.mybringback.com/tutorials/>



Des outils pour compléter le développement

- Pour générer les icônes et autres graphiques à la bonne taille
 - <https://romannurik.github.io/AndroidAssetStudio/>
- Développer en natif Android et IOS avec... Javascript !
 - <https://www.nativescript.org/>
 - Source : <http://i-programmer.info/news/167/8367.html>
 - <http://www.appcelerator.com/>
- Développer en natif Android et IOS avec C#
 - <http://xamarin.com/platform>
 - <https://xamarin.com/getting-started/android>
 - <http://developer.xamarin.com/guides/android/>
 - http://developer.xamarin.com/guides/android/getting_started/



Quelques outils utilisés par les E-Services en Platine ou en stage

- Robotium
 - <https://code.google.com/p/robotium/>
 - Robotium is an Android test automation framework that has full support for native and hybrid applications...
 - Hébergé maintenant à <https://bintray.com/robotium/generic/releases/view>
- Spoon (pas celui de l'INRIA)
 - <http://square.github.io/spoon/>
 - Distributing instrumentation tests to all your Androids
- Push Notification
 - <http://urbanairship.com/>
 - Payant mais avec une base gratuite qui suffit pour des tests en Platine
 - <https://parse.com>
 - Propose aussi d'autres solutions autour du mobile dont le push <https://parse.com/products/push>
- Butter Knife
 - <http://jakewharton.github.io/butterknife/>
 - View "injection" library for Android
 - Permet d'écrire moins de lignes de code et + lisibles
- Crashlytics
 - <https://firebase.google.com/docs/crashlytics>
- Picasso
 - <https://github.com/square/picasso>
 - Pour télécharger des images et gérer un cache sans pb
- AndroidAsync
 - <http://koush.com/AndroidAsync>
 - Pour faire de l'asynchrone sans pb
 - Voir sur le même site:
 - <http://koush.com/ion> : Android Asynchronous Networking and Image Loading
- CouchBase
 - Accès à BDD NoSQL externes très facilement et très rapidement (???)
 - A priori, stage → bcp + compliqué que ça!
 - <http://www.couchbase.com/fr>
- PouchDB :
 - BDD local et distante synchronisée
 - <https://pouchdb.com/>



Quelques outils utilisés par les E-Services en Platine ou en stage

- Asciidoctor
 - <http://asciidoctor.org/>
 - Pour générer des docs toujours à jours, et très jolies (et incluant des diagrammes)
- Swagger
 - <http://swagger.io/>
 - permet de spécifier, visualiser, et consommer des web-services REST
 - permet à un développeur frontend de s'abstraire d'une connaissance précise du web-service
- JSON-Generator
 - <http://www.json-generator.com/>
 - générer des données au format JSON
 - très bien pour travail séparé entre équipe Front et équipe Back
- EventBus
 - <https://github.com/greenrobot/EventBus/blob/master/README.md>
- API REST sécurisé simplement
 - <https://jwt.io/>
- FireBase
 - Beaucoup utilisé en 2016-2017, 2017-2018, de + en + utilise !
- Glide
- Retrofit 2
- Dagger...

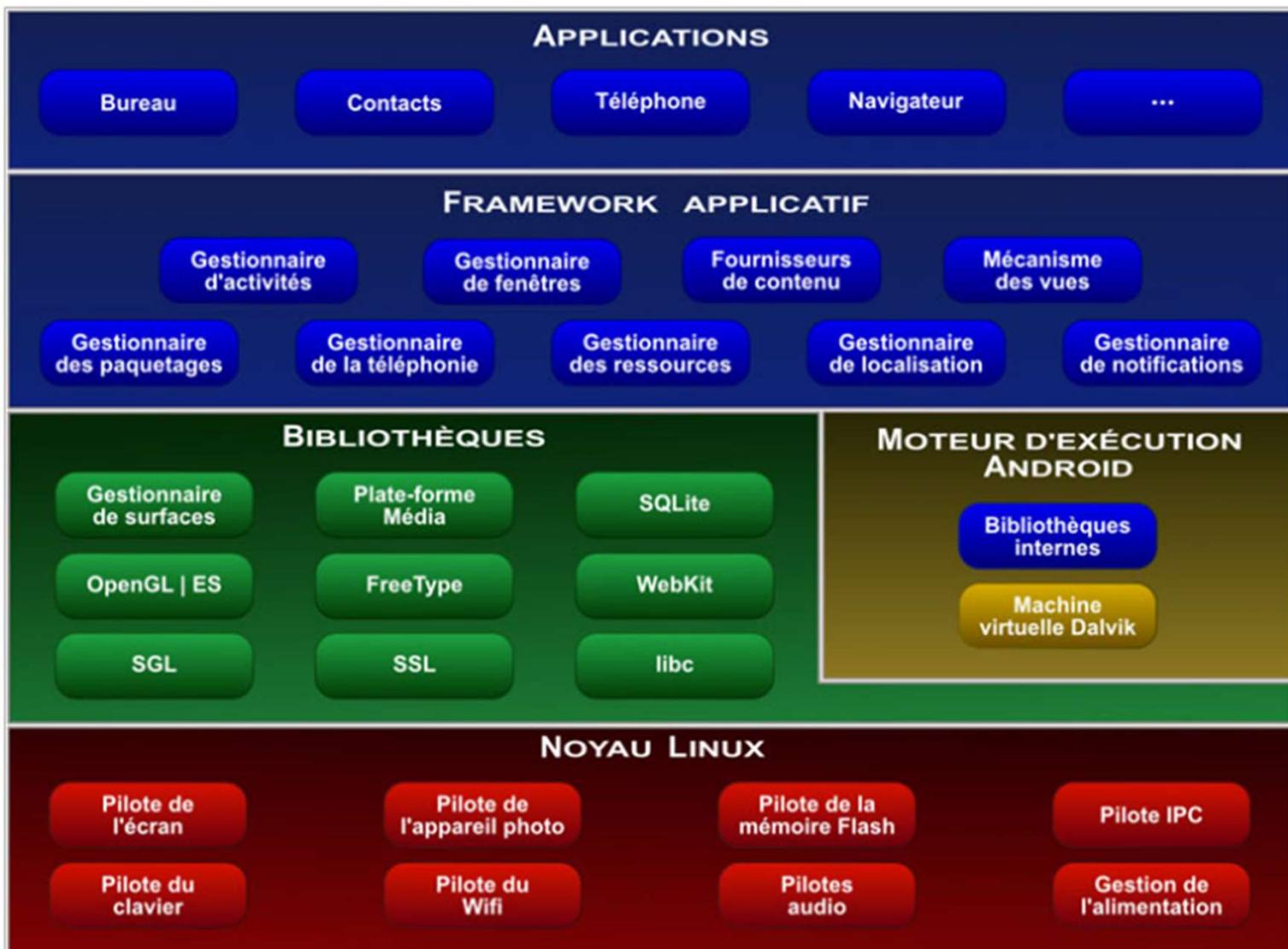


Préambule

- Android est né en septembre 2008 (V 1.0). Aujourd’hui V 9.
 - Comparatif TB fait des versions :
<http://socialcompare.com/fr/comparison/android-versions-comparison>
- Sachez que les SDK Android contiennent beaucoup d’exemples de code (dans « samples »), pensez à les consulter).
 - Vous pouvez aussi les utiliser directement dans Android Studio comme support de nouveaux projets
 - Voir aussi <https://developer.android.com/samples/>



Préambule





Pause « clavier »

- Je sais que les doigts vous brûlent, alors on va faire une pause clavier avec un « hello world »
 - Permettra de voir si votre IDE est OK



Introduction rapide à Android

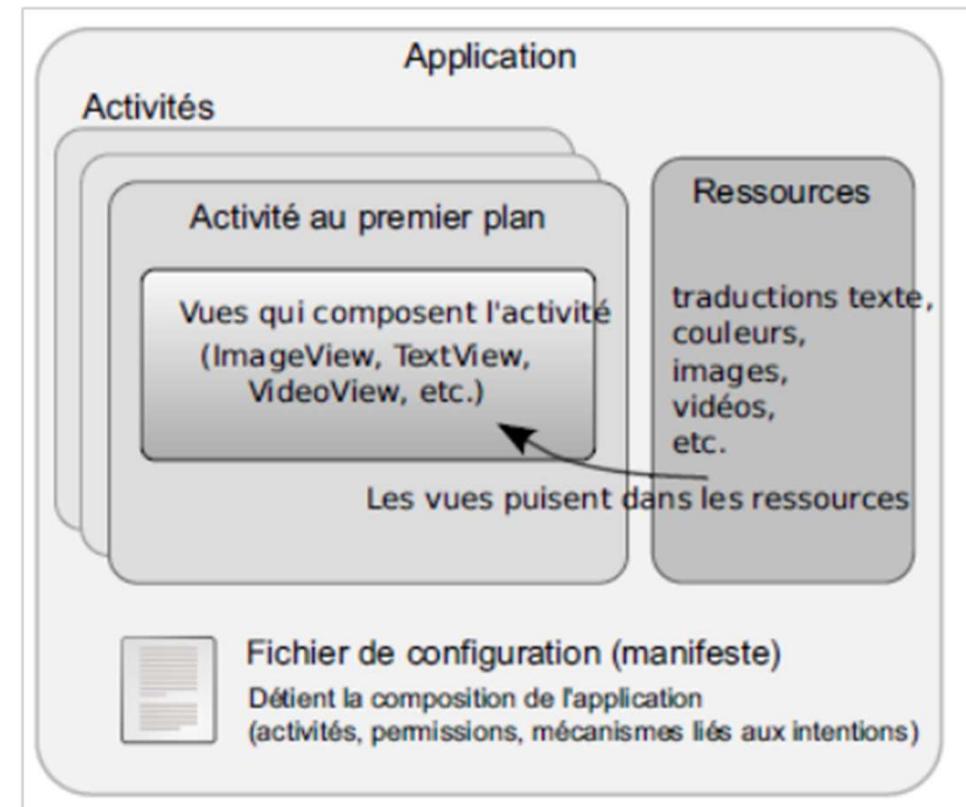
Quelques concepts de base

- Une application est décomposée en *activités* (et *fragments*).
- Une activité peut appeler d'autres activités (de l'application ou d'autres applications).
 - Si Activité 1 appelle Activité 2, on empile A2 sur A1
 - Android gère la mémoire et ces piles d'activités en conséquence
 - Bouton « back » = dépiler la dernière activité
- Les appels d'activités se font par des *Intents*
 - Intent, Activité, Vue
 - « qui sait faire X ? », « qui sait faire Y ? »
 - « métaphore de personnes avec habits »



Quelques concepts de base

- Activité
- Vues et contrôles (+ leur mise en page) ;
- Ressources
- le fichier Manifest





Activité

- Une Activité
 - Est un écran contenant des vues et des contrôles composant l'IHM de façon logique
 - Elle contient une hiérarchie de vues et de sous-vues.
 - Exemple : un formulaire d'ajout de contacts.
 - Est décomposée en :
 - logique + gestion du cycle de vie dans le **code java** de sa classe (héritant de Activity)
 - Il existe des classes d'activité qui facilitent (parfois) le travail, par exemple ListActivity
 - **IHM** : soit définie dans le code Java ci-dessus soit dans un fichier XML
- En général, si une application a « N » écrans, elle aura « N » d'activités.
 - Ce n'est plus vraiment le cas avec les Fragments...



Vues et Contrôles

- **Vues**
 - Ce sont les parties composant l'IHM
 - Elles contiennent des composants organisés suivant des mises en page (layout)
- **Contrôles** = widgets (boutons, checkbox...)
 - sous-ensemble des vues
 - accèdent aux textes et aux images qu'ils affichent en puisant dans les fichiers « ressources » de l'application



Ressources

- Stockées dans le dossier « res ».
- Contiennent les textes, images, sons, ... mais aussi les layouts, etc.

Type de ressource	Répertoire associé	Description
Valeurs simples	res/values	Fichiers XML convertis en différents types de ressources. Ce répertoire contient des fichiers dont le nom reflète le type de ressources contenues : 1. <code>arrays.xml</code> définit des tableaux ; 2. <code>string.xml</code> définit des chaînes de caractères ; 3. ...
Drawables	res/drawable	Fichiers <code>.png</code> , <code>.jpeg</code> qui sont convertis en bitmap ou <code>.9.png</code> qui sont convertis en "9-patches" c'est-à-dire en images ajustables. Note : à la construction, ces images peuvent être optimisées automatiquement. Si vous projetez de lire une image bit à bit pour réaliser des opérations dessus, placez-la plutôt dans les ressources brutes.
Layouts	res/layout	Fichiers XML convertis en mises en page d'écrans (ou de parties d'écrans), que l'on appelle aussi gabarits.
Animations	res/anim	Fichiers XML convertis en objets animation.
Ressources XML	res/xml	Fichiers XML qui peuvent être lus et convertis à l'exécution par la méthode <code>resources.getXML</code> .
Ressources brutes	res/raw	Fichiers à ajouter directement à l'application compressée créée. Ils ne seront pas convertis.



Multi-lingue...facile !

- res/values
 - strings.xml
- res/values-**fr**
 - strings.xml
- res/values-**es**
 - strings.xml



Mode portrait et paysage...facile !

- res/layout
- res/layout-land



Disposition sur l'écran...facile !(?)

- res/layout
- res/layout-large
- res/layout-xlarge



Disposition sur l'écran...facile !(?)

- res/layout
- res/layout-large
- res/layout-xlarge

« deprecated » depuis Android 3.2.

Utilisez plutôt les « largeurs » comme ci-après.



Résolution et Style... facile ?

- res/values
 - dimens.xml
 - styles.xml
- res/values-v11
- res/values-v14
- res/values-w820dp
- res/values-sw820dp
- ...
- Solution : les fragments (?)



Ressources

- Toutes ces ressources seront placées dans l'APK.
- Android crée une **classe statique R** qui sera utilisée pour accéder aux ressources depuis le code.
 - Dans « /gen » ou « /app/build »

```
package com.eyrolles.android.exemples;
public final class R {
    public static final class string { ①
        public static final int bienvenue-0x7f040000; ②
        public static final int texte_bouton_quitter-0x7f040001;
        public static final int texte_titre_ecran-0x7f040002;
    };
    public static final class Layout {
        public static final int ecran_de_demarrage-0x7f030001;
        public static final int ecran_principal-0x7f030000;
    };
    public static final class drawable {
        public static final int image_android-0x7f020000;
    };
}
```



Fichier Manifest

- Fichier indispensable à chaque application qui décrit entre autres :
 - le *point d'entrée* de votre application (quel code doit être exécuté au démarrage de l'application) ;
 - quels *composants* constituent ce programme ;
 - les *permissions* nécessaires à l'exécution du programme (accès à Internet, accès à l'appareil photo...).
 - dans les dernières versions, les « tools: »
 - <http://tools.android.com/tech-docs/tools-attributes>
 - Exemple : tools:listItem="@+id/simple_list_item_2" />



Fichier Manifest

- Une application « complexe » Android peut faire appel à :
 - des **activités** (composant applicatif)
 - des **services** (composant applicatif)
 - des **fournisseurs de contenus** (composant applicatif), par exemple pour accéder à des données en BDD
 - des **gadgets** (composant applicatif)
 - des **Intents**
 - des **Récepteurs d'Intents**
 - des **Notifications**.



Intents

- Les objets **Intents** permettent de diffuser des messages demandant la réalisation d'une action → permettent de fournir ou de demander des services. C'est Android qui « choisit » QUI fera l'action demandée.
- **Récepteurs d'Intents** : permettent à l'application d'être à l'écoute pour répondre aux objets Intents qui lui demanderaient quelque chose.
- **Notification** : signale une information à l'utilisateur sans interrompre ses actions en cours.
- **Filtres d'Intents** : un objet Intent peut mentionner explicitement un composant cible. Dans le cas contraire, Android choisit le meilleur composant grâce aux filtres (cf. <intent-filter> du fichier de config Manifest).



Permissions

- Pour accéder à certaines fonctionnalités, on doit déclarer des permissions dans le Manifest
 - par exemple pour envoyer des SMS, accéder au compte Google, etc.
- Permet à l'utilisateur d'en avoir conscience et de choisir.
- Permet aussi au Play Store de filtrer les applications pour ne proposer que celles qui peuvent fonctionner sur le périphérique connecté.
 - Peut être contourné en utilisant intelligemment `<uses-permission>` et `<uses-features android:required = "false">`
 - À vous de faire attention dans votre code !



Permissions

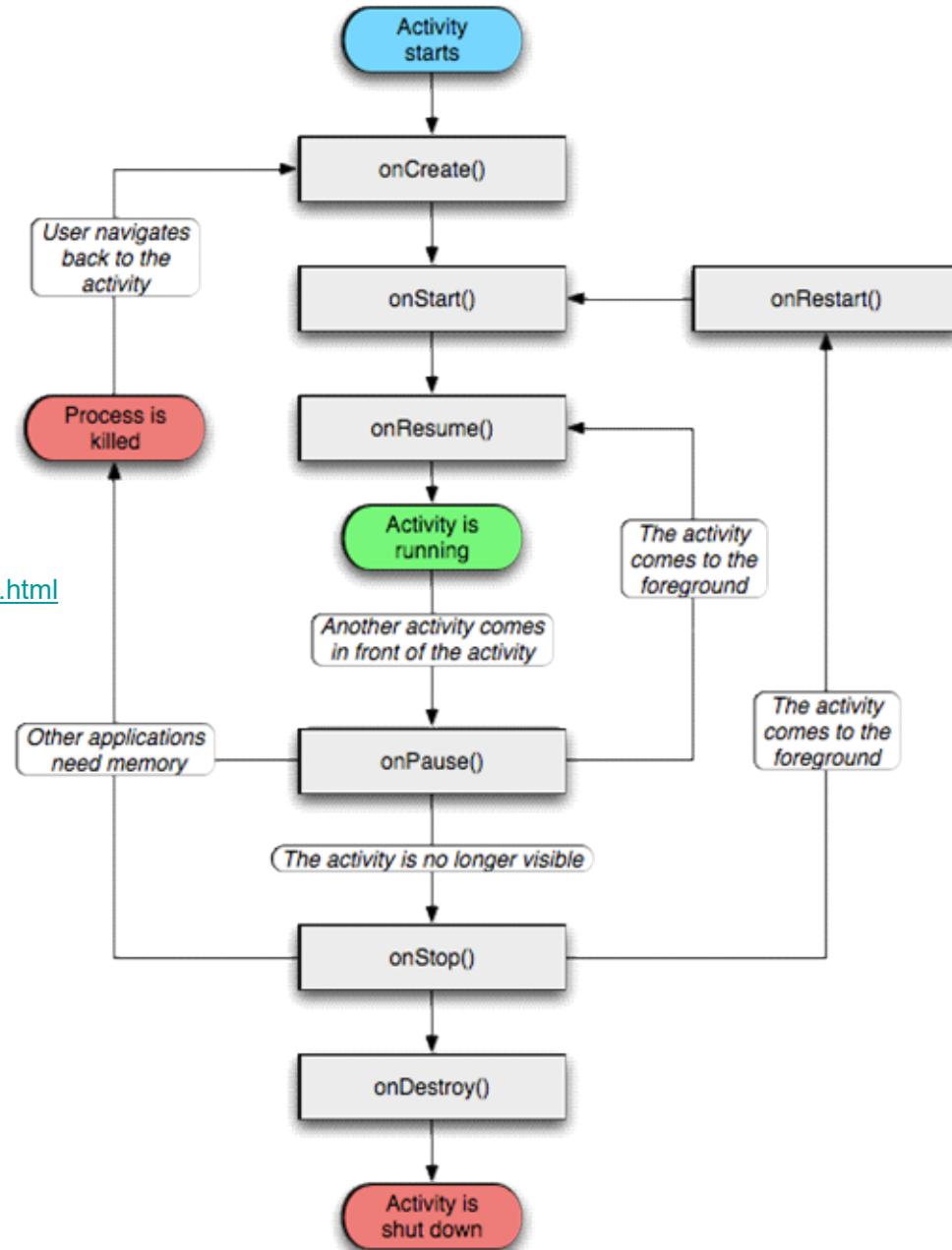
- Pour accéder à certaines fonctionnalités, on doit déclarer des permissions dans le Manifest
 - par exemple pour envoyer des SMS, accéder au compte Google, etc.
- Permet à l'utilisateur d'avoir conscience et de choisir.
- Permet aussi... Attention aux changements avec Android 6+... !
pour ne prop... Les permissions se font maintenant « à la volée » !
le périphérique
- Peut être contour... et... à... iligent `<uses-permission>` et `<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" android:required="false">`
 - À vous de faire attention dans votre code !



Cycle de vie d'une application

source : <http://developer.android.com/reference/android/app/Activity.html>

On verra un exemple plus tard...





Outils de développement (1/2)

- **ADT / Eclipse + plugin Android**
 - <http://developer.android.com/sdk/adt.html>
 - Bug de sécurité
 - Importer un projet
 - Détails : <http://developer.android.com/tools/support-library/setup.html>
 - » Détails : <http://developer.android.com/tools/support-library/features.html>
- **Android Studio**
 - Basé sur IntelliJ IDEA
 - <http://developer.android.com/sdk/installing/studio.html>

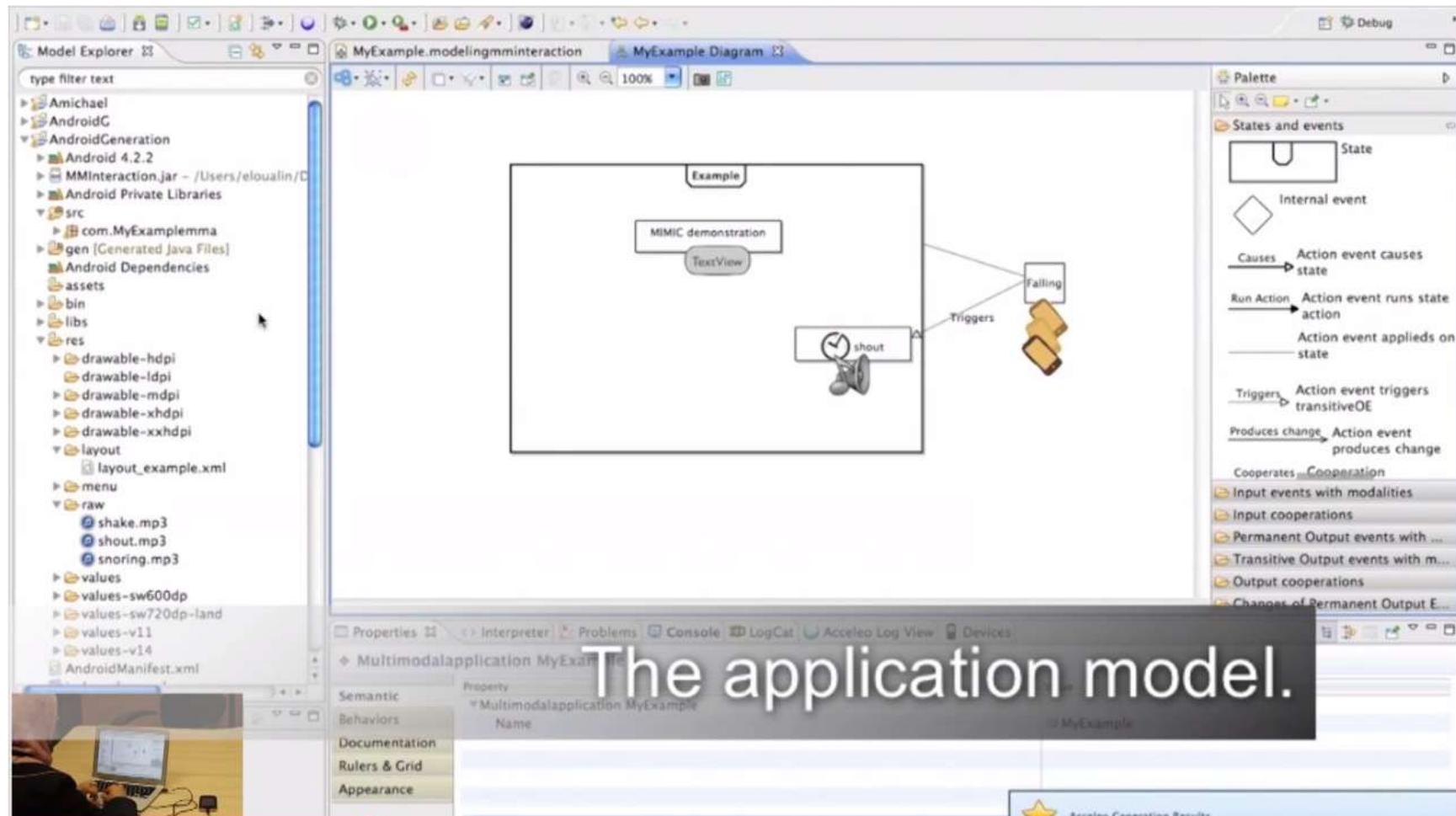


Outils de développement (2/2)

- **AppInventor** (MIT)
 - Basé sur Scratch du MIT : <http://scratch.mit.edu/>
 - <http://appinventor.mit.edu>
- Notepad++ et autres éditeurs + compilateur en ligne de commande...(si vous êtes courageux !)



Futurs outils ?



<http://youtu.be/g0Ekioe1CQ0?list=UUofGGFqq9t1kfKOAzn5HJGQ>



Android Studio

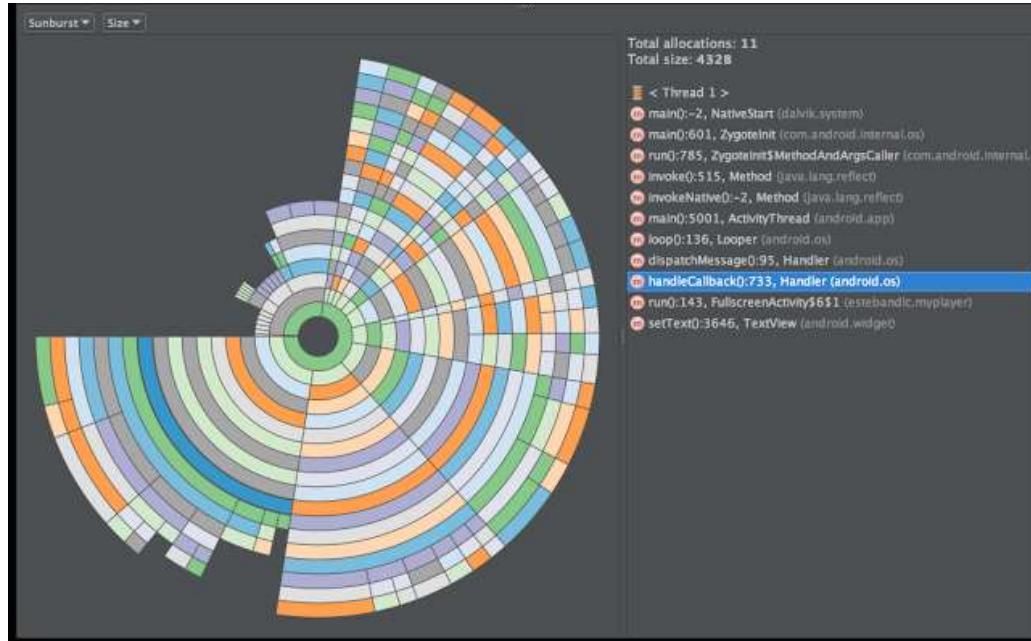
The screenshot shows the Android Studio interface for a project named "MyApplication".

- Project Structure:** Shows the project tree with files like `build.gradle`, `build.xml`, `gradlew`, and `gradlew.bat`.
- Code Editor:** Displays `MainActivity.java` and the XML layout file `activity_main.xml`. The layout file contains a single `TextView` with the text "Hello world!".
- Design View:** Shows the visual representation of the application's main screen, titled "My Application", with the text "Hello world!" displayed.
- Properties:** A panel showing the properties of the selected component.
- Logcat:** A window displaying log messages from the device. The log includes entries related to battery service updates and debugger acceptance.

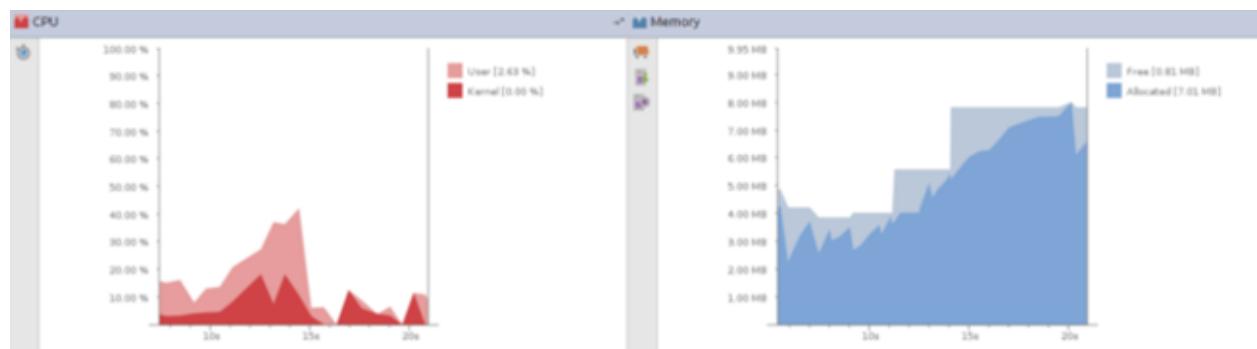
Plus d'infos à
<http://developer.android.com/tools/studio/index.html>



Android Studio : outil puissant... ?



CVS,
Git,
Mercurial...





Android Studio : outil puissant... ?

VCS

```
package fr.univ_lille.ieea.tarby.myfirstapplication;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;

public class MyActivity extends AppCompatActivity {

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

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar
        getMenuInflater().inflate(R.menu.menu_my, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so
        // as you specify a parent activity in AndroidManifest.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }
    }
}
```

Current

```
1 package fr.univ_lille.ieea.tarby.myfirstapplication;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.view.KeyEvent;
6 import android.view.Menu;
7 import android.view.MenuItem;
8
9 public class MyActivity extends AppCompatActivity {
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_my);
15     }
16
17     @Override
18     public boolean onKeyLongPress(int keyCode, KeyEvent event) {
19         return super.onKeyLongPress(keyCode, event);
20     }
21
22     @Override
23     public boolean onOptionsItemSelected(MenuItem item) {
24         // Handle action bar item clicks here. The action bar will
25         // automatically handle clicks on the Home/Up button, so
26         // as you specify a parent activity in AndroidManifest.
27         int id = item.getItemId();
28
29         //noinspection SimplifiableIfStatement
30         if (id == R.id.action_settings) {
31             return true;
32         }
33     }
}
```

3 differences Deleted Changed Inserted



Android Studio : outil puissant... ?

VCS

C:\Users\Jean-Claude\AndroidStudioProjects\MyFirstApplication\app\src\main\java\fr\univ_lille\ieea\tarby\myfirstapplication\MainActivity.java

```
07/09/2015 17:15 - MainActivity.java (Read-only)
package fr.univ_lille.ieea.tarby.myfirstapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;

public class MainActivity extends AppCompatActivity {

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

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is available.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }
    }
}
```

Current

1 package fr.univ_lille.ieea.tarby.myfirstapplication;
2 import android.support.v7.app.AppCompatActivity;
3 import android.os.Bundle;
4 import android.view.Menu;
5 import android.view.MenuItem;
6
7 public class MainActivity extends AppCompatActivity {
8
9 @Override
10 protected void onCreate(Bundle savedInstanceState) {
11 super.onCreate(savedInstanceState);
12 setContentView(R.layout.activity_main);
13 }
14
15 @Override
16 public boolean onCreateOptionsMenu(Menu menu) {
17 // Inflate the menu; this adds items to the action bar if it is available.
18 getMenuInflater().inflate(R.menu.menu_main, menu);
19 return true;
20 }
21
22 @Override
23 public boolean onOptionsItemSelected(MenuItem item) {
24 // Handle action bar item clicks here. The action bar will
25 // automatically handle clicks on the Home/Up button, so long
26 // as you specify a parent activity in AndroidManifest.xml.
27 int id = item.getItemId();
28
29 //noinspection SimplifiableIfStatement
30 if (id == R.id.action_settings) {
31 return true;
32 }
33 }
34}

Ignore whitespace: Do not ignore | Highlight: By word | Current | 1 file | 3 differences | Deleted | Changed | Inserted

Voir aussi l'inspecteur de code (Analyse) pour vos projets ☺



Android Studio : outil puissant... ?

- Sur la bonne voie en tous cas...
- Exemple :

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
    <EditText android:id="@+id/edit_message"
        android:layout_weight="1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="@string/edit_message" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/button_send" />
</LinearLayout>
```



Android Studio : outil puissant... ?

- Sur la bonne voie en tous cas...
- Exemple

The screenshot shows the Android Studio interface with the 'activity_my.xml' file open. A context menu is displayed over the XML code for an EditText element. The menu items are:

- Replace size attribute with 0dp
- Disable inspection
- Edit 'Inefficient layout weight' inspection settings
- SUPPRESS: Add tools:ignore="InefficientWeight" attribute
- Override Resource in Other Configuration...
- Set Namespace Prefix to Empty

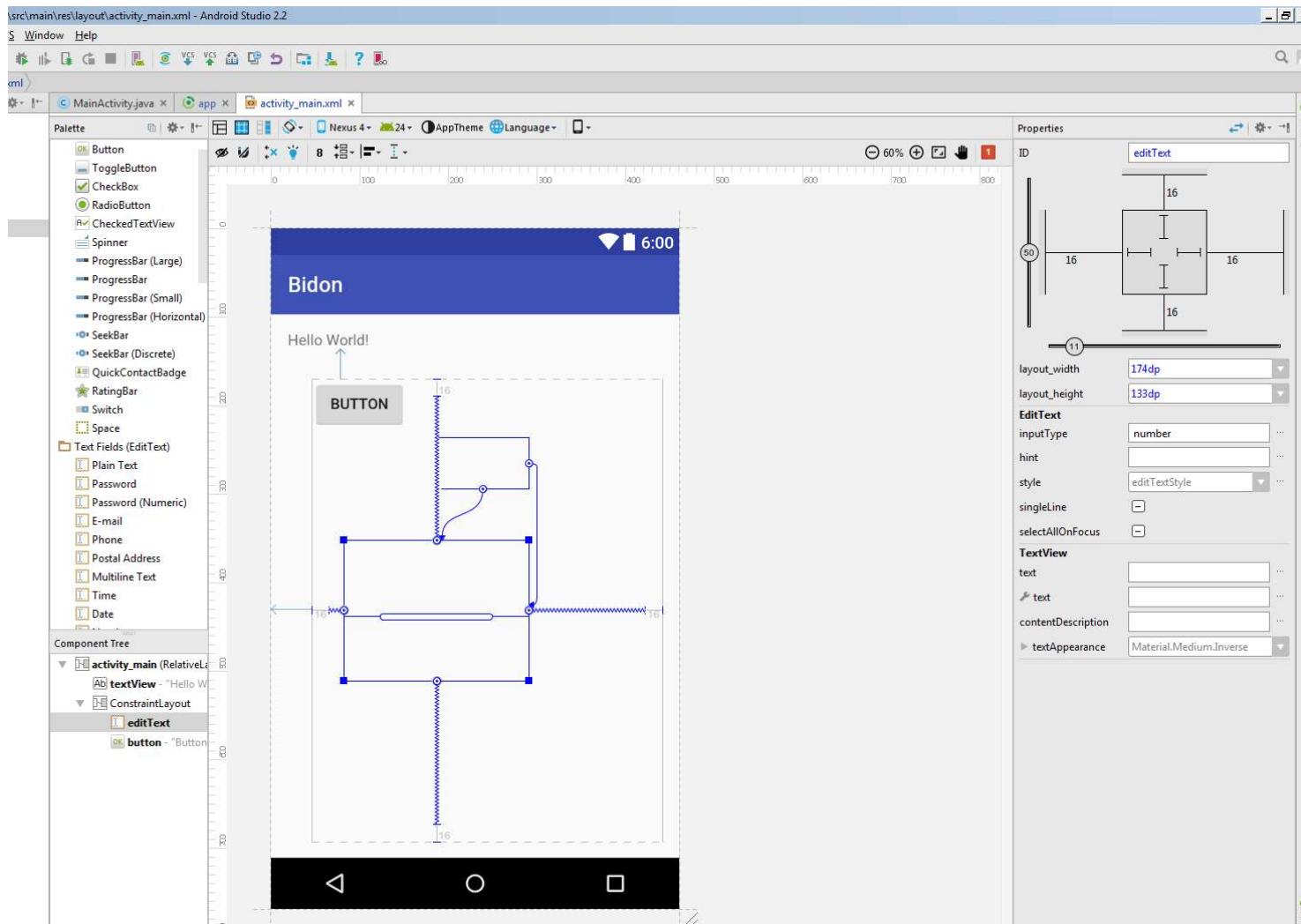


Android Studio : outil puissant... ?

- Emulateur toujours (très) long à démarrer (encore en 2018 ???)
- Solution : émulateur de **GenyMotion**
 - www.genymotion.com
 - (sous forme de plugin pour Android Studio)
- (Solution 2 : voir le téléphone en live à l'écran)
 - plugin **Vysor** sur chrome
 - <https://play.google.com/store/apps/details?id=com.koushikdutta.vysor&hl=fr>
 - et aussi AirDroid
 - <http://web.airdroid.com> et <https://play.google.com/store/apps/details?id=com.sand.airdroid&hl=fr>



Android Studio: toujours en évolution (les ConstraintLayouts...)





Comment bien configurer Android Studio

- <https://developer.android.com/studio/build/optimize-your-build.html>
 - dont comment utiliser Android Studio en mode offline, etc.

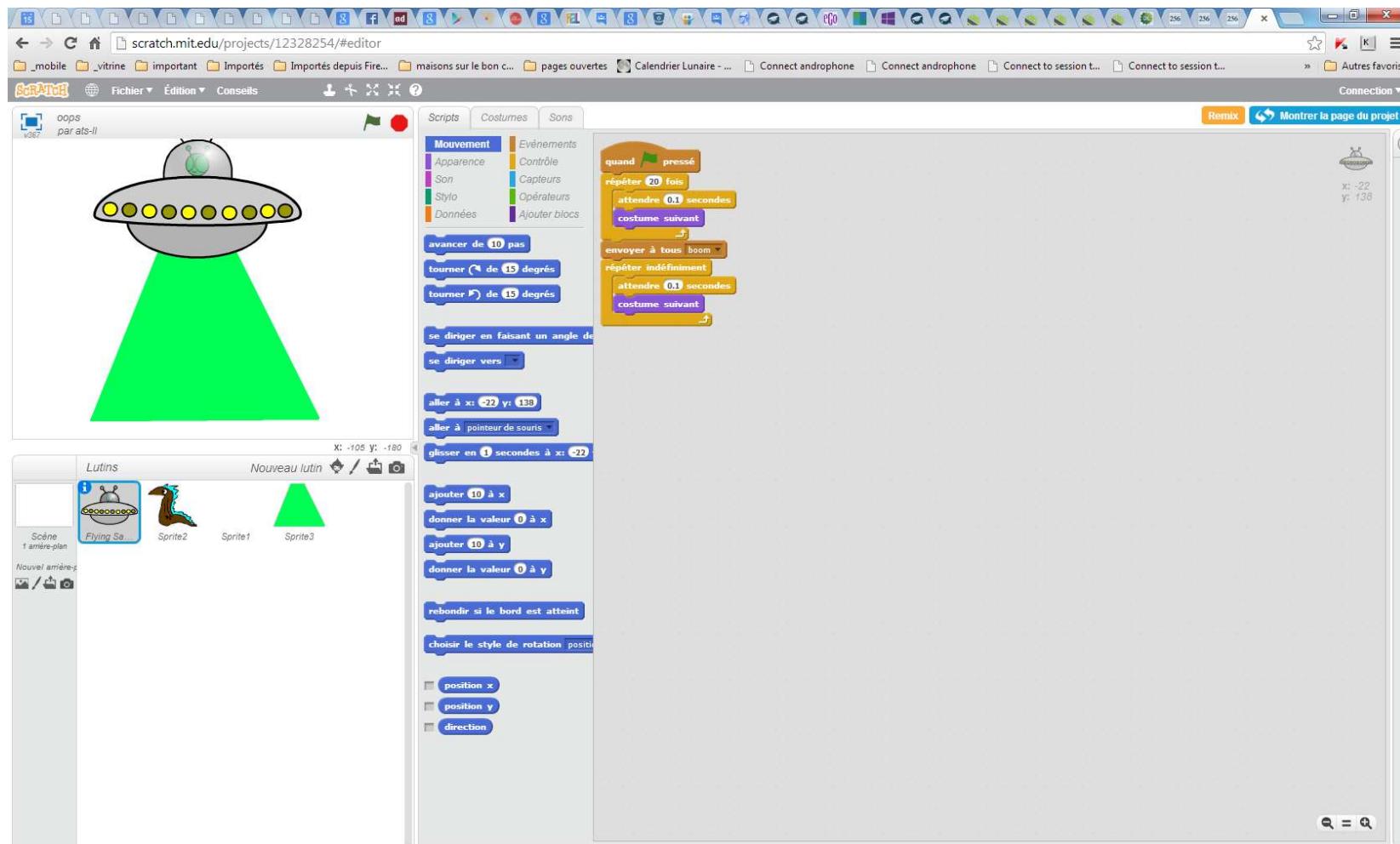


- Dépendances de Android Studio
 - <https://developer.android.com/studio/build/dependencies>
 - Avec les nouvelles directives ! :
https://developer.android.com/studio/build/dependencies?utm_source=android-studio#dependency_configurations

New configuration	Deprecated configuration
implementation	compile
api	compile
compileOnly	provided
runtimeOnly	apk
annotationProcessor	compile

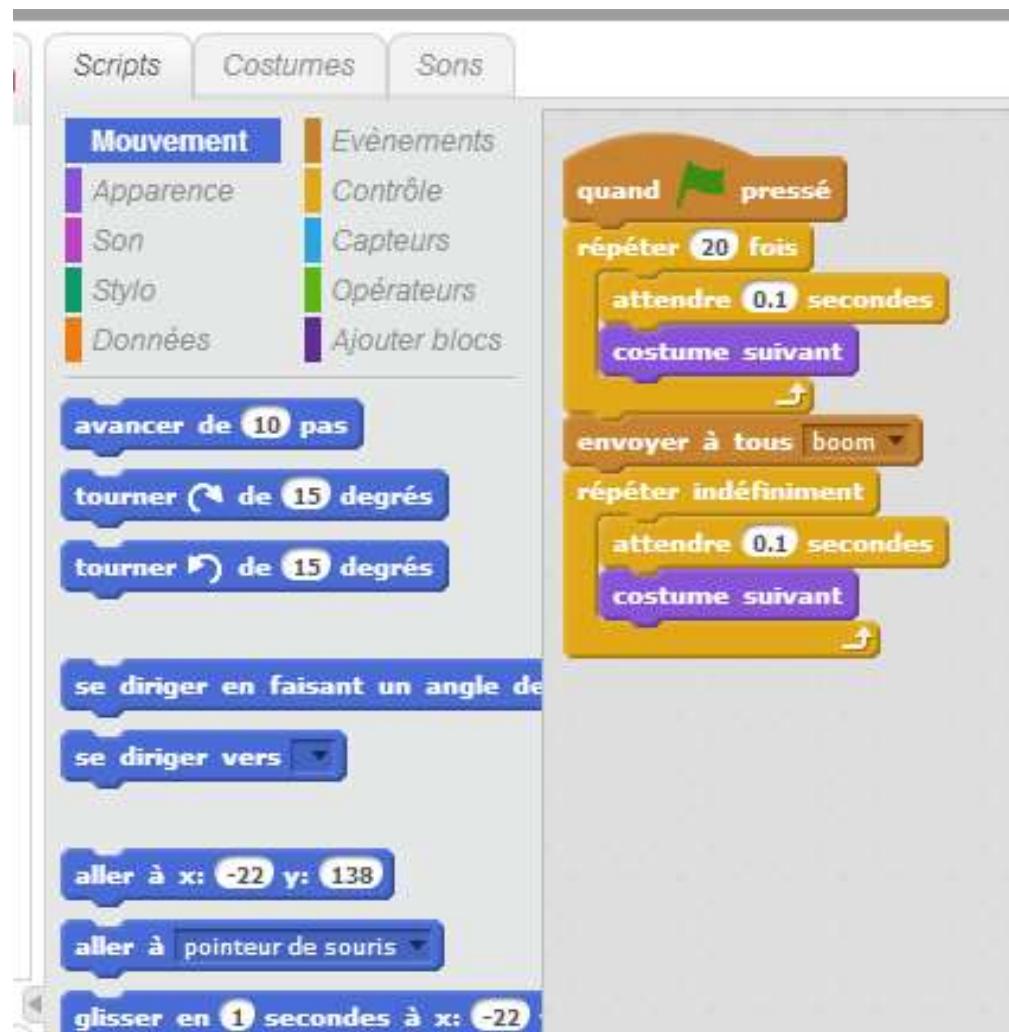


Scratch (à partir de 8 ans !)



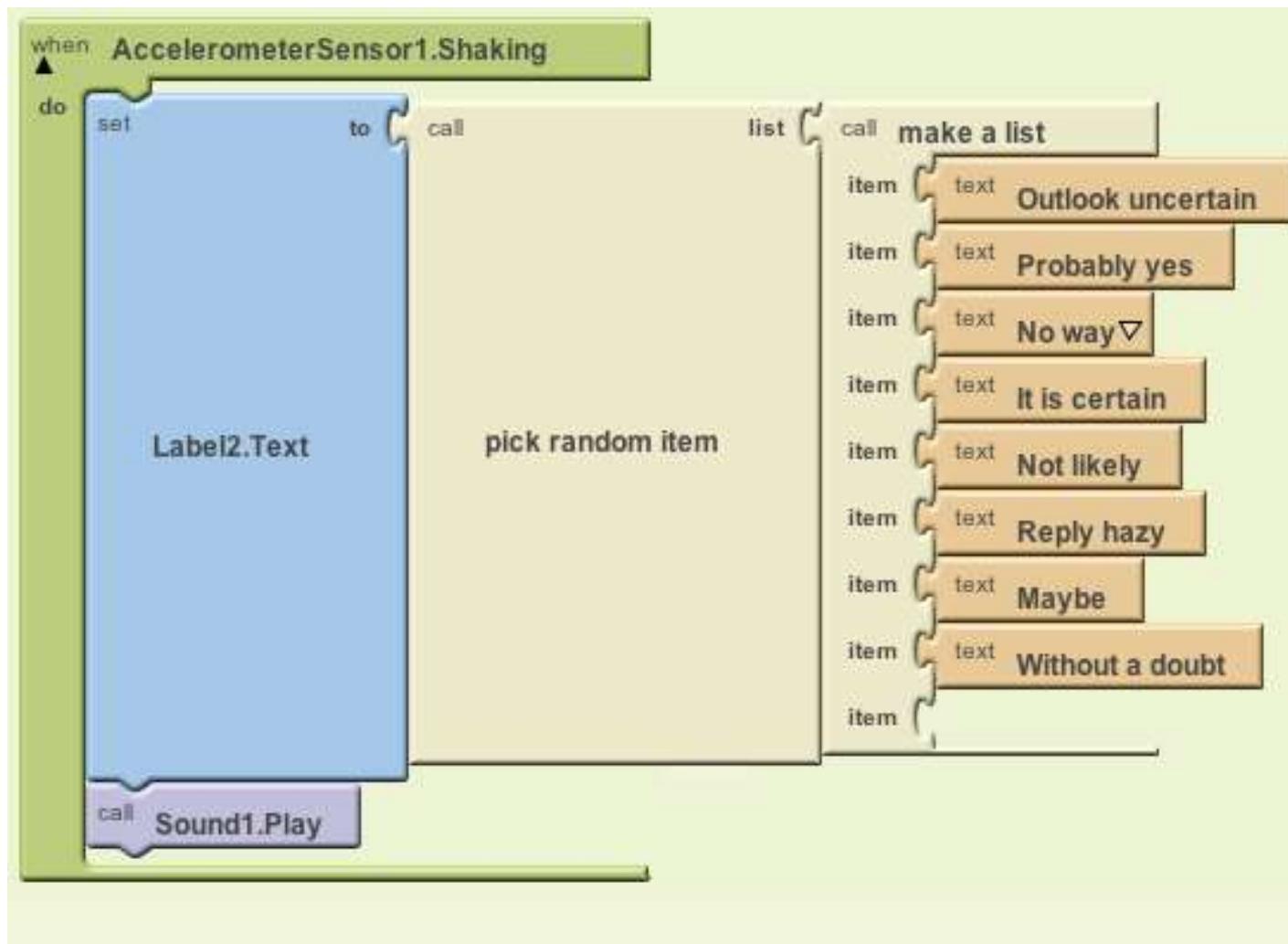


Scratch (à partir de 8 ans !)





App Inventor (à partir de 8 ans ???)





Page de référence

The screenshot shows the official Android developer website at <https://developer.android.com>. The top navigation bar includes links for Applications, Vitrine, important, pages ouvertes, suivre stages et alternance, absences alternants, Mon itinérant :: Food, Documentation technique, VisUML page de lanc, bibliothèque illiad, PEG.js visualisation g, and Autres favoris. The "Android Studio" link is circled in red. Below the navigation, there's a "Developers" section with links for Platform, Android Studio, Google Play, Android Jetpack, Docs, and Blog. A search bar is also present.

FEATURED

Introducing Android 9 Pie

Powered by AI to make your smartphone smarter, simpler and tailored to you.

[GET STARTED](#)



FEATURED

Google I/O 2018 videos

View all the videos from Google I/O 2018, introducing new platform features, tools, and deep-dives.



[WATCH](#)

FEATURED

Introducing Android Jetpack

Components, tools and architectural guidance to accelerate Android development, eliminate boilerplate code, and build high quality, robust apps.



[LEARN MORE](#)

<https://developer.android.com/studio/>



Quelques raccourcis...

- Go to class CTRL + N
- Go to file CTRL + Shift + N
- Navigate open tabs ALT + Left-Arrow; ALT + Right-Arrow
- Look up recent files CTRL + E
- Go to line CTRL + G
- Navigate to last edit location CTRL + SHIFT + BACKSPACE
- **Go to declaration CTRL + B**
 - ou CTRL clic
- **Go to implementation CTRL + ALT + B**
- Go to source F4
- Go to super Class CTRL + U
- Show Call hierarchy CTRL + ALT + H
- Search in path/project CTRL + SHIFT + F
- **Reformat code CTRL + ALT + L**
- **Optimize imports CTRL + ALT + O**
- Code Completion CTRL + SPACE
- Issue quick fix ALT + ENTER
- **Surround code block CTRL + ALT + T**
- Rename and Refactor Shift + F6
- Line Comment or Uncomment CTRL + /
- **Block Comment or Uncomment CTRL + SHIFT + /**
- Go to previous/next method ALT + UP/DOWN
- Show parameters for method CTRL + P
- Quick documentation lookup CTRL + Q
- Delete a line CTRL + Y
- View declaration in layout CTRL + B



Prêt !

- A présent, le poste est configuré pour créer une première application Android et la tester sur :
 1. l'émulateur
 2. le smartphone
- Avant de passer à la suite, allumez et réinitialisez les téléphones !
 - Puis autoriser « sources inconnues » (cf. rappels ci-après)
 - Et « Débogage USB » actif

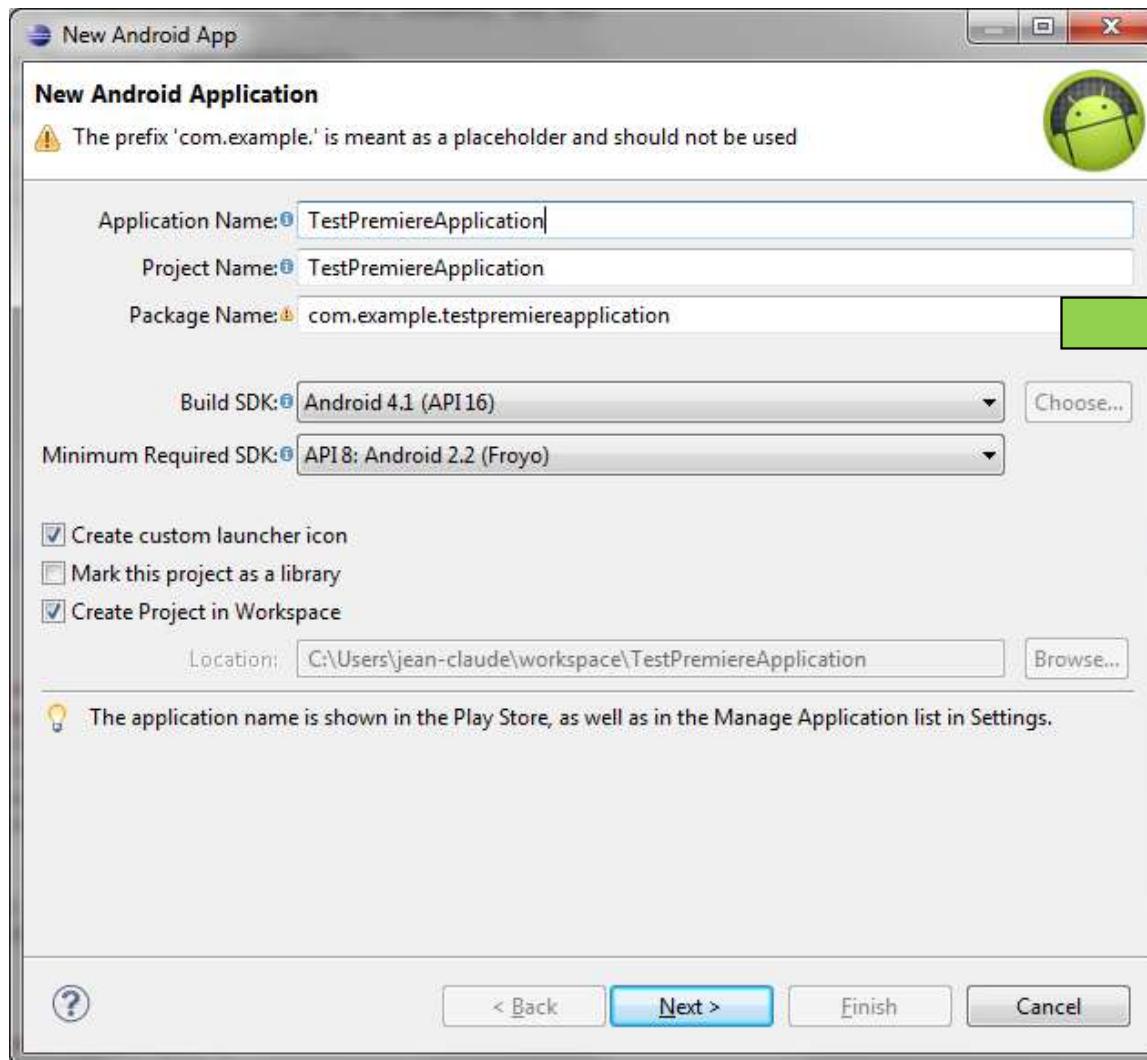


Pour activer le mode développeur !

- On Android 4.2 and newer, Developer options is hidden by default. To make it available, go to Settings > About phone and tap Build number (ou n° de version en français) seven times.
 - Affichage nouveaux paramètres généraux (options développeurs) → déboggage USB...



Première application

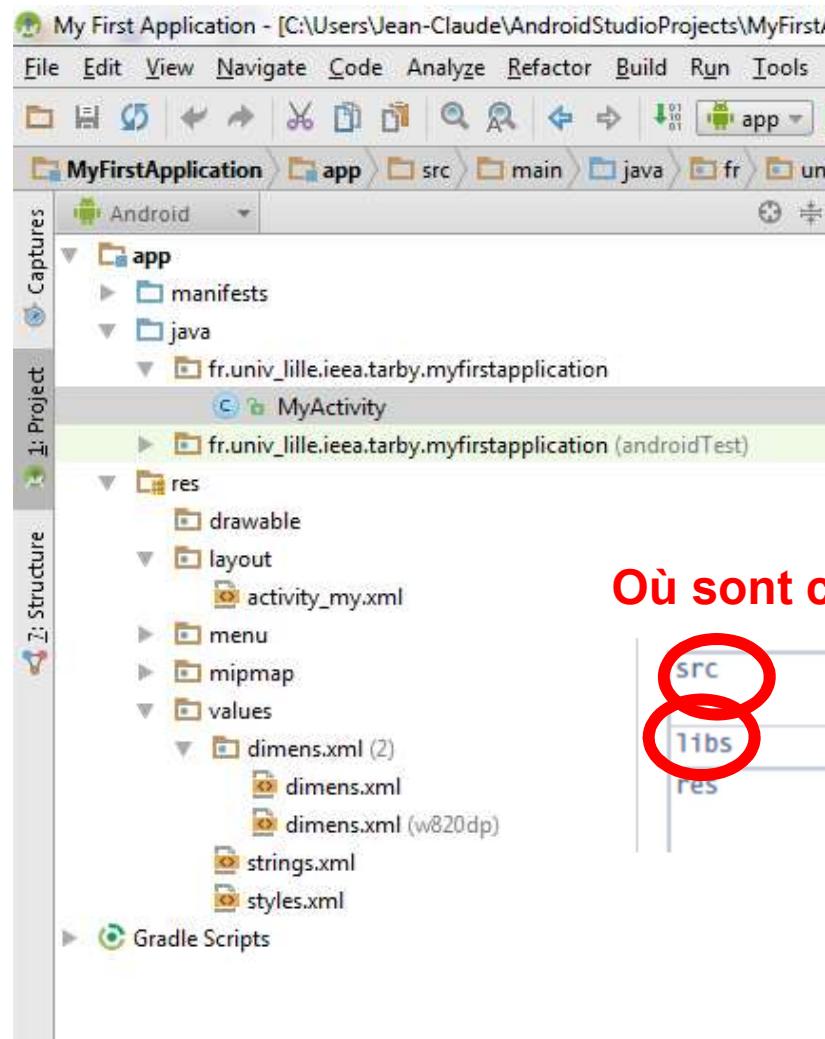


Anciennement
avec ADT

Testons avec
Android Studio...



Première application (Android Studio)



Où sont ces dossiers ?

src	Le répertoire de l'ensemble des sources du projet. C'est dans ce répertoire que vous allez ajouter et modifier le code source de l'application.
libs	Contient les bibliothèques tierces qui serviront à votre application.
res	Contient toutes les ressources telles que les images, les dispositions de l'interface graphique, etc. nécessaires à l'application. Ces ressources seront accessibles grâce à la classe R décrrite plus haut.



Première application (Android Studio)

My First Application - [C:\Users\Jean-Claude\AndroidStudioProjects\MyFirstApplication] - [app] - ...app\src\main\java\fr\univ_lille\ieea\tarby\myfirstapplication\MyActivity.java - Android Studio 1.3

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

MyFirstApplication app src main java univ_lille ieea tarby myfirstapplication MyActivity

Android Captures Project Structure

app manifests java fr.univ_lille.ieea.tarby.myfirstapplication MyActivity fr.univ_lille.ieea.tarby.myfirstapplication (androidTest) res drawable layout activity_my.xml menu mipmap values dimens.xml (2) strings.xml styles.xml Gradle Scripts

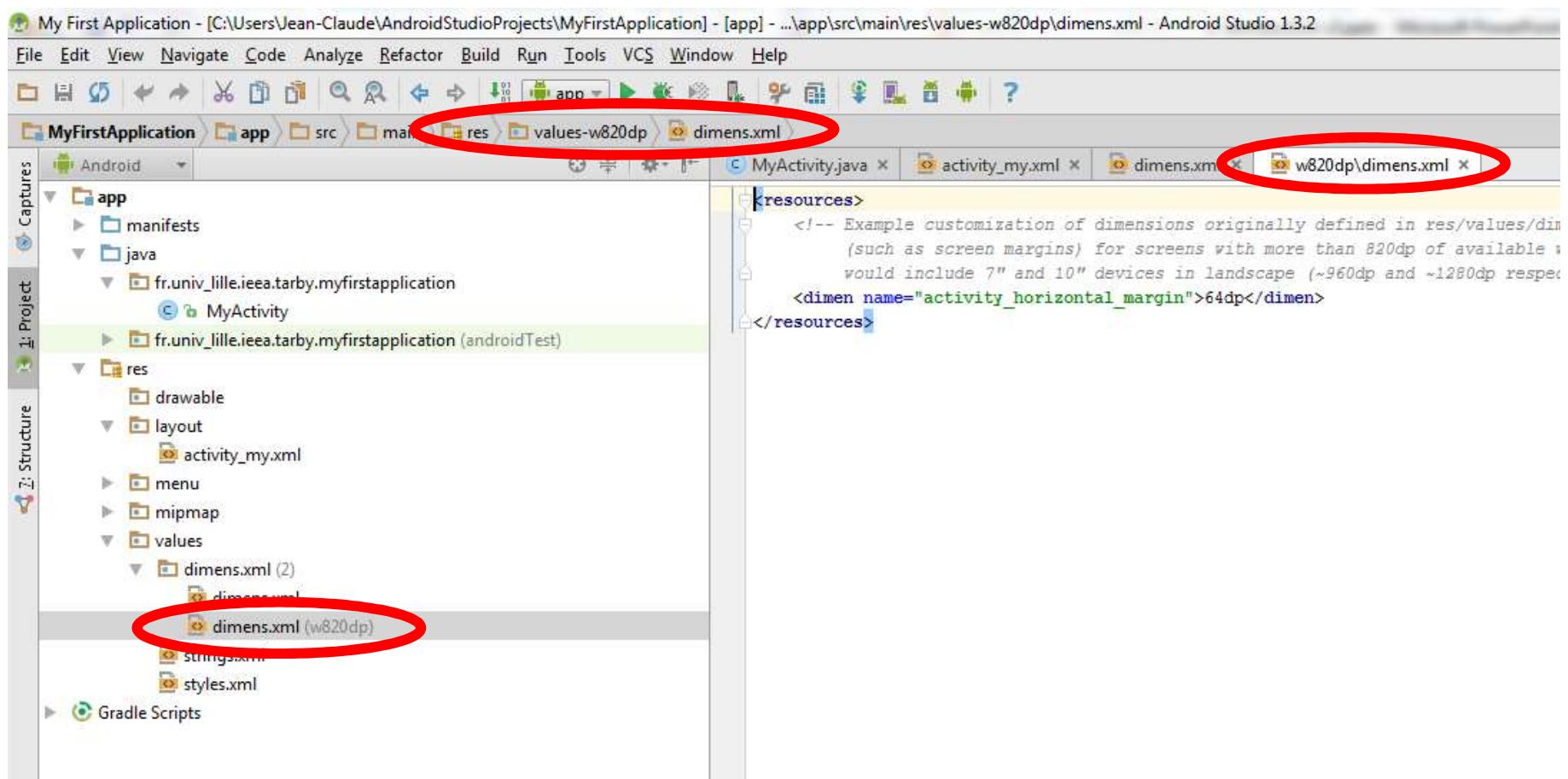
MyActivity.java activity_my.xml dimens.xml w820dp\dimens.xml

```
package fr.univ_lille.ieea.tarby.myfirstapplication;  
import ...  
  
public class MyActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_my);  
  
        public boolean onCreateOptionsMenu(Menu menu) {  
            // Inflate the menu; this adds items to the action bar if it is present  
            getMenuInflater().inflate(R.menu.menu_my, menu);  
            return true;  
        }  
  
        @Override  
        public boolean onOptionsItemSelected(MenuItem item) {  
            // Handle action bar item clicks here. The action bar will  
            // automatically handle clicks on the Home/Up button, so long  
            // as you specify a parent activity in AndroidManifest.xml.  
            int id = item.getItemId();
```

Mais pas de dossier SRC visible ici...



Première application (Android Studio)





Première application (Android Studio)

The screenshot shows the Android Studio interface for a project named "My First Application". The project tree on the left shows the structure: res (circled in green), app, src, main, java, fr, univ_lille, ieea, tarby, myfirstapplication, and MyActivity.java. The code editor on the right displays the contents of MyActivity.java:

```
package fr.univ_lille.ieea.tarby.myfirstapplication;

import ...

public class MyActivity extends AppCompatActivity {

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

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present
        getMenuInflater().inflate(R.menu.menu_my, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
```

A green box highlights the text "Réponse : parce qu'on est en vue « Android » et pas en vue « Project »".



Première application

- Le répertoire **res** est composé de différents sous-répertoires dont les suivants :
 - **res/drawableXXX** : contient les ressources de type image (PNG, JPEG et GIF) ;
 - **res/layout** : contient les descriptions des interfaces utilisateur ;
 - **res/values** : contient les chaînes de caractères, les dimensions, etc. ;
 - **res/xml** : contient les fichiers XML supplémentaires (préférences, etc.) ;
 - **res/menu** : contient la description des menus ;
 - **res/raw** : contient les ressources autres que celles décrites ci-dessus qui seront empaquetées sans aucun traitement

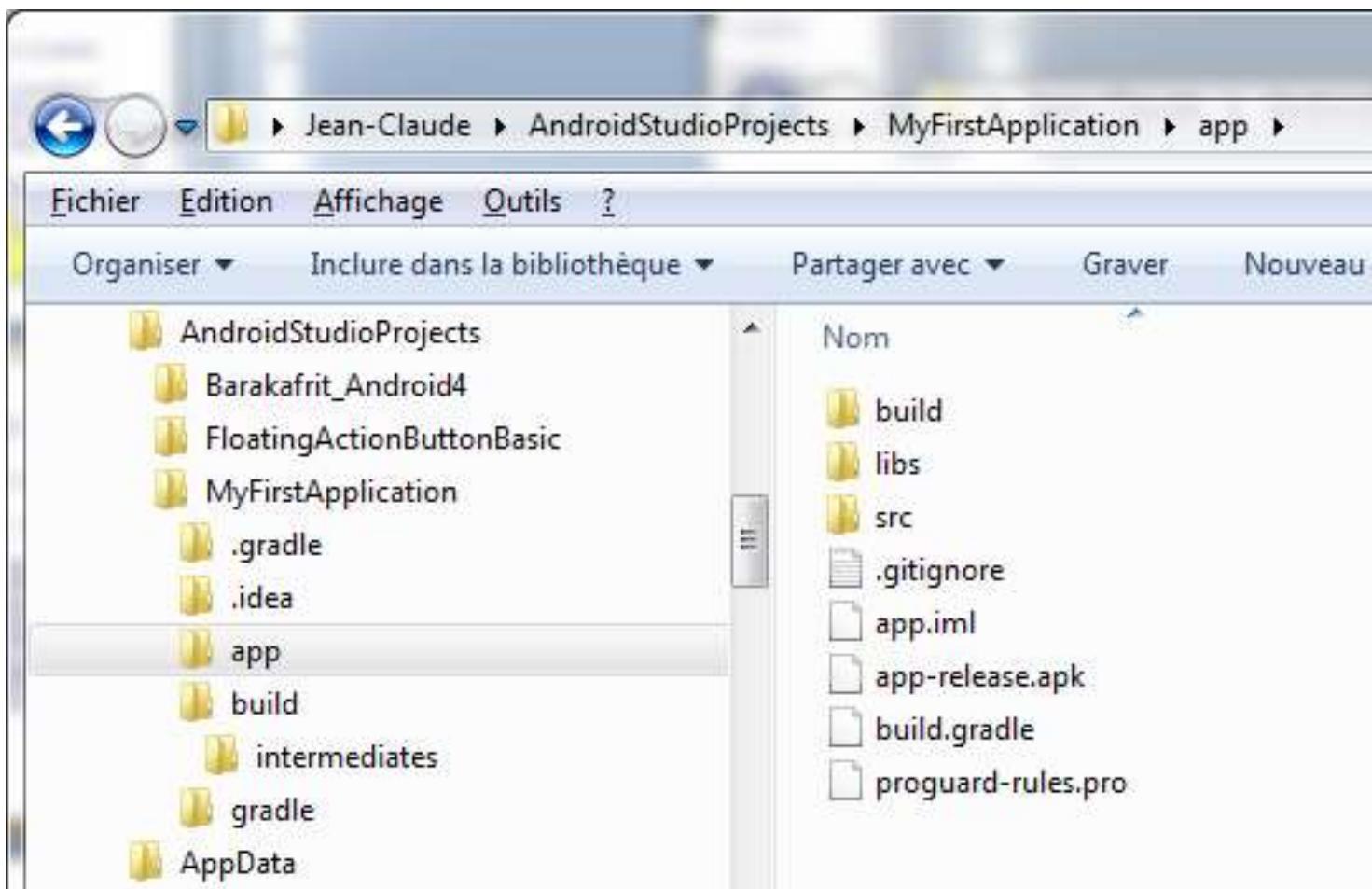


Première application

- Quand vous compilez une application, le résultat sera placé selon l'arborescence suivante :
 - **app/build/intermediates/classes/(votre package)**: les classes compilées en '.class' ;
 - **app/build/outputs/apk** : il s'agit de votre application compilée (mais pas signé !) et empaquetée pour être déployée dans le système Android. Ce fichier contient le code java compilé, les ressources compilées et non compilées (celles contenues dans le répertoire /raw) et enfin le fichier de configuration de l'application.
 - C'est l'apk qu'on donne à quelqu'un pour tester l'application sur un autre téléphone, mais il faut encore le signer numériquement pour le déposer réellement.
 - Signature de l'apk → apk dans dossier des projets Android Studio/MonAppli/app
 - Exemple dans C:\Users\Jean-Claude\AndroidStudioProjects\MyFirstApplication\app (cf. slide suivant)



Première application

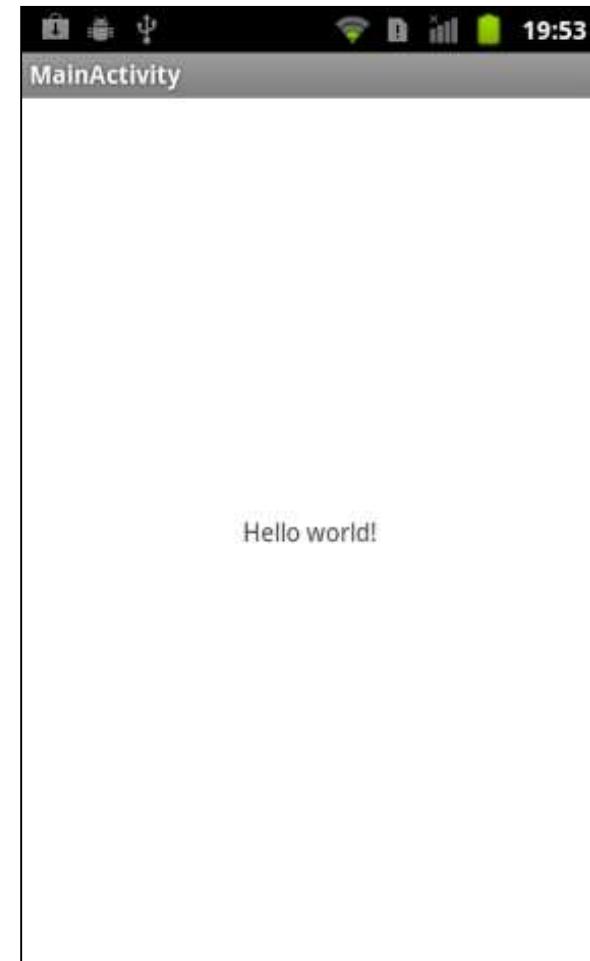




Première application

1. Test de l'application sur l'émulateur

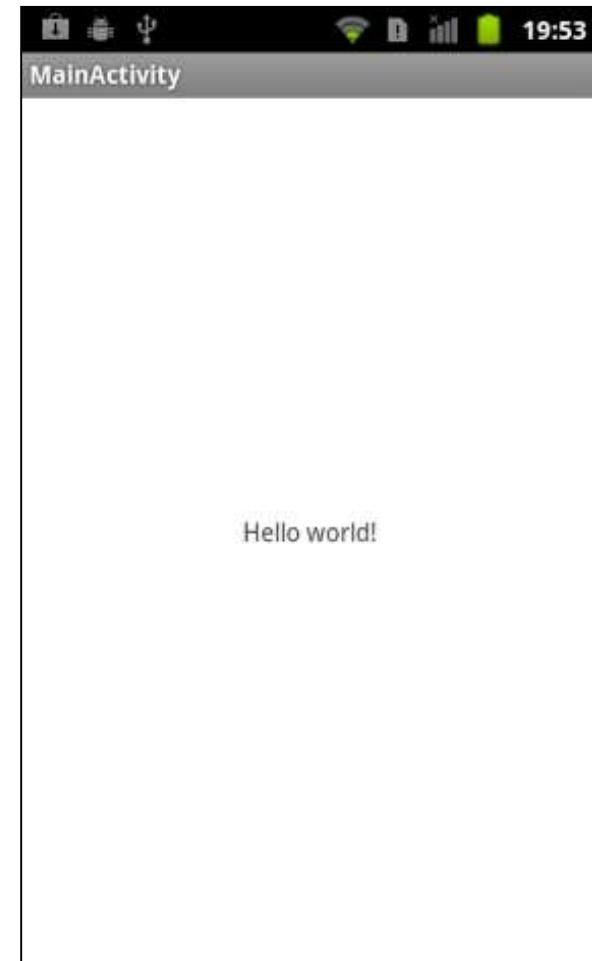
- On doit avoir un device virtuel (AVD) existant (le créer la première fois)
- Long à démarrer la 1^{ère} fois, donc ne pas le fermer chaque fois
- Peut-être des soucis pour l'arrêter ?





Première application

1. Test de l'application sur l'émulateur
 - On doit avoir un device virtuel (AVD) existant (le créer la première fois)
 - Long à démarrer la 1ère fois, donc ne pas le fermer chaque fois
 - Peut-être des soucis pour l'arrêter ?
2. **Test de l'application sur le téléphone connecté**
 - Connecter un téléphone sur le PC (et vérifier que tout est OK) ; vérifier dans Android Studio (onglet Android en bas à gauche) que le téléphone est bien reconnu
 - Lancer l'application (flèche verte dans la barre d'outils)
3. Si tout est OK :
 - (regarder rapidement le dossier generated)
 - l'icône créée sur le téléphone





Première application

- Jetez aussi un œil sur les fichiers « gradle »
 - Dependencies
 - Version du sdk
 - ...
 - Pensez à « mettre votre nez dedans » + tard si problème de compatibilité (par exemple)



Publier sur le Play Store

- Plus facile et rapide que pour iOS ! ☺



Nouvelle version en novembre 2016 :
<https://support.google.com/googleplay/android-developer/answer/7159011>

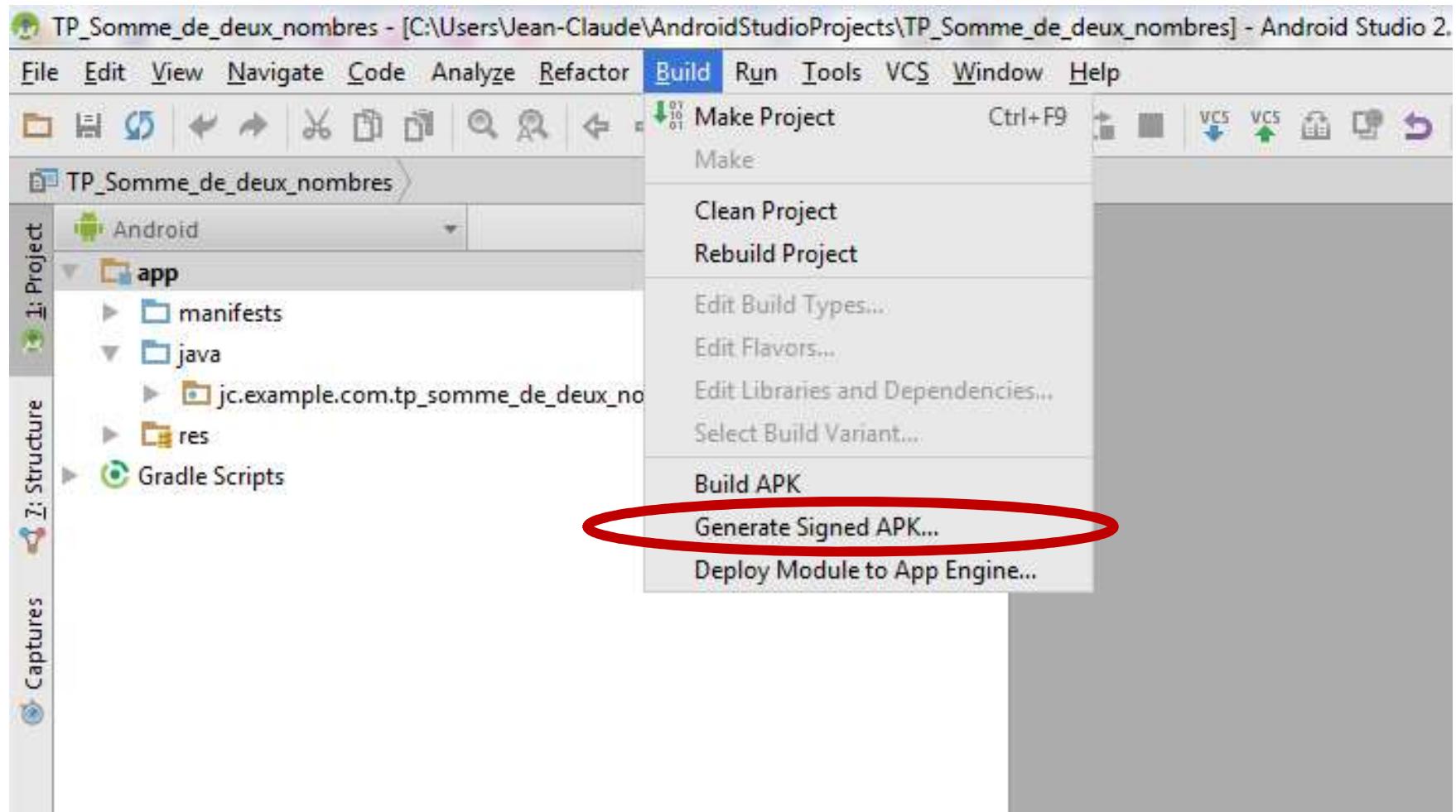


Publier sur le Play Store

1. Créer le projet et le **tester à fond** ! ☺
2. Générer l'APK signé
 - Utiliser (par exemple) l'assistant dans l'onglet du Manifest
 - Ou bien « Export... »
3. Si nécessaire, créer un compte développeur (payant, une seule fois !)
4. Publier
 - Attention ! Les packages de l'application sont l'identifiant sur le Play Store
 - Exemple:
 - com.andrilex.barakafrit.activities
 - <https://play.google.com/store/apps/details?id=com.andrilex.barakafrit.activities>

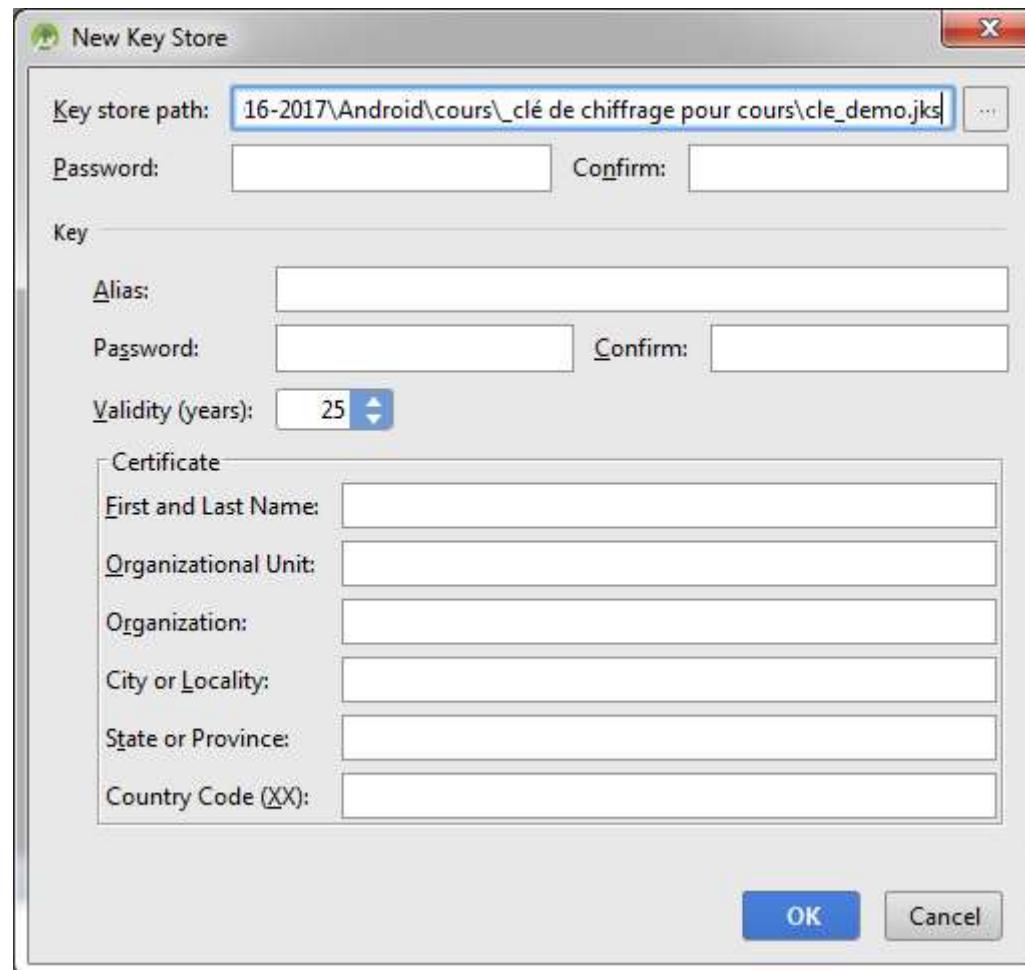


Générer un APK signé





Créer une clé de signature





Créer une clé de signature

- **NE LA PERDEZ SURTOUT PAS ENSUITE**
sinon vous ne pourrez plus jamais mettre à jour votre application !
 - Google ne les stocke pas, vous ne pouvez donc pas la redemander.
- Depuis peu, Google propose un autre système qui ne nécessite plus de clé de ce type
 - (cherchez sur Google...)



APK et fiche

- Déposez ensuite votre APK sur le Play Store et remplissez la fiche associée
 - Ceci peut être fait dans n'importe quel ordre, mais la fiche est obligatoire
 - La fiche doit être remplie correctement
 - N'oubliez pas de spécifier les pays pour votre application !
 - Ensuite, on peut demander à publier l'application
 - Peut être publiée en version alpha, beta, ou en production
- Google ajoute régulièrement de nouveaux services pour le Play Store



Les applications publiées

The screenshot shows the 'Toutes les applications' (All Applications) page in the Google Play Developer Console. The URL in the browser is https://play.google.com/apps/publish/?dev_acc=04946778639877766815&pli=1#AppListPlace. The page displays two published applications:

NOM DE L'APPLICATION	PRIX	INSTALLATIONS ACTUELLES/TOTAL	NOTE MOY./NOMBRE TOTAL	PLANTAGES ET ANR	DERNIÈRE MISE À JOUR	ÉTAT
Barakafrit 2.4.1	Gratuite	715 / 4 442	★ 4,19 / 52	4	1 oct. 2013	Publiée
Boite à coucou 1.2	Gratuite	—	—	—	10 nov. 2014	Publiée



Fiche d'une application

The screenshot shows the Google Play Developer Console interface. On the left, there's a sidebar with various tabs: Applications, _mobile, _vitrine, important, maisons sur le bon c..., pages ouvertes, Calendrier Lunaire, Connect androphone, Connect androphone, and Autres favoris. The 'Fiche Google Play Store' tab is currently selected. The main area displays the app 'BARAKAFRIT' (com.andrilex.barakafrit.activities). The title bar shows 'Publiée' (Published). The 'INFORMATIONS SUR LE PRODUIT' section includes fields for 'Titre*' (Title), 'Description courte*' (Short Description), and 'Description complète*' (Full Description). The 'Titre*' field contains 'Barakafrit'. The 'Description courte*' field contains 'Trouvez, Notez, Ajoutez, les friteries autour de vous en un clic !'. The 'Description complète*' field contains a detailed description of the app's features. A note at the bottom of the description section says '318 caractère(s) sur 4000'.



Les APK

The screenshot shows the Google Play Developer Console interface for managing APK files. The main page displays the app's logo, name (BARAKAFRIT), developer information (Jean-Claude Tarby, Adrien Gooris, Alexandre Wambre), and a status bar indicating the app is published.

FICHIERS APK

- PRODUCTION**: Version 10
- TESTS BÊTA**: Configurez des tests bêta pour votre application.
- TESTS ALPHA**: Configurez des tests alpha pour votre application.

CONFIGURATION DE LA VERSION EN PRODUCTION

FICHIER APK ACTUEL: date de publication : 14 févr. 2012 04:38:35

Appareils compatibles	Appareils exclus
5327	0

ACTIONS

VERSION	IMPORTÉ LE	ÉTAT
10 (2.4.1)	-	en production

AUTRES FICHIERS APK

Masquer

Master Services



Les commentaires

The screenshot shows the Google Play Developer Console interface. The left sidebar has a navigation menu with icons for apps, games, stats, notes, ANRs, optimization tips, APK files, Play Store info, pricing, integrated products, and services/APIs. The main content area is titled 'BARAKAFRIT - com.andrilex.barakafrit.activities'. It features a 'Publiée' (Published) status indicator. Below it is a section for 'Télécharger les notes et les avis' (Download reviews and ratings) with a note that CSV files are available on Google Cloud Storage. A 'NOTES ET COMMENTAIRES' section displays a summary: 52 reviews, a 4.19 average rating, and a star distribution chart. The chart shows the following counts for each star rating:

Rating	Count
5 stars	28
4 stars	14
3 stars	4
2 stars	4
1 star	2

The 'AVIS' section shows two review snippets:

- Christophe Buyse le 14 oct. 2014 à 16:10: Galaxy Note3 Neo (hlite) ★★★★★
Dommage ne fonctionne pas sur note 3 lite
[Répondre à cet avis](#)
- Un utilisateur Google le 1 juil. 2014 à 14:16: Version de l'application : 2.4.1
Galaxy S4 (jflte) ★★★★★
Traduit automatiquement à partir de la langue suivante : Anglais
[Super Afficher l'avis original](#)

At the bottom right, it says 'Page 1 sur 1'.



Statistiques de téléchargement

Statistiques - Barakafrit - https://play.google.com/apps/publish/?dev_acc=04946778639877766815&pli=1#StatsPlace:p=com.andrilex.barakafrit.activities&statm=1

Applications _mobile _vitrine important maisons sur le bon c... pages ouvertes Calendrier Lunaire - ... Connect androphone Connect androphone » Autres favoris

Google play Developer Console

Toutes les applications Services de jeux Rapports financiers Paramètres Alertes Annonces

BARAKAFRIT - com.andrilex.barakafrit.activities Afficher sur le Google Play Store

STATISTIQUES Installations actuelles (appareils) pour la période 8 oct. 2014 - 8 nov. 2014 Exporter au format CSV Afficher : le mois dernier 3 mois 6 mois 1 an toutes ✓ Publiée

Nombre d'appareils actifs sur lesquels l'application est installée actuellement En savoir plus

Version d'Android Appareil Tablettes Pays Langue Version Opérateur

INSTALLATIONS ACTUELLES (APPAREILS) PAR VERSION D'ANDROID

INSTALLATIONS ACTUELLES (APPAREILS) LE 8 NOV. 2014

VOTRE APPLICATION	TOUTES LES APPLICATIONS DANS LA CATÉGORIE "VOYAGE ET INFO LOCALE"	TOP 10 DES VERSIONS D'ANDROID POUR LA CATÉGORIE "VOYAGE ET INFO LOCALE"
Android 4.4 198 27,69 %	198 27,69 %	Android 4.4 28,10 %
Android 4.1 136 19,02 %	136 19,02 %	Android 2.3.3 - 2.3.7 18,79 %
Android 2.3.3 - 2.3.7 124 17,34 %	124 17,34 %	Android 4.1 17,94 %
Android 4.3 91 12,73 %	91 12,73 %	Android 4.0.3 - 4.0.4 13,82 %
Android 4.0.3 - 4.0.4 66 9,23 %	66 9,23 %	Android 4.2 10,76 %
Android 4.2 52 7,27 %	52 7,27 %	Android 4.3 6,36 %
Android 2.2 44 6,15 %	44 6,15 %	Android 2.2 3,19 %
Android 2.1 4 0,56 %	4 0,56 %	Android 3.2 0,46 %
		Android 2.1 0,34 %
		Android 3.1 0,13 %

DÉCOUVREZ DAVANTAGE DE STATISTIQUES AVEC GOOGLE ANALYTICS POUR LES APPLICATIONS MOBILES



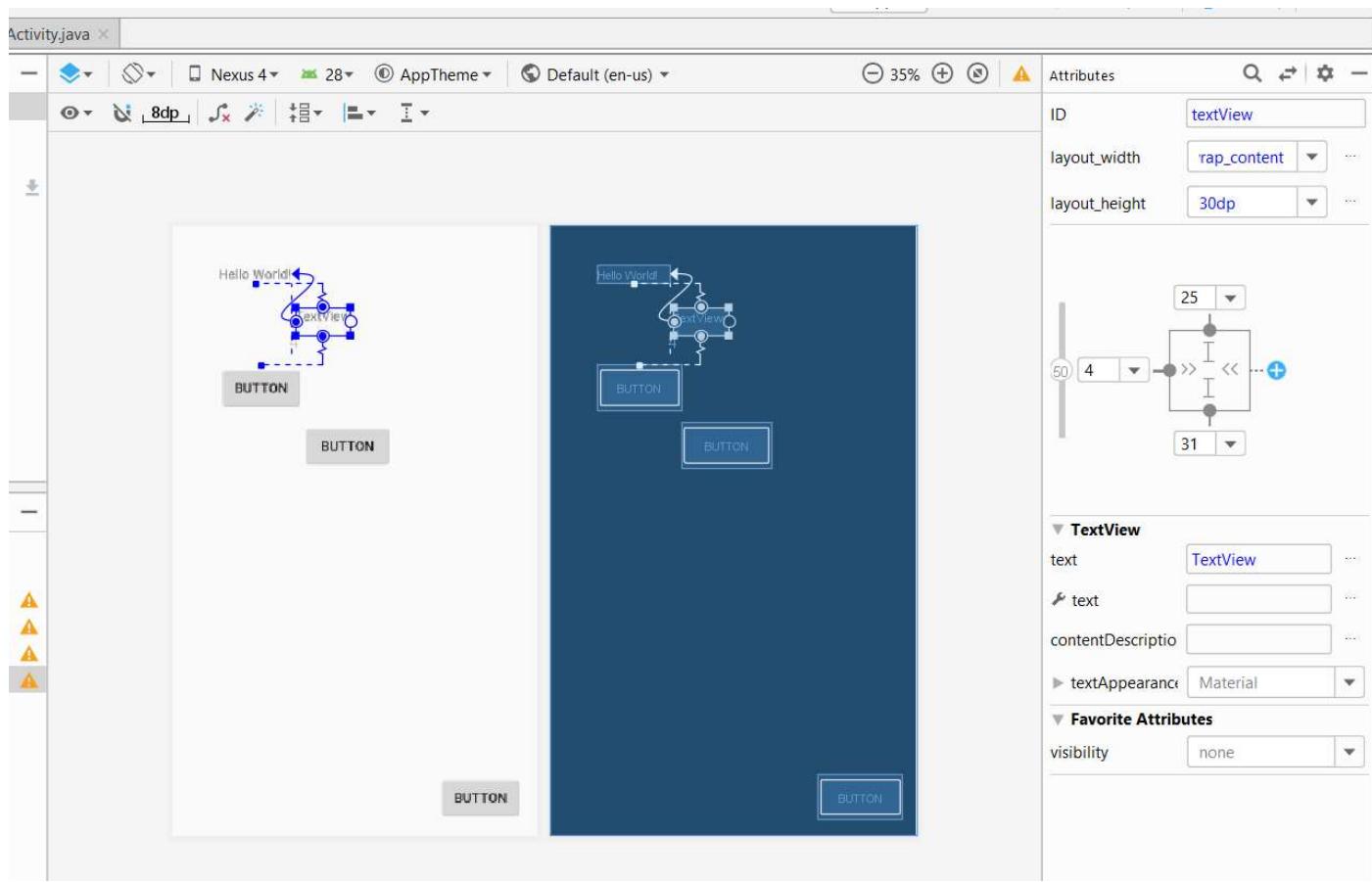
Attention à vos périphériques !





« pause clavier »

- On joue avec le Constraint Layout...
 - Cf. source <http://tutos-android-france.com/constraintlayout-parte-1/>





Constraint Layout

- Faites des essais par vous-mêmes maintenant
 - Faites des mises en page pour jouer avec les widgets
 - Faites des mises en page que vous aurez faites sur papier avant
 - Dans tous les cas, essayez ensuite de modifier ces mises en page...
 - Vous pouvez aussi essayer les RelativeLayout ou les LinearLayout si vous voulez...
- Cf.
https://gitlab.com/m2eservices/constraint_layout.git



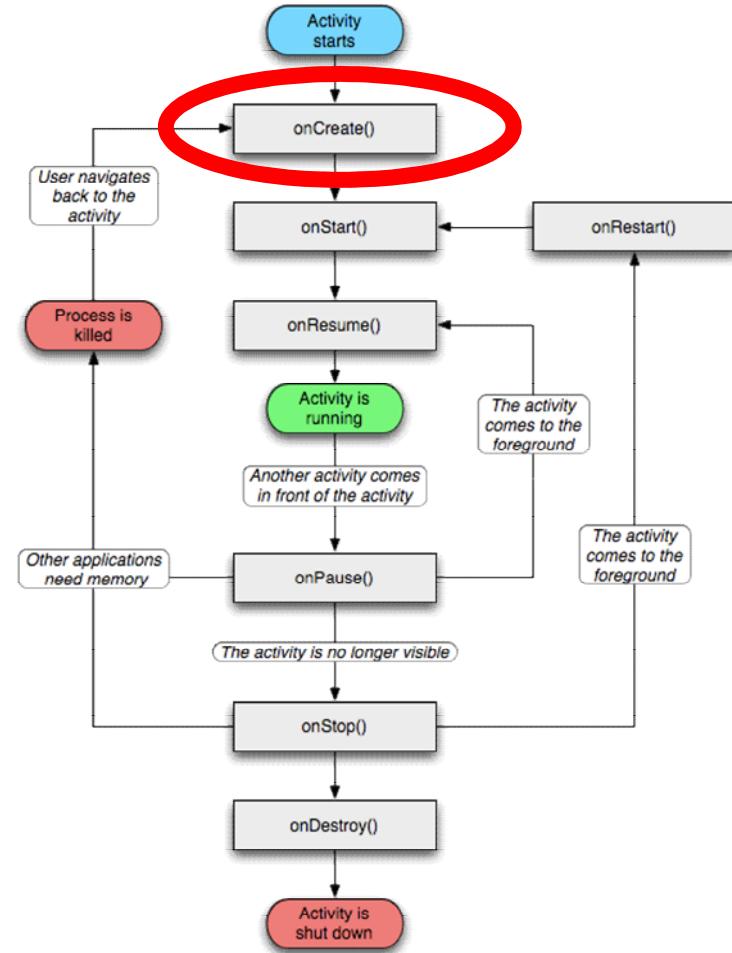
Cycle de vie d'une activité

- Des (bonnes) pratiques différentes...
- Voici quelques exemples...



Squelette d'une activité 1/5

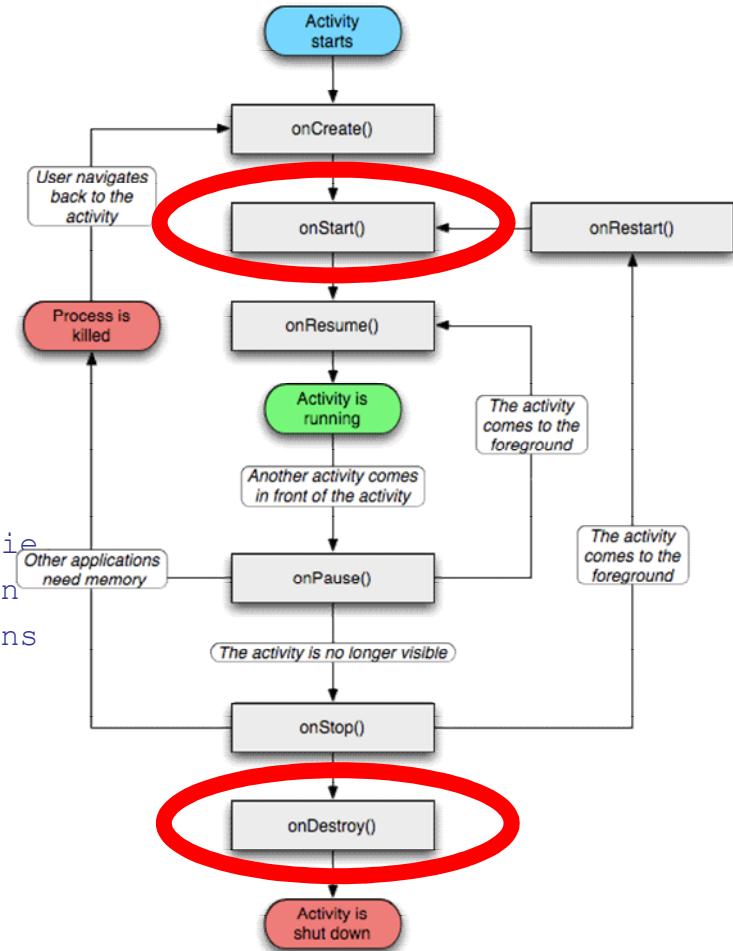
```
package com.eyrolles.android.activity;  
import android.app.Activity;  
import android.os.Bundle;  
  
public final class TemplateActivity extends Activity {  
  
    /**  
     * Appelée lorsque l'activité est créée.  
     * Permet de restaurer l'état de l'interface  
     * utilisateur grâce au paramètre savedInstanceState.  
     */  
  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        // Placez votre code ici  
    }  
}
```





Squelette d'une activité 2/5

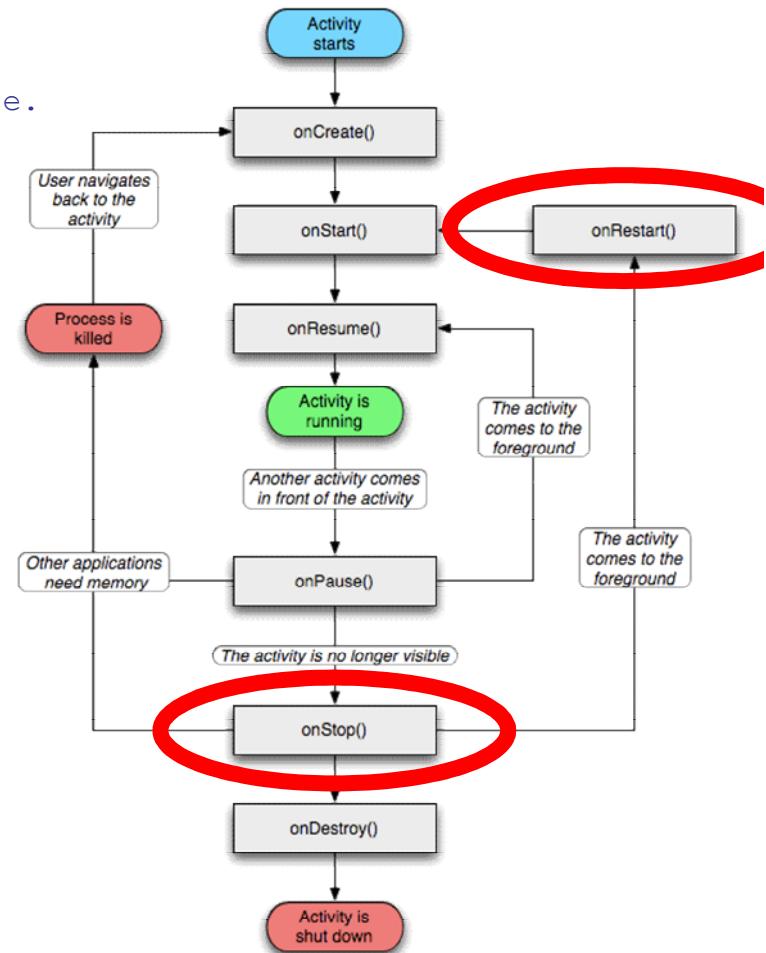
```
/**  
 * Appelée lorsque l'activité démarre.  
 * Permet d'initialiser les contrôles.  
 */  
  
@Override  
  
public void onStart(){  
super.onStart();  
// Placez votre code ici  
}  
  
/**  
 * Appelée lorsque que l'activité a fini son cycle de vie  
 * C'est ici que nous placerons notre code de libération  
 * de mémoire, fermeture de fichiers et autres opérations  
 * de "nettoyage".  
 */  
  
@Override  
  
public void onDestroy(){  
// Placez votre code ici  
super.onDestroy();  
}
```





Squelette d'une activité 3/5

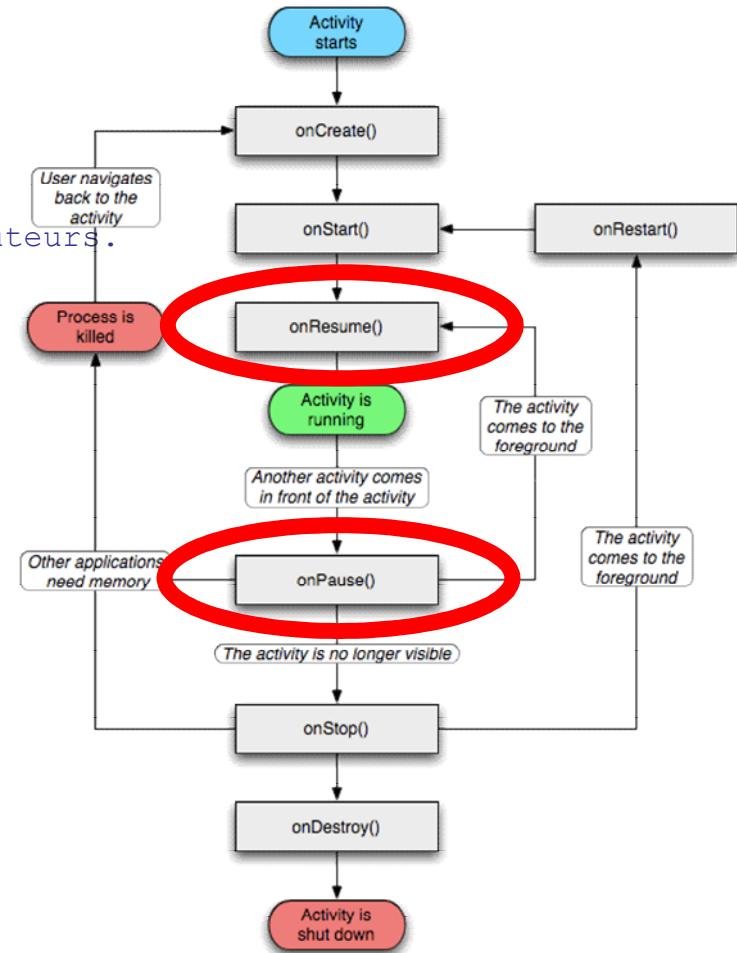
```
/**  
 * Appelée lorsque l'activité sort de son état de veille.  
 */  
  
@Override  
  
public void onRestart() {  
    super.onRestart();  
    //Placez votre code ici  
}  
  
/**  
 * Appelée lorsque l'activité passe en arrière plan.  
 * Libérez les écouteurs, arrêtez les threads, votre  
 * activité peut disparaître de la mémoire.  
 */  
  
@Override  
  
public void onStop() {  
    // Placez votre code ici  
    super.onStop();  
}
```





Squelette d'une activité 4/5

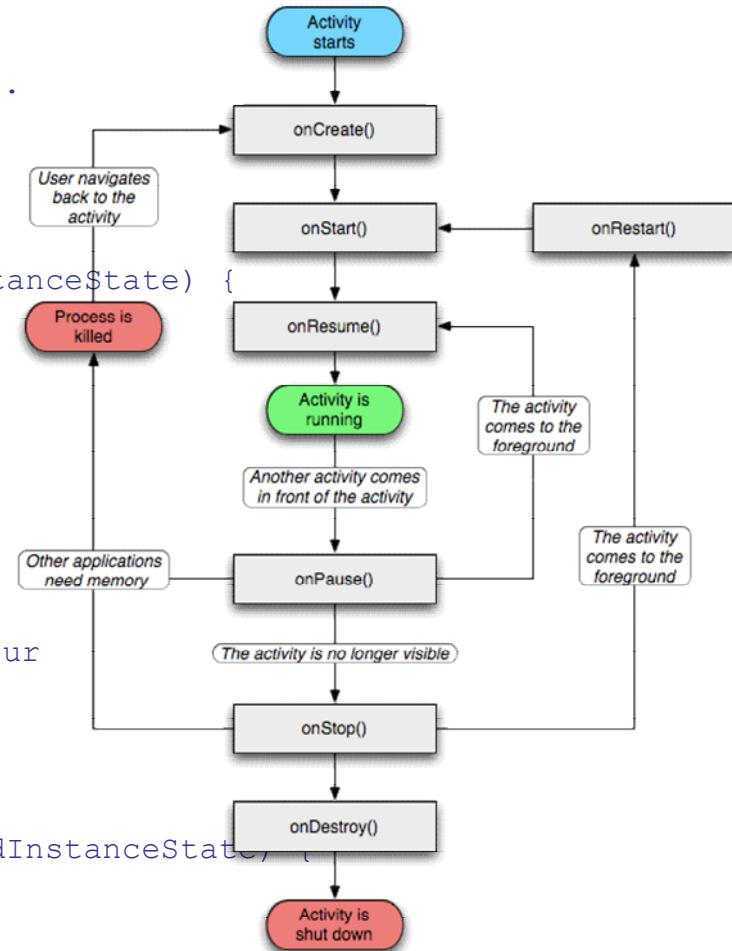
```
/**  
 * Appelée après le démarrage ou une pause.  
 * Relancez les opérations arrêtées (threads).  
 * Mettez à jour votre application et vérifiez vos écouteurs.  
 */  
  
@Override  
  
public void onResume() {  
    super.onResume();  
    // Placez votre code ici  
}  
  
/**  
 * Appelée lorsque que l'activité est suspendue.  
 * Stoppez les actions qui consomment des ressources.  
 * L'activité va passer en arrière-plan.  
 */  
  
@Override  
  
public void onPause() {  
    //Placez votre code ici  
    super.onPause();  
}
```





Squelette d'une activité 5/5

```
/**  
 * Appelée lorsque l'activité termine son cycle visible.  
 * Sauvez les données importantes.  
 */  
  
@Override  
  
public void onSaveInstanceState(Bundle savedInstanceState)  
    // Placez votre code ici  
    // sans quoi l'activité aura perdu son état  
    // lors de son réveil  
    super.onSaveInstanceState(savedInstanceState);  
}  
  
/**  
 * Appelée après onCreate.  
 * Les données sont rechargées et l'interface utilisateur  
 * est restaurée dans le bon état.  
 */  
  
@Override  
  
public void onRestoreInstanceState(Bundle savedInstanceState)  
    super.onRestoreInstanceState(savedInstanceState);  
    //Placez votre code ici  
}
```





onCreate

- Source : <http://mathias-seguy.developpez.com/tutoriels/android/comprendre-cyclevie-activite/>
- La méthode `onCreate` est appelée :
 - au premier lancement de l'activité ;
 - si l'activité est ressuscitée, le bundle passé en paramètre sera celui sauvegardé par `onSaveInstanceState()` ;
 - si l'état du terminal change et que l'activité est associée à cet état (passage du mode portrait au mode paysage).
- Elle configure les IHM et tous les traitements d'initialisation qui ne sont effectués qu'une seule fois au lancement de l'activité.



onDestroy

- La méthode `onDestroy` est appelée lors de la mort de l'activité (soit naturelle, soit par le système).
- Cette méthode doit libérer les ressources allouées dans la méthode `onCreate`.
- Parfois, l'urgence du système détruira l'activité sans même appeler `onDestroy`. La seule méthode qui est appelée avec certitude avant la destruction de l'activité est `onPause()`.
- Les méthodes `onStart`, `onRestart` et `onStop` n'ont pas grand intérêt.



Sauvegarde/restauration de l'état

- Les méthodes `onPause` et `onResume` sont celles dans lesquelles l'activité doit sauvegarder ses états et les restituer.
- La méthode `onResume` est appelée immédiatement avant que l'activité ne passe au premier plan. De ce fait, c'est le bon endroit pour reconstruire les données affichées par l'IHM et mettre celle-ci à jour, quitte à lancer un thread de reconstruction des données.
- Inversement, la méthode `onPause` est la seule par laquelle il est certain que l'application passera avant de se faire détruire. Il convient donc de sauvegarder l'état de l'application dans cette méthode.
- La sauvegarde et la restauration s'effectuent au travers des méthodes `onSaveInstanceState` et `onRestoreInstanceState`. L'objet Bundle est passé en paramètre de l'une et renvoyé par l'autre. L'objet Bundle est une carte (map) intelligente qui stocke des paires typées (Key, Object) et possède l'ensemble des méthodes pour enregistrer et récupérer ces paires (`putBoolean(String key, boolean value)` et `boolean getBoolean(String key)` par exemple et ce pour la plupart des types connus).
- Toute la subtilité du mécanisme réside dans le choix fait par le développeur des données à sauvegarder. Il ne doit pas vouloir y mettre la terre entière et être précis dans ce qu'il choisit de stocker et de restaurer.



Gérer ce qui ne se gère pas avec un Bundle

- Typiquement, une connexion, un ensemble de thread filles, les sockets ouverts... comment les sauvegarder et les restaurer quand l'activité passe en pause ?
- Pour cela, il faut utiliser les méthodes `onRetainNonConfigurationInstance` pour leur sauvegarde et `getLastNonConfigurationInstance` pour la restauration. Ces méthodes sont à utiliser en plus des méthodes `onSaveInstanceState` et `onRestoreInstanceState`. La seule restriction est que ces méthodes ne peuvent faire référence à une ressource détruite lors du passage de pause à resume.
- Ces méthodes sont très utiles lors de la rotation qui, rappelons-le, détruit et reconstruit l'application.



Exemple de sauvegarde / restauration d'état

Source : <https://developer.android.com/guide/components/activities/activity-lifecycle.html>

Sauvegarde de « l'état »

```
static final String STATE_SCORE = "playerScore";
static final String STATE_LEVEL = "playerLevel";

@Override
public void onSaveInstanceState(Bundle savedInstanceState) {

    // Save the user's current game state
    savedInstanceState.putInt(STATE_SCORE, mCurrentScore);
    savedInstanceState.putInt(STATE_LEVEL, mCurrentLevel);

    // Always call the superclass so it can save the view hierarchy state
    super.onSaveInstanceState(savedInstanceState);
}
```



Exemple de restauration d'état

Restauration de « l'état » (solution avec onCreate)

- Both the `onCreate()` and `onRestoreInstanceState()` callback methods receive the same `Bundle` that contains the instance state information.
- Because the `onCreate()` method is called whether the system is creating a new instance of your activity or recreating a previous one, you must check whether the state Bundle is null before you attempt to read it. If it is null, then the system is creating a new instance of the activity, instead of restoring a previous one that was destroyed

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState); // Always call the superclass first  
  
    // Check whether we're recreating a previously destroyed instance  
    if (savedInstanceState != null) {  
        // Restore value of members from saved state  
        mCurrentScore = savedInstanceState.getInt(STATE_SCORE);  
        mCurrentLevel = savedInstanceState.getInt(STATE_LEVEL);  
    } else {  
        // Probably initialize members with default values for a new instance  
    }  
}
```



Exemple de restauration d'état

Restauration de « l'état » (solution avec `onRestoreInstanceState()`)

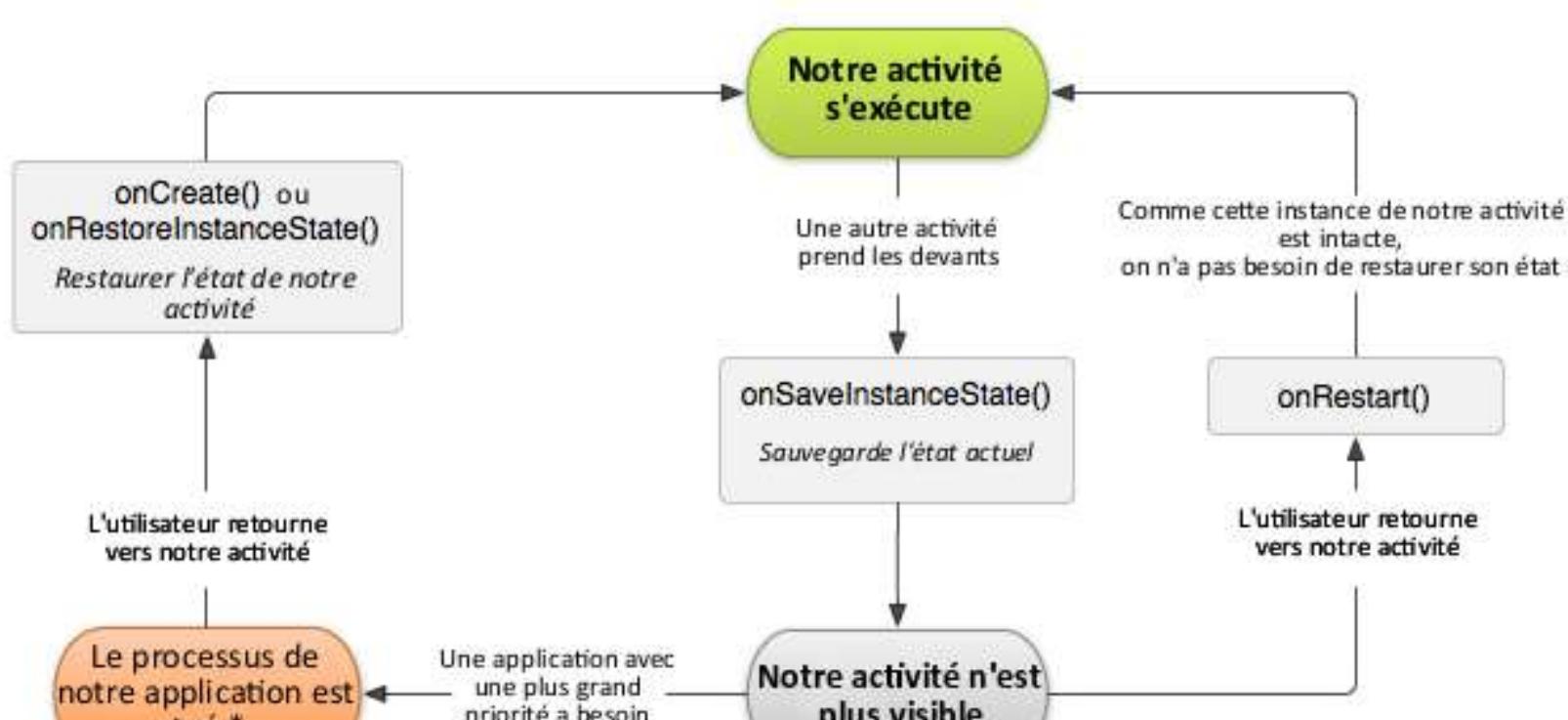
- Attention ! Always call the superclass implementation of `onRestoreInstanceState()` so the default implementation can restore the state of the view hierarchy
- Instead of restoring the state during `onCreate()` you may choose to implement `onRestoreInstanceState()`, which the system calls after the `onStart()` method. The system calls `onRestoreInstanceState()` only if there is a saved state to restore, so you do not need to check whether the `Bundle` is null:

```
public void onRestoreInstanceState(Bundle savedInstanceState) {  
  
    // Always call the superclass so it can restore the view hierarchy  
    super.onRestoreInstanceState(savedInstanceState);  
  
    // Restore state members from saved instance  
    mCurrentScore = savedInstanceState.getInt(STATE_SCORE);  
    mCurrentLevel = savedInstanceState.getInt(STATE_LEVEL);  
}
```



État et cycle de vie

- Source : <https://openclassrooms.com/courses/creez-des-applications-pour-android/preamble-quelques-concepts-avances>



* L'instance de notre application est tuée mais les données du onSaveInstanceState() sont conservées

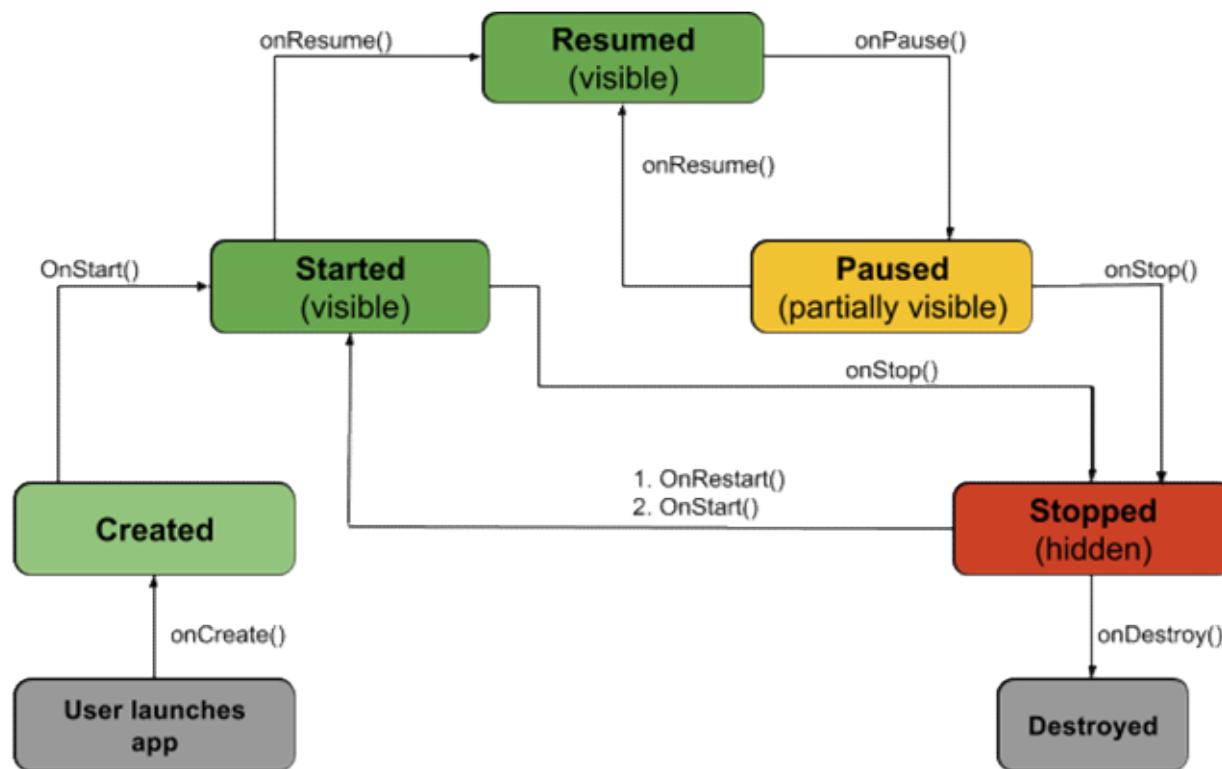


État et cycle de vie

- `onRetainNonConfigurationInstance` est appelée uniquement en cas de changement de configuration.
- Elle est appelée après `onStop()` mais avant `onDestroy()`
 - ne pas conserver des objets qui dépendent de la configuration (par exemple des chaînes de caractères qui changent en fonction de la langue) ou des objets qui sont liés à l'activité



Une autre proposition du cycle de vie et de l'utilisation des états



<https://www.raywenderlich.com/500-introduction-to-android-activities-with-kotlin>



- **onCreate()**
 - Called by the OS when the activity is first created. This is where you initialize any UI elements or data objects. You also have the savedInstanceState of the activity that contains its previously saved state, and you can use it to recreate that state.
- **onStart()**
 - Just before presenting the user with an activity, this method is called. It's always followed by onResume(). In here, you generally should start UI animations, audio based content or anything else that requires the activity's contents to be on screen.
- **onResume()**
 - As an activity enters the foreground, this method is called. Here you have a good place to restart animations, update UI elements, restart camera previews, resume audio/video playback or initialize any components that you release during onPause().
- **onPause()**
 - This method is called before sliding into the background. Here you should stop any visuals or audio associated with the activity such as UI animations, music playback or the camera. This method is followed by onResume() if the activity returns to the foreground or by onStop() if it becomes hidden.



- **onStop()**
 - This method is called right after onPause(), when the activity is no longer visible to the user, and it's a good place to save data that you want to commit to the disk. It's followed by either onRestart(), if this activity is coming back to the foreground, or onDestroy() if it's being released from memory.
- **onRestart()**
 - Called after stopping an activity, but just before starting it again. It's always followed by onStart().
- **onDestroy()**
 - This is the final callback you'll receive from the OS before the activity is destroyed. You can trigger an activity's destruction by calling finish(), or it can be triggered by the system when the system needs to recoup memory. If your activity includes any background threads or other long-running resources, destruction could lead to a memory leak if they're not released, so you need to remember to stop these processes here as well.
- Note: You do not call any of the above callback methods directly in your own code (other than superclass invocations) — you only override them as needed in your activity subclasses. They are called by the OS when a user opens, hides or exits the activity.



Testez !

- Cf. projet Demo_cycledevie_Activity
 - <https://gitlab.com/m2eservices/democycledevie.git>
- Lancez l'appli et une fois lancée, tournez l'écran !