Android


Cupcake
Donut
Eclair
Froyo
Gingerbread
Honeycomb
ICE Cream-Sandwitch
Jelly Bean
Kitkat
lollipop
Lollipop
Marshmallow
Nougat

Android dans le temps…

Nouveautés

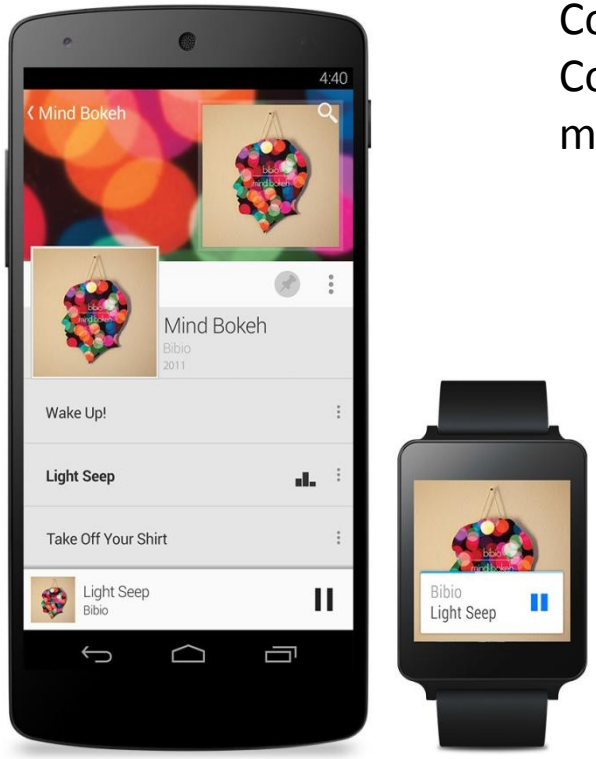
Expérience utilisateur : Material Design et Multi fenêtres
Assistance vocale
Batterie
Sécurité : Autorisations des applications / reconnaissance digitale
Paiement sans contact

Expérience utilisateur : Multi fenêtres et réalité augmentée
Sécurité : chiffrement
Réalité augmentée

http://www.phonandroid.com/android-m-recapitulatif-principales-nouveautes-attendues.html
http://www.journaldunet.com/solutions/dsi/1174773-android-nougat-ou-android-7-0-les-nouveautes/

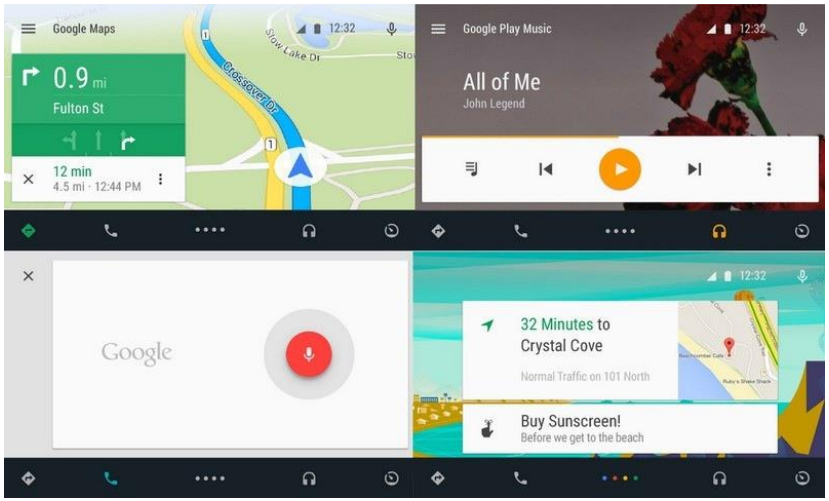
Les dispositifs visés

Des IHM et usages adaptés



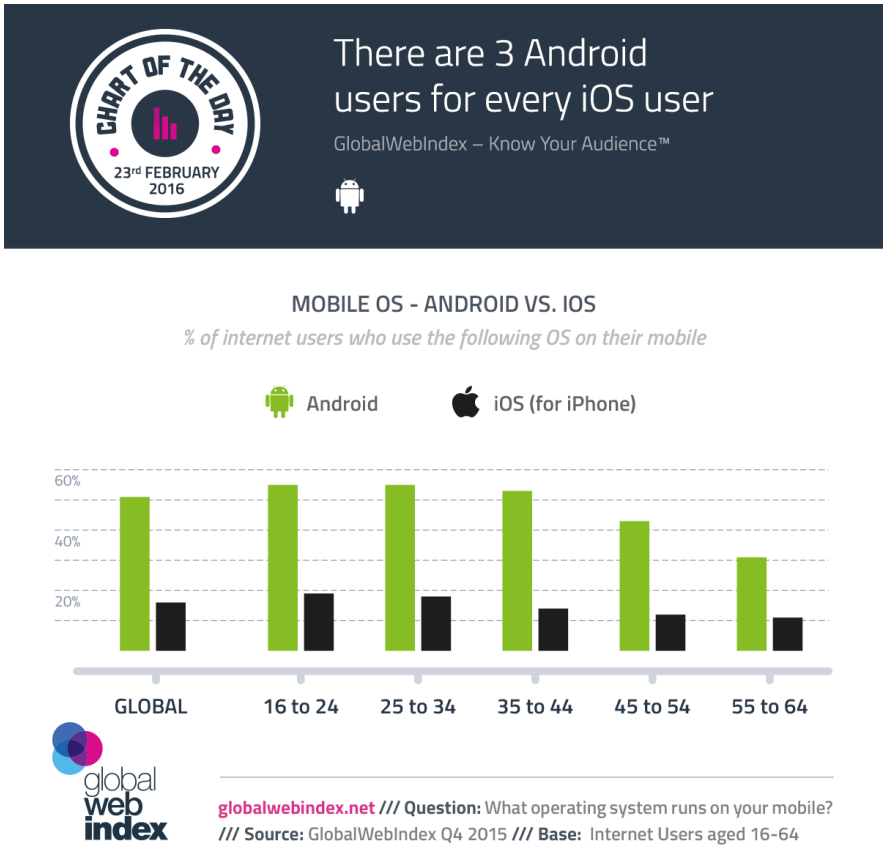
voix et touch
notifications
Commandes déportées

Consommation de médias
Commandé par voix,
manette, téléphone, tablette



Intégration avec l'ordinateur de bord de la
Voiture, Données capteurs de la voiture (vitesse, gps …),
Centre multimédia, Navigation par Google Maps
Interface adaptée : Gros boutons, Lisible…

Android vs IOS

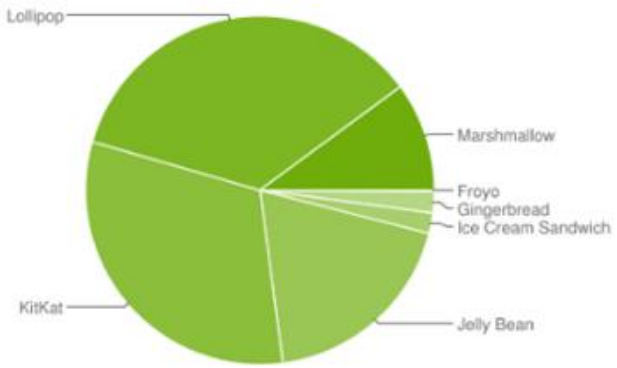


https://www.globalwebindex.net/blog/there-are-3-android-users-for-every-ios-user

Dispositifs et versions

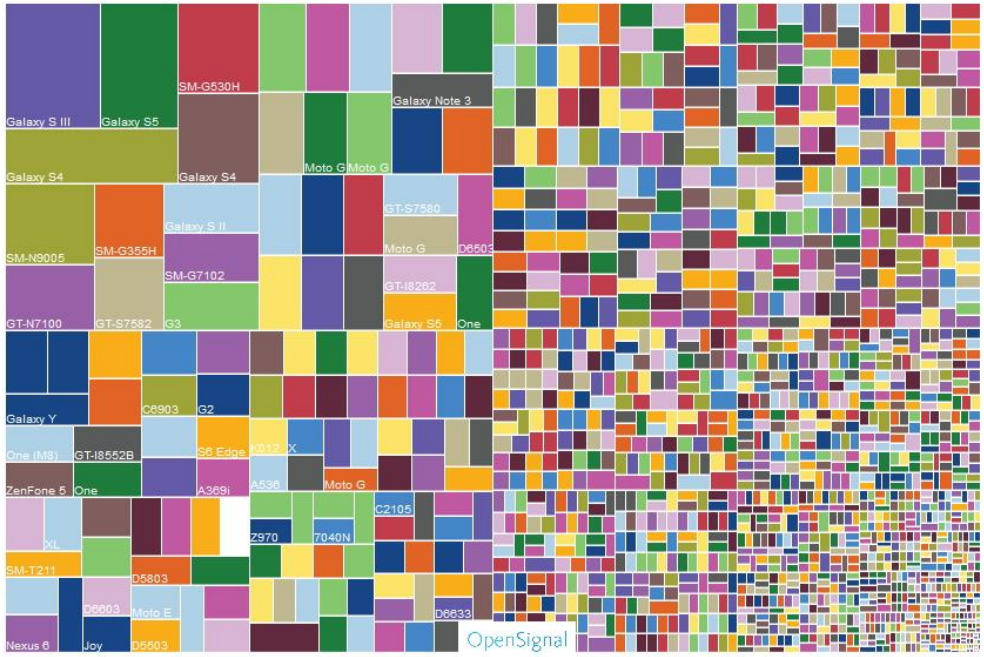
Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	2.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.9%
4.1.x	Jelly Bean	16	6.8%
4.2.x		17	9.4%
4.3		18	2.7%
4.4	KitKat	19	31.6%
5.0	Lollipop	21	15.4%
5.1		22	20.0%
6.0	Marshmallow	23	10.1%

Data collected during a 7-day period ending on June 6, 2016.
Any versions with less than 0.1% distribution are not shown.



https://www.nextinpact.com/news/100168-android-6-0-depasse-10-parts-marche-android-5-1-arrive-a-20.htm

Fragmentation du marché

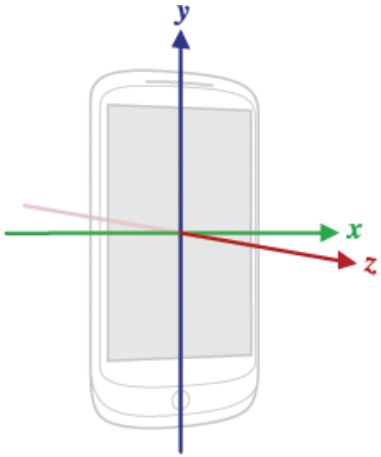


http://www.silicon.fr/android-os-mobile-sacrement-fragmente-123384.html

http://mathias-seguy.developpez.com/tutoriels/android/utiliser-capteurs/

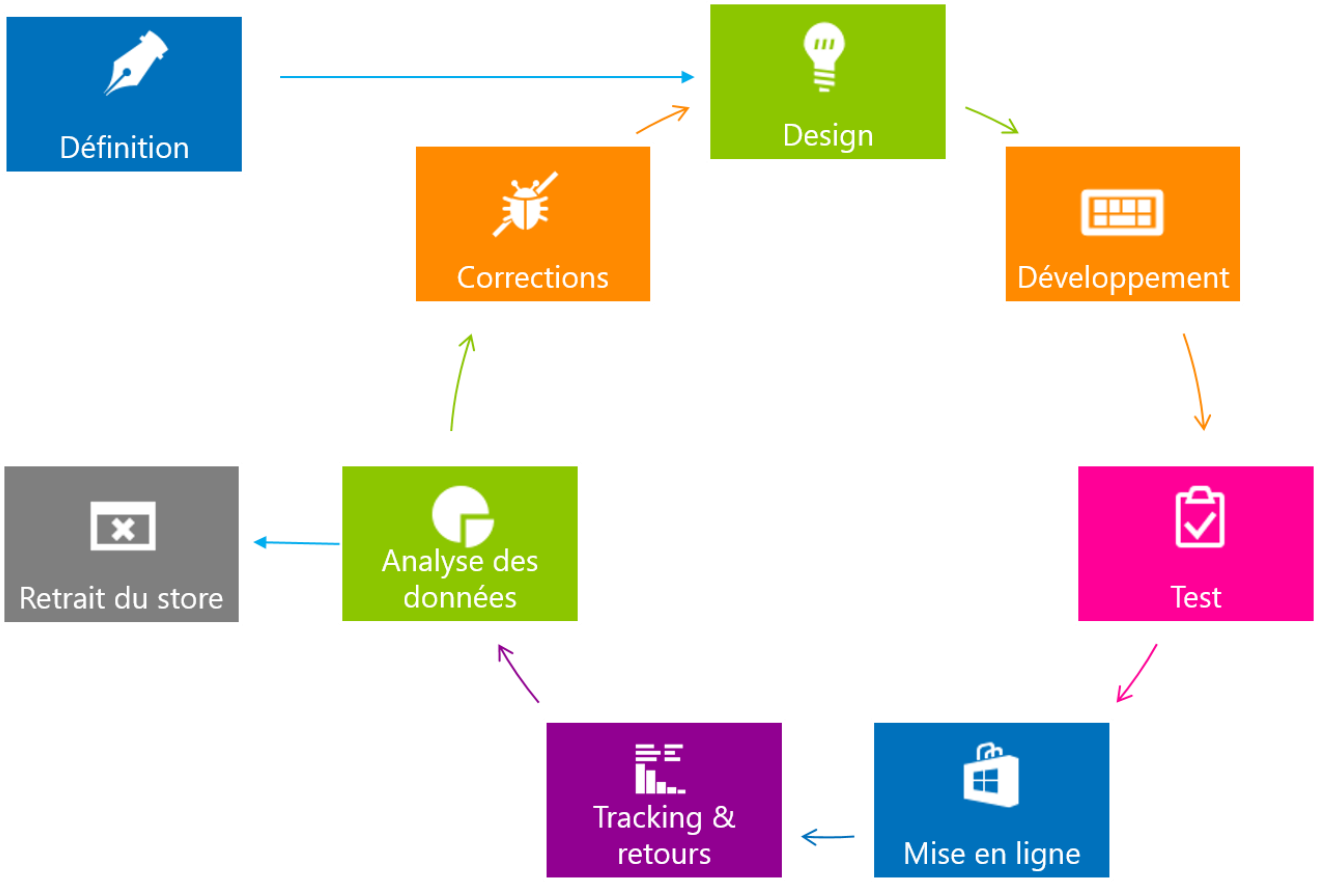
Les spécificités Android : accéder aux capteurs

Nom	Dimension du vecteur	Unité	Sémantique	Values[]
Accelerometer	3	m/s2	Mesure de l'accélération (gravité incluse)	[0] axe x [1] axe y [2] axe z
Gyroscope	3	Radian/seconde	Mesure la rotation en termes de vitesse autour de chaque axe	[0] vitesse angulaire autour de x [1] vitesse angulaire autour de y [2] vitesse angulaire autour de z
Light	1	Lux	Mesure de la luminosité	[0]valeur
Magnetic_Field	3	µTesla	Mesure du champ magnétique	[0] axe x [1] axe y [2] axe z
Orientation	3	degrés	Mesure l'angle entre le nord magnétique	[0] Azimut entre l'axe y et le nord [1] Rotation autour de l'axe x (-180,180) [2] Rotation autour de l'axe y (-90,90)
Pressure	1	KPascal	Mesure la pression	[0]valeur
Proximity	1	mètre	Mesure la distance entre l'appareil et un objet cible	[0]valeur
Temperature	1	Celsius	Mesure la température	[0]valeur



C’est différent du responsive
Pourquoi ?
Quid du cross platform ?

Cycle de vie d'une application



Démarche ?

- Une idée : répondre à un besoin ou en générer
- Des usages : vérifier l'intérêt ou le susciter
- Des utilisateurs satisfaits ou des applications Kleenex
- Des applications très orientées IHM :

Ce qui est l'atout c'est le front end,
des back end minimalistes

Tests unitaires, d'intégration d'usages et dispositifs

DEVELOPPEMENT

Avec quoi programme-t-on ?

Langages :

- JAVA pour le code
- XML pour l'interface
- Disponible sur Windows, MacOS, Linux…

- IDE : Android Studio

http://zenit.senecac.on.ca/wiki/index.php/APK

Les grandes étapes de développement

Ce n'est pas un simple programme java que l'on édite, compile et exécute.

De la configuration
http://developer.android.com/guide/topics/manifest/manifest-intro.html
Les configurations
- Configuration du projet
- Configuration de l'application

Des ressources
http://developer.android.com/guide/topics/resources/index.html
Gestions des ressources graphiques : texte, image, etc.

Du développement : mettre en place les vues et les interactions associées
(Activités, intents,…), d'éventuelles bases de données ou connexions à des serveurs distants.

Du déploiement : distribuer l'application sur les différents devices.
http://developer.android.com/distribute/index.html

http://designthing.net/the-basic-components-in-an-android-app/

Anatomie d'une application

Activité ⟺ une page de site web
une vue (généralement en xml)

Service : traitements lourds

Broadcast Receiver : Intercepteur d'évènements

Content Provider : surcouche à une base de données

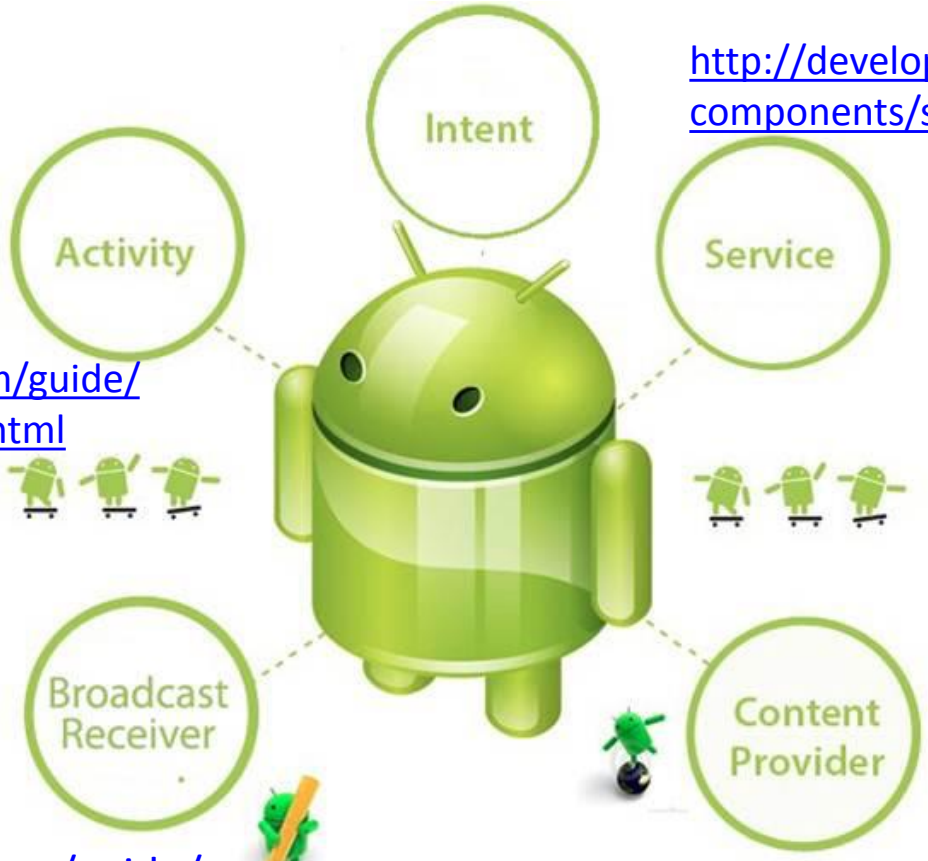
Intent : message envoyé au sein du système, communication inter composants
Peut contenir des données

/

Anatomie d'une application

http://developer.android.com/guide/components/intents-filters.html

http://developer.android.com/guide/components/services.html



https://developer.android.com/guide/components/activities/index.html

https://developer.android.com/guide/components/broadcasts.html

http://developer.android.com/guide/topics/providers/content-providers.html

Types de ressources

*res/values-**fr***

*res/drawable-**large***

*res/drawable-**hdpi***

res/layout-fr

*res/drawable-**small***

animator/	XML pour les propriétés animations
anim/	XML descriptifs des tween animations (avec transitions)
color/	XML pour la liste de couleurs
drawable/	Fichiers images (.png, jpg, gif) ou XML
mipmap/	Icone de lancement de différentes densités
layout/	XML de description de layouts
menu/	XML de descriptions de menus
raw/	Lié aux InputStream : de données brutes (musiques, html…)
values/	Pour les valeurs simples : Chaines, entiers, couleurs…(strings, colors, dimens, styles…)
xml/	Des fichiers XML utiles

Accès aux ressources

A la compilation, un fichier R est créé contenant des références sur les ressources de l'application.
C'est à l'aide de ce fichier qu'on peut accéder au ressources depuis du code.
Aperçu du contenu du fichier R.java :
Depuis le code, on peut alors accéder aux différentes ressources grâce à leur identifiant et ce fichier R.
Par exemple, pour accéder à du texte défini dans strings.xml, on appelle :
getString(**R.string. section_format**);

http://developer.android.com/guide/topics/resources/index.html

Gestion des ressources graphiques

Dossier drawable-XXXX : images
- XXHDPI (ultra haute densité)
- XHDPI (très haute densité)
- HDPI (haute densité)
- MDPI (moyenne densité)
- LDPI (basse densité)

Toujours essayer d'avoir les ressources dans toutes les densités
- meilleurs rendus
- scaling consommateurs de ressources (risque de OutOfMemoryException)

http://developer.android.com/guide/practices/screens_support.html

Manifest.xml

Configuration globale cœur du projet :

- déclaration des activités (écrans)

- paramètre des activités (ex : définir l'activité ou verrouiller une activité en mode paysage)

- déclaration des permissions de l'application (par exemple, permission d'accès à internet pour charger les images)

http://developer.android.com/guide/topics/manifest/manifest-intro.html

Gestion des accès

```
1 <activity android:name=".Activity1" android:label="@string/app_name">
2 <intent-filter>
3 <action android:name="android.intent.action.MAIN" />
4 <category android:name="android.intent.category.LAUNCHER" />
5 </intent-filter>
6 </activity>
```

```
1 <uses-feature android:name="android.hardware.bluetooth" />
2 <uses-feature android:name="android.hardware.camera" />
```

Gère l'accès au matériel (hardware requis)

Définit les composants de l'application ainsi que leurs propriétés (Activities, Services ...)

```
1 <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
2 <uses-permission android:name="android.permission.INTERNET" />
```

ACTIVITE

/

Anatomie d'une application

http://developer.android.com/guide/components/intents-filters.html

http://developer.android.com/guide/components/services.html

http://designthing.net/the-basic-components-in-an-android-app

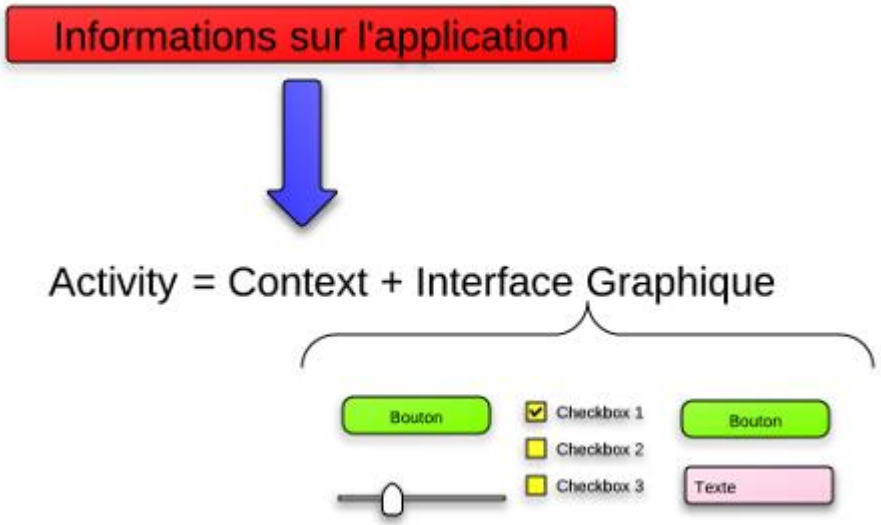
http://developer.android.com/guide/topics/providers/content-providers.html

Activité : en quelques mots

Une activité représente un écran avec son interface utilisateur.

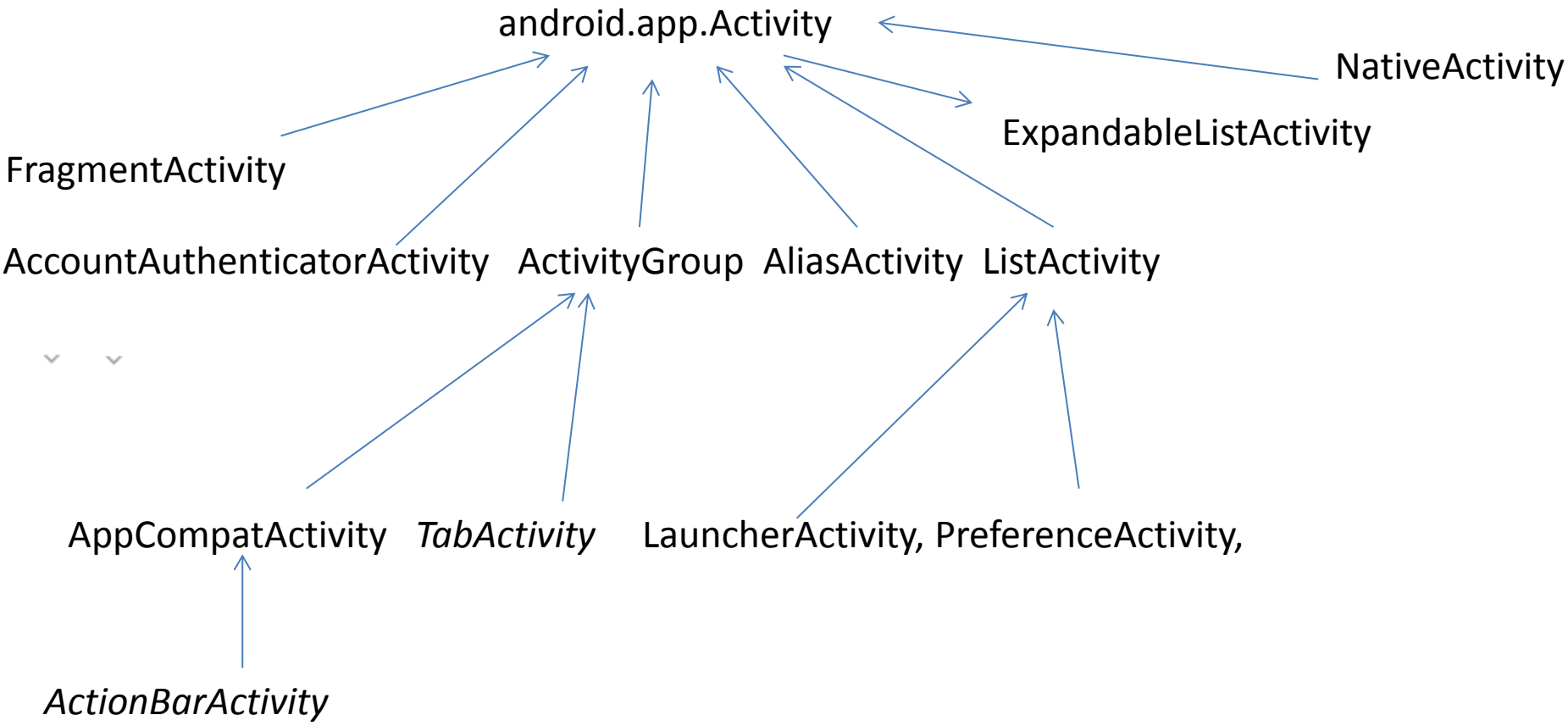
Par exemple dans Polynews, vous avez l'activité principale et une activité correspondant à chaque vue spécifique associée à un type d'actualités.

Bien que l'ensemble des activités réponde à un besoin commun qui est celui de l'application visée, chaque activité est indépendante.
Dans tous les cas il y a une activité de démarrage (classe main / run java) susceptible d'utiliser d'autres ativités.
Par exemple, vous pouvez faire démarrer une app de camera pour prendre une photo et contribuer à une actualité.



Toute activité est une sous classe ***d'Activity***
Qui hérite de l'interface *Context* pour représenter tous les composants d'une application. Dans le package **android.app.Activity.**.

Arbre d’héritage d’activités



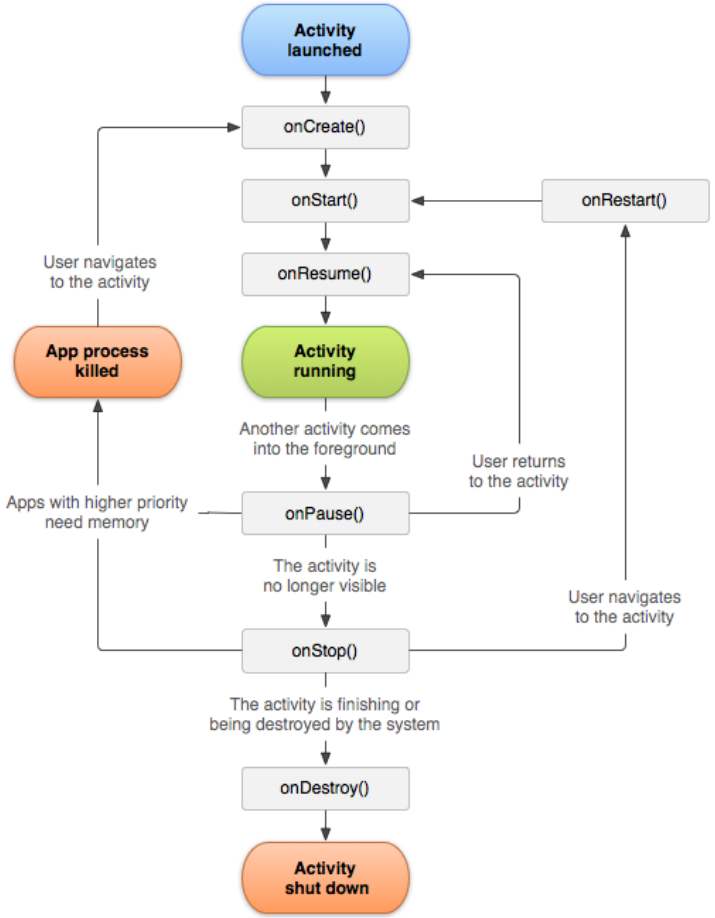
Cycle de vie d'une activité

Cycle de vie ***global***
onCreate() -> onDestroy()

Cycle de vie ***visible***
onStart() -> onStop()

Affichée à l'écran mais peut ne pas être utilisable
(en second plan)

- Cycle de vie ***en premier plan***
- onResume() -> onPause()



http://supertos.free.fr/supertos.php?page=1076

Implémentation d'une activité

Implementer les méthodes correspondant aux transitions entre différents états du cycle de vie.
Les deux plus importantes sont :

onCreate()
Appelée à la création de l'activité : intialiser les principaux composants. C'est là qu'on appelle **setContentView()** pour définir le layout de l'interface utilisateur de cette activité.

onPause()
appelée lorsque l'utilisateur quitte l'activité ce qui n'est pas toujours équivalent à détruire l'activité. Il faut penser à sauvegarder ce qui doit être persistant pour la prochaine session.

Implémenter les méthodes du cycle de vie permet de fournir une expérience utilisateur fluide.

https://openclassrooms.com/courses/creez-des-applications-pour-android/votre-premiere-application-1

Evénements et lien entre activités

Resumed

L'activité a le focus utilisateur.

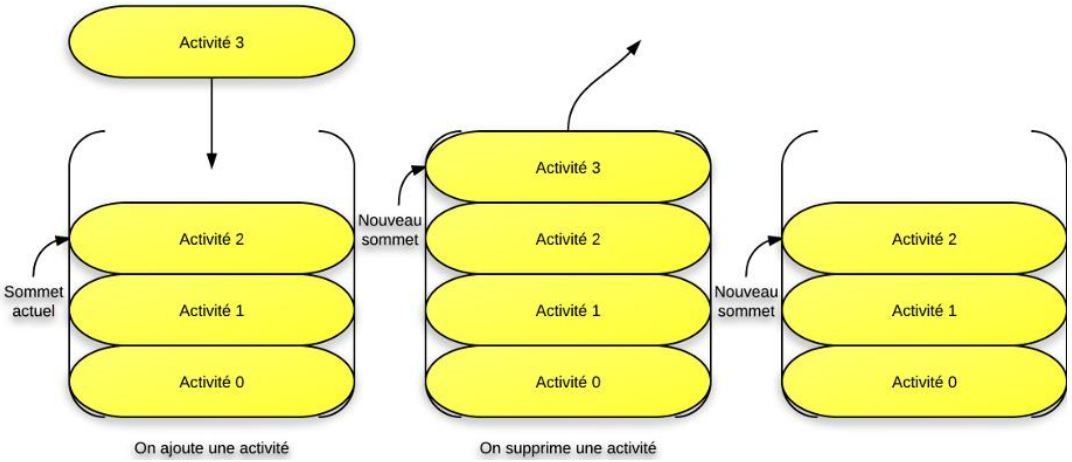
Paused

Une autre activité a le focus, mais celle-ci est toujours visible. L'activité est en mémoire et reste attachée au gestionnaire de fenêtre mais elle peut être tuée par le système si il devient trop lent.

Stopped

L'activité est en background. L'activité est en mémoire n'est pas attachée au gestionnaire de fenêtre. Elle n'est pas visible à l'utilisateur et peut être tuée par le système.

Gestion d'une Pile d'activités



Passage de données

Lorsque l'activité est détruite pour récupérer de la mémoire l'instance Activity est détruite et recréée si besoin.
Pour que l'état de l'activité soit préservée il faut implémenter la méthode onSaveInstanceState().

Cette méthode est appelée par le système avant une susceptible destruction.
Paramètre : un Bundle dans lequel sauver l'état avec des paires nom/valeur.

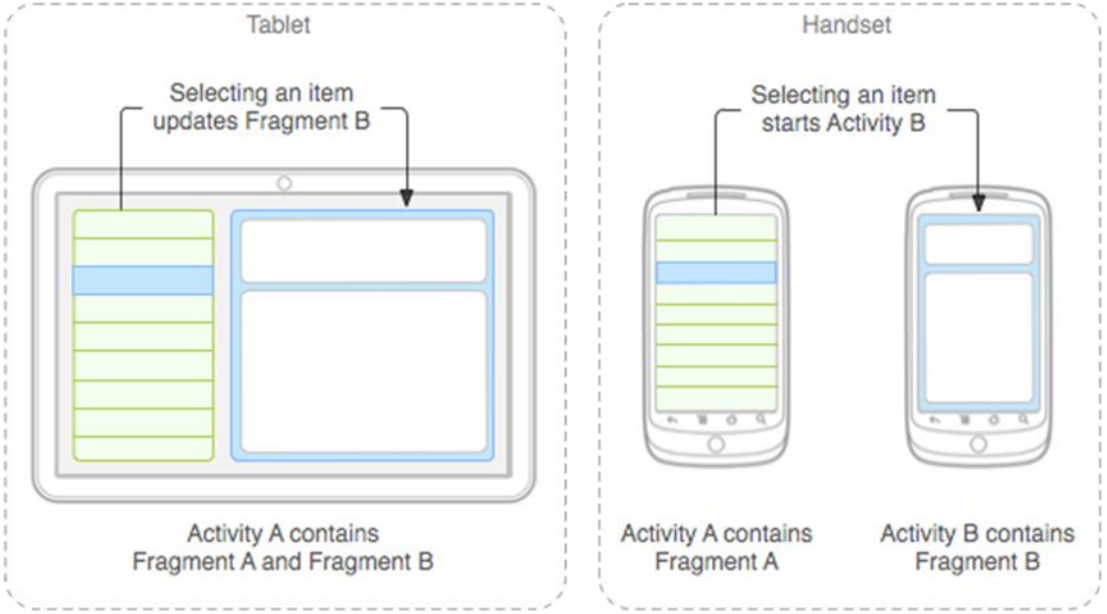
Le Bundle est ensuite passé en paramètre à onCreate() et onRestoreInstanceState().

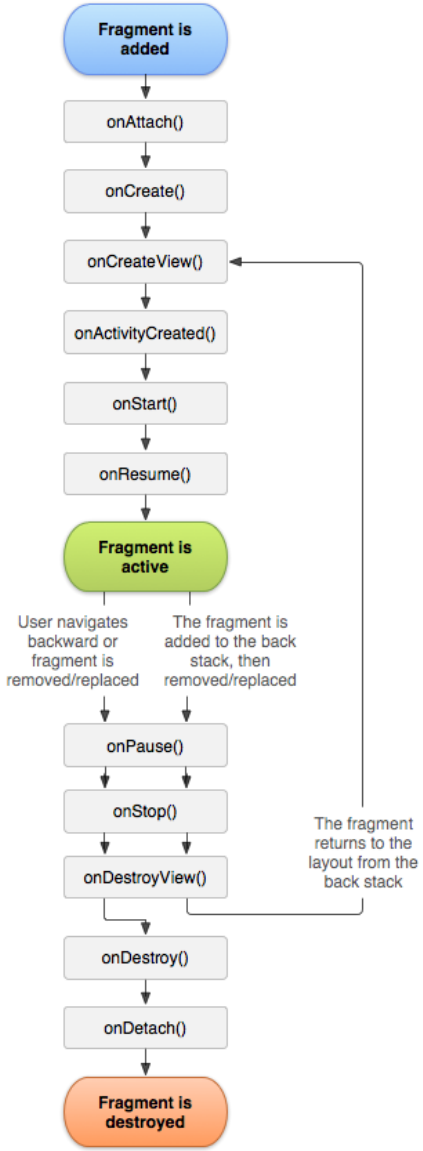
ATTENTION aux changements de configuration des devices à l'exécution (orientation, keyboard, language).
Ils impliquent un onDestroy(), suivi de onCreate().

FRAGMENT

Un fragment pourquoi ?

En Android, les fragments sont des portions d'interface graphiques. On peut combiner plusieurs fragments dans une activité et réutiliser les fragments dans différentes activités. Ils permettent donc un meilleur découpage ainsi qu'une meilleure évolutivité du code. La combinaison de fragments permet par exemple de créer des vues différentes selon le type d'appareil (tablette ou téléphone), sans avoir à dupliquer du code.





Un fragment a :
Son propre cycle de vie,
ses propres événements en entrée

on peut rajouter ou retirer un fragment d'une activité en cours d'exécution

Le cycle de vie d'un fragment est directement impacté par le cycle de vie de l'activité dans laquelle il se trouve,

Si activité est paused/destroyed alors tous les fragments sont paused/destroyed

Pendant qu'une activité est utilisée on peut manipuler chaque fragment indépendamment (les ajouter ou les supprimer)

http://developer.android.com/guide/components/fragments.html#Creating

http://designthing.net/the-basic-components-in-an-android-app/

Anatomie d'une application

Activité ⟺ une page de site web
une vue (généralement en xml)

Service : traitements lourds

Broadcast Receiver : Intercepteur d'évènements

Content Provider : surcouche à une base de données

Intent : message envoyé au sein du système, communication inter composants
Peut contenir des données

INTENTS

Pourquoi des Intents ?

Séparation forte entre chaque application : 1 application / 1 processus minimum
mais nécessité de communiquer entre applications et activités

Solution : envoi de messages synchrones
Intra application : facile, même espace mémoire
Inter-applications : IPC spécifique Android (AIDL)

Bienvenue dans le monde du réseau !

Objets messages pour faciliter la communication entre composants.

Pour démarrer une activité
Pour démarrer un service
Pour délivrer un broadcast

Types d'Intent

Intents explicites : interne à une app avec le nom complet de l'activité ou du service à appeler.
Par exemple pour démarrer une activité en fonction d'une action de l'utilisateur.

Intents implicites en déclarant une action générale qui est supportée par une autre app.
Par exemple pour montrer un emplacement sur une carte

```
<activity android:name="ShareActivity">
   <intent-filter>
      <action android:name="android.intent.action.SEND"/>
      <category android:name="android.intent.category.DEFAULT"/>
      <data android:mimeType="text/plain"/>
   </intent-filter>
</activity>
```

Pas conseillé pour les services à cause de problèmes de sécurité car sans retour utilisateur.

http://tutos-android-france.com/passer-des-donnees-entre-activites

Aperçu sur les AsyncTask vs les services

Même problématique : lancer des tâches en tache de fond pour faire des opérations longues et non IHM.
Cette prise en charge dans un Thread indépendant permet de ne pas affecter la qualité de l'interface (temps de réponse).

Lors du développement d'une application, il faut bien avoir en tête que toutes les tâches consommatrices de ressources (requêtes http, calculs lourds, …) doivent se faire dans un Thread séparé. En effet, le système affiche un message d'erreur et ferme l'application lorsque le Thread principal (appelé UI Thread) est bloqué trop longtemps.

Fonctionnement en asynchrone

Problème de priorité…

AsyncTasK

Pour réaliser des tâches de manière asynchrone, à la manière de la classe Thread de java.

Simple d'utilisation et d'implémentation
Le Thread secondaire est créé automatiquement et la communication entre les Thread est simplifiée.

Une Tâche Asynchrone hérite du la classe AsyncTask

Les trois paramètres attendus lors de la déclaration sont des types génériques :

- Le premier est le type des paramètres fournis à la tâche
- Le second est le type de données transmises durant la progression du traitement
- Enfin le troisième est le type du résultat de la tâche

Fonctionnement

Une AsyncTask doit obligatoirement implémenter la méthode *doInBackground.* C'est elle qui réalisera le traitement de manière asynchrone dans un Thread séparé.

Les méthodes *onPreExecute* (appelée avant le traitement), *onProgressUpdate* (appelée pour afficher sa progression) et *onPostExecute* (appelée après le traitement) sont optionnelles.

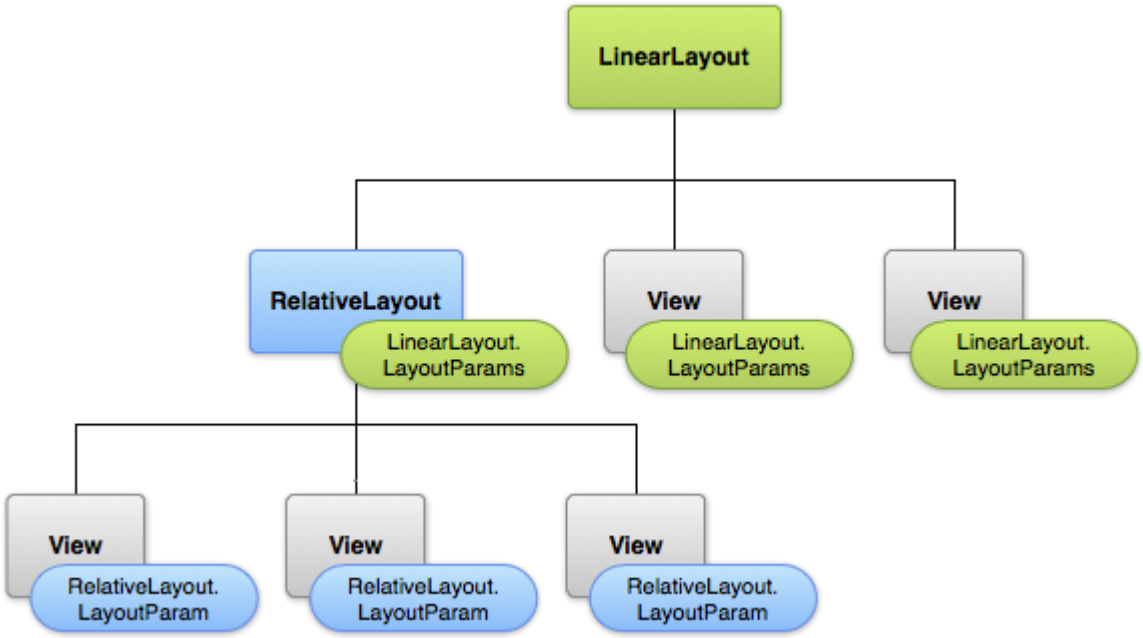
Un appel à la méthode *publishProgress* permet la mise à jour de la progression. On ne doit pas appeler la méthode *onProgressUpdate* directement.

Les trois méthodes (onPreExecute, onProgressUpdate et onPostExecute) s'exécutent depuis l'UI Thread ! C'est d'ailleurs grâce à cela qu'elles peuvent modifier l'interface. On ne doit donc pas y effectuer de traitements lourds.

http://www.tutos-android.com/asynctask-android-traitement-asynchrone-background

LAYOUT

Organisation générale d'une vue



Définition

Groupes de vues dérivés de *ViewGroup*
fournissent un modèle de présentation de leurs vues “filles”.

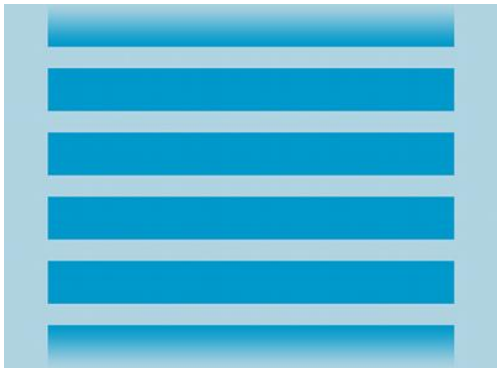
Vous pouvez aussi hériter de classes existantes pour créer votre propre layout et l’appliquer ensuite à votre activité.

Exemples de layout

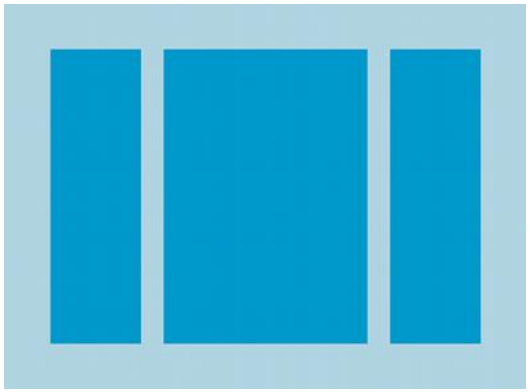
Conçus pour le Responsive design

- En lignes
- En colonnes
- En grille
- Relatif

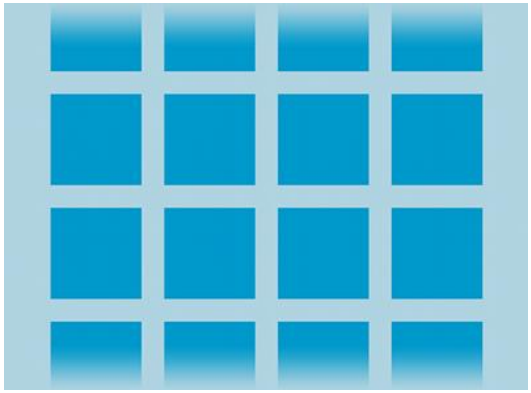
ListView



LinearLayout



GridView



RelativeLayout

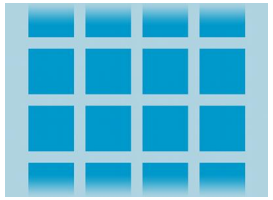
https://developer.android.com/guide/topics/ui/declaring-layout.html

Adaptateurs et Layout

Pour un contenu dynamique non prédéfini,
AdapterView (par spécialisation…) permet de placer les vues dans le layout à l'éxécution.
Il faut un Adaptateur pour relier les données au layout. .

Par exemple **ListView** permet d'avoir des items scrollables. Les items sont automatiquement insérés dans la liste via un **Adaptateur** qui met les items d'un tableau ou d'une base de données.

Ce n'est pas le seul Layout qui se construit avec des Adaptateurs regardez aussi GridLayout



Un peu plus sur les adaptateurs

ArrayAdapter si source de données tableau : utilise toString() pour chaque item et met le résultat dans un TextView.

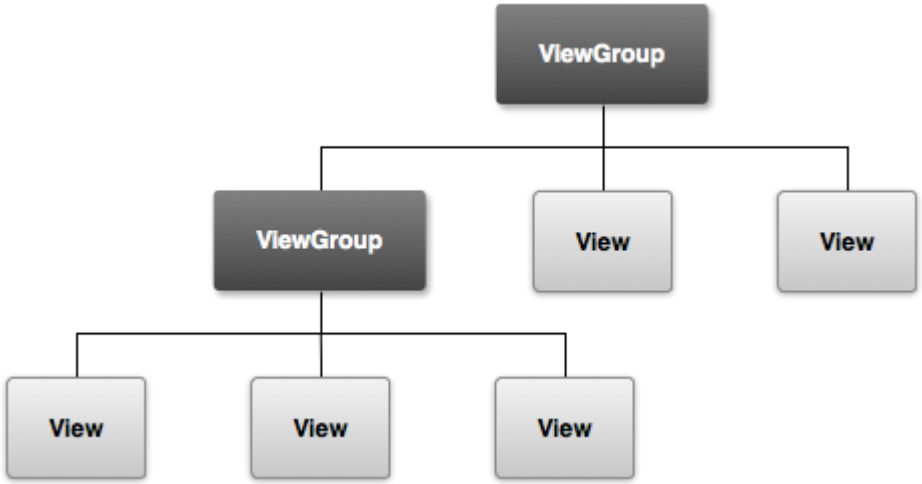
ArrayAdapter<String> adapter = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, myStringArray)
ListView listView = (ListView) findViewById(R.id.listview);
listView.setAdapter(adapter);

param 1 : contexte de l'appli
param 2 : layout pour 1 item
param 3 : le tableau de chaîne

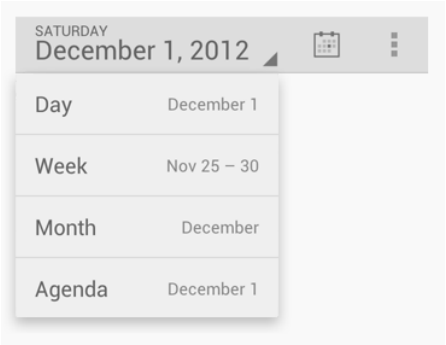
Pour faire votre propre visualisation de chaque item

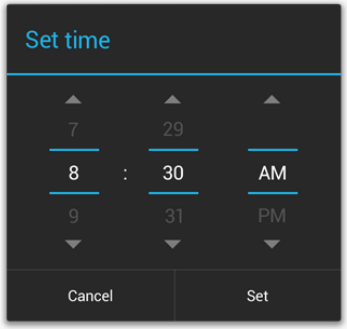
- surcharger le toString
- ou spécialiser ArrayAdapter et surcharge de *getView()* pour remplacer le TextView (par exemple, par un ImageView)

Arborescences de vues

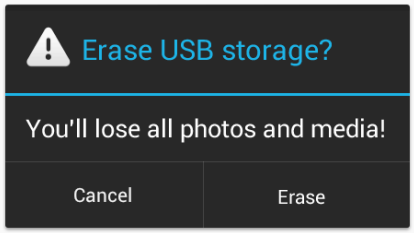


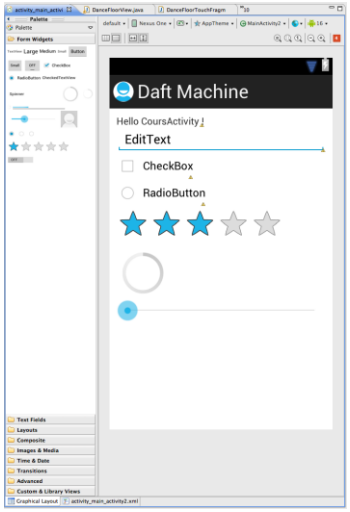
NE PAS CONFONDRE
HIERARCHIE DE COMPOSANTS
ET DE CLASSES

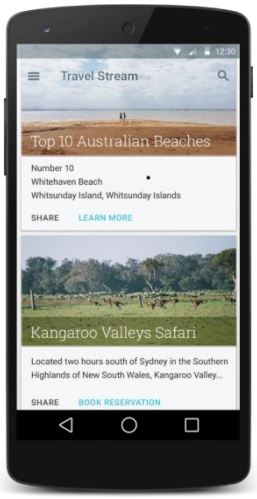

SATURDAY
December 1, 2012
Day December 1
Week Nov 25 – 30
Month December
Agenda December 1

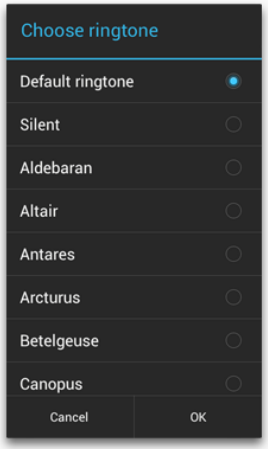

Set time
7 29
8 : 30 AM
9 31 PM
Cancel
Set

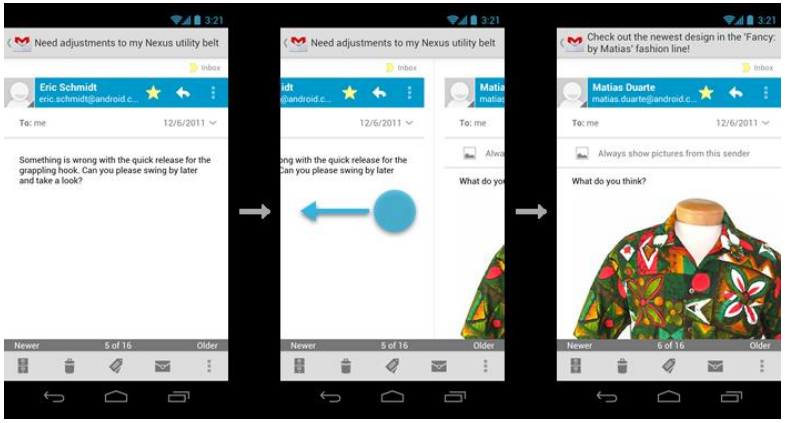

Alarm
Alarm

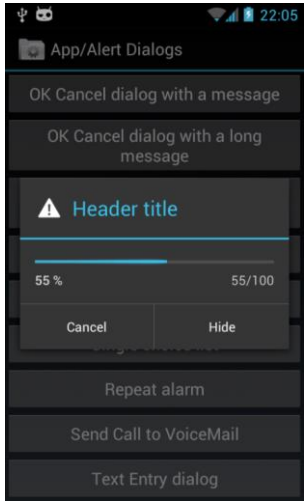

Erase USB storage?
You'll lose all photos and media!
Cancel
Erase

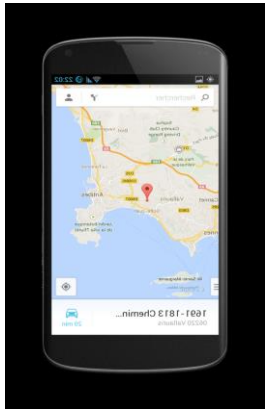

Daft Machine
Hello CoursActivity !
EditText
CheckBox
RadioButton

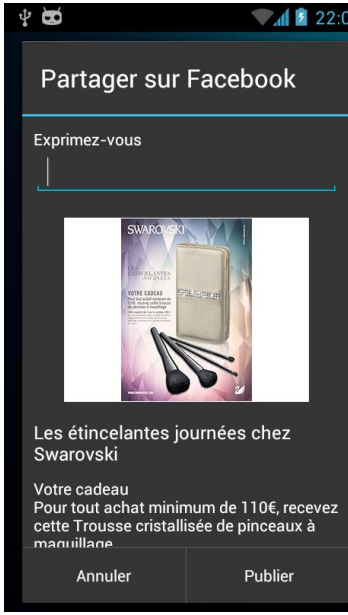

Travel Stream
Top 10 Australian Beaches
Number 10
Whitehaven Beach
Whitsunday Island, Whitsunday Islands
SHARE LEARN MORE
Kangaroo Valleys Safari
Located two hours south of Sydney in the Southern Highlands of New South Wales, Kangaroo Valley...
SHARE BOOK RESERVATION


Choose ringtone
Default ringtone
Silent
Aldebaran
Altair
Antares
Arcturus
Betelgeuse
Canopus
Cancel
OK


Need adjustments to my Nexus utility belt
Eric Schmidt
Something is wrong with the quick release for the grappling hook. Can you please swing by later and take a look?
Need adjustments to my Nexus utility belt
Check out the newest design in the 'Fancy: by Matias' fashion line!
Matias Duarte
What do you think?
Newer 5 of 16 Older


App/Alert Dialogs
OK Cancel dialog with a message
OK Cancel dialog with a long message
Header title
55 %
55/100
Cancel
Hide
Repeat alarm
Send Call to VoiceMail
Text Entry dialog




Partager sur Facebook
Exprimez-vous
SWAROVSKI
Les étincelantes journées chez Swarovski
Votre cadeau
Pour tout achat minimum de 110€, recevez cette Trousse cristallisée de pinceaux à maquillage
Annuler
Publier

Références

Fragments

http://developer.android.com/guide/components/fragments.html#Creating
http://mathias-seguy.developpez.com/tutoriels/android/comprendre-fragments/
http://fr.slideshare.net/CanElmas/android-working-with-fragments

Intents

http://developer.android.com/guide/components/intents-filters.html
http://tutos-android-france.com/passer-des-donnees-entre-activites/
http://cyrilmottier.com/2009/05/26/tutorial-android-4-les-intents/

Services

http://www-igm.univ-mlv.fr/~forax/ens/java-avance/cours/pdf/Android4-Intent-Service.pdf
http://developer.android.com/guide/components/services.html

Capteurs
https://openclassrooms.com/courses/creez-des-applications-pour-android/les-capteurs