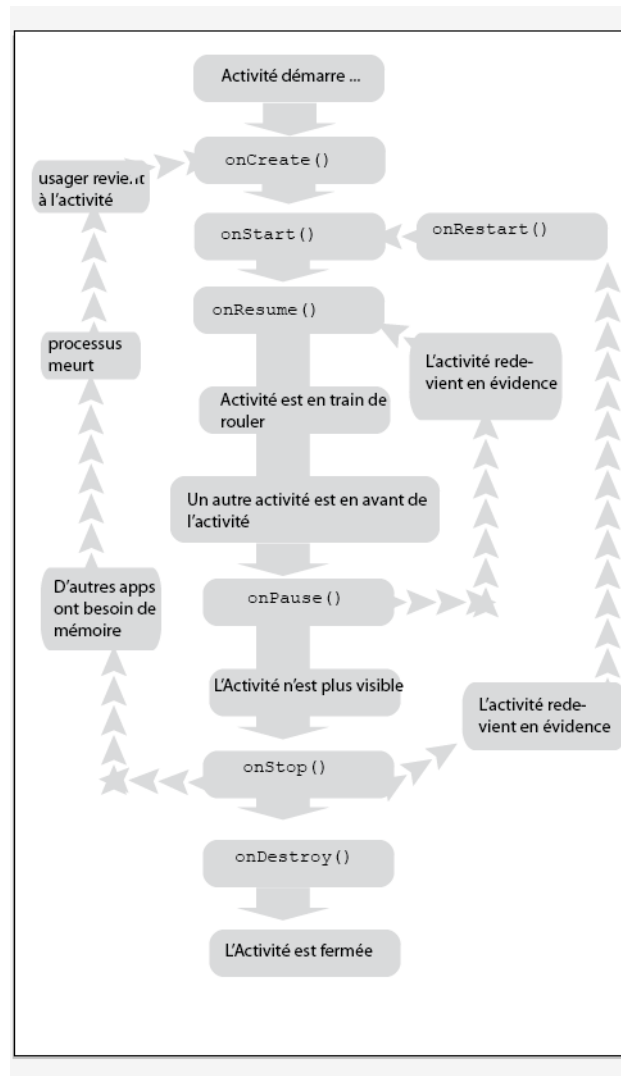


Annexe 4 – Cycle de vie / intentions

A) Cycle de vie d'une activité



vie entière :

vie fonctionnelle :

vie en évidence :

B) Intention (*Intent*)

Quand une application a plus qu'une activité, il est fort probable qu'on veuille pouvoir naviguer d'une activité à l'autre : on a alors besoin d'une intention (*intent*)

1. Créer un nouveau fichier de positionnement .xml dans le dossier layout

2. Créer une nouvelle classe, modifier méthode onCreate :

3. On doit ajouter la nouvelle activité au fichier AndroidManifest.xml :

```
<activity android :name=_____
           android :label=_____ >
    <intent-filter>
        <action android:name=_____/>
        <category android:name=_____/>
    </intent-filter>
</activity>
```

4. Créer l'intent : méthode startActivity

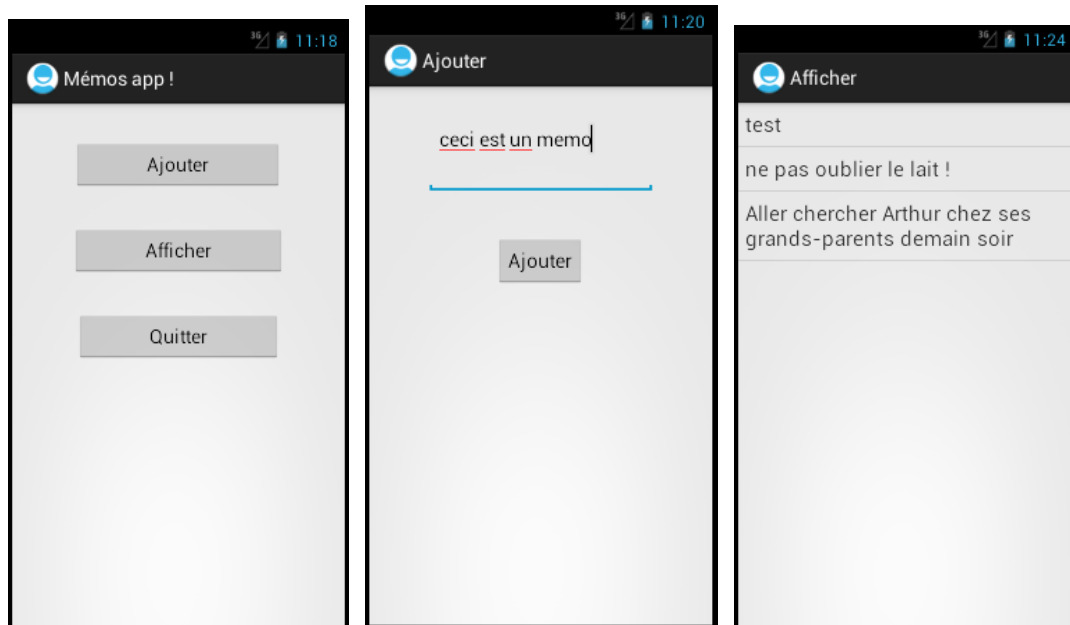
Passer des informations par l'intent

Méthodes à retenir


<code>putExtra (String, différents types (String, int, etc)</code>	Méthode de la classe <code>Intent</code> permettant d'ajouter des informations à l' <code>Intent</code> sous forme de paires clé-valeurs
<code>startActivity(Intent)</code>	Transmettre l'objet <code>Intent</code> à une autre activité
<code>getIntent()</code>	Méthode de la classe <code>Activity</code> permettant de retrouver l'objet <code>Intent</code> à la destination
<code>getStringExtra (String clé)</code>	Méthode de la classe <code>Intent</code> permettant de récupérer le l'information à partir de sa clé


Exercice : Faire une application comprenant 3 Activités permettant d'ajouter des notes/mémos.

- L'application possède 3 activités
 - o MainActivity (page d'accueil)
 - o AjouterActivity (page permettant l'ajout d'un mémo)
 - o ListeActivity (page permettant de consulter la liste des mémos)
- Évidemment, lorsque l'on quitte le programme, ces informations ne sont pas sauvegardées. Ceci fera partie du laboratoire suivant.





Activité 1 (Principal) :

A) Avec un `RelativeLayout`, quels attribut est nécessaire pour que les boutons soient centrés sur la largeur ? 


B) Quels attributs sont nécessaires pour que les boutons aient cette largeur ? 

Activité2 (Ajouter) :

A) Quel attribut XML permet de pouvoir entrer plusieurs lignes de texte dans un widget `EditText` ? 

Comment pouvoir stocker les mémos pour y avoir accès en tout temps ? 

Activité3 (Afficher) :

A) Quel genre d'adaptateur va pouvoir meubler le widget `ListView` ? 

C) Appeler des applications externes du téléphone à l'aide d'intentions

Le téléphone / AVD contient plusieurs applications de base qu'on peut utiliser à partir de nos propres applications (internet, géolocalisations, liste de contacts, le téléphone lui-même)

Les objets `Intent` qu'on doit créer dans ces situations sont constitués de deux paramètres la plupart du temps :

- Une Action : une constante représentant ce qu'on veut faire
- Des données (data) : les données qu'on a besoin de fournir afin de réaliser l'action ci-haut

Voici quelques couples action / données utiles; il en existe un très grand nombre dans les fichiers de l'API:

Type de l'action	Constante action	Données
Ouvrir une page web	<code>android.content.Intent.ACTION_VIEW</code>	<code>Uri.parse("http://...")</code>
Faire un appel téléphonique	<code>android.content.Intent.ACTION_DIAL</code>	<code>Uri.parse("tel:+789789")</code>
Faire afficher un endroit sur maps.google (l'AVD doit supporter Google	<code>android.content.Intent.ACTION_VIEW</code>	<code>Uri.parse("geo:0,0?q=ville, +province, +pays ")</code>

APIS)		
Choisir dans sa liste de contacts	<code>android.content.Intent.ACTION_PICK</code>	Pas de données à passer, on doit choisir un contact

On peut également passer les données nécessaires à la réussite de la demande à l'aide de la méthode `setData`.

On peut avoir à déterminer le type de données qu'on s'attend à recevoir suite à une intention. On peut le faire à l'aide de la méthode `setType`.

Ex. : `intention.setType (ContactsContract.Contacts.CONTENT_TYPE)` retournera les résultats (data) sous un type MIME permettant de regrouper toutes les infos d'un contact.



Exercice : Quoi faire aujourd'hui ?

1) Ajoutez 2 contacts à votre AVD.

→ lanceur d'applications, application Contacts puis suivez les directives sur le téléphone

2) Développez l'application ci-contre composée de 4 Buttons

3) Dans le fichier .java, gérez les événements `onClick` à l'aide de notre bonne vieille méthode **des 3 étapes**.

4) Faites en sorte qu'un clic sur le premier bouton fasse apparaître le site www.amazon.com

5) Faites en sorte qu'un clic sur le 2^{ème} bouton démarre un appel téléphonique

6) Faites en sorte qu'un clic sur le 3^{ème} bouton fasse apparaître une carte de la ville de Sorel-Tracy

7) Faites en sorte qu'un clic sur le dernier bouton permette d'afficher la liste de contacts présents sur cet AVD. Dans ce cas-ci, utilisez `startActivityForResult` de manière à récupérer le contact choisi par l'utilisateur dans votre application. Vous aurez besoin de récupérer les données choisies et de les afficher en créant un nouvel `Intent`.