



# Java et Android

## Activités multiples

# Sommaire

<a href="#">1.1 - Gérer une ou plusieurs sous-activités.....</a>	<a href="#">3</a>
<a href="#">1.1.1 - Présentation.....</a>	<a href="#">3</a>
<a href="#">1.1.1.1 - Objectif.....</a>	<a href="#">3</a>
<a href="#">1.1.1.2 - Gestion des appels aux sous-activités et des retours.....</a>	<a href="#">4</a>
<a href="#">1.1.1.3 - Syntaxe pour le fichier Manifest.....</a>	<a href="#">5</a>
<a href="#">1.1.1.4 - Syntaxes des appels.....</a>	<a href="#">6</a>
<a href="#">1.1.1.5 - Syntaxes pour la récupération de data dans une activité secondaire.....</a>	<a href="#">7</a>
<a href="#">1.1.1.6 - Syntaxe des retours dans l'activité secondaire.....</a>	<a href="#">8</a>
<a href="#">1.1.1.7 - Syntaxe des retours dans l'activité principale.....</a>	<a href="#">9</a>
<a href="#">1.1.1.8 - Démarche générale.....</a>	<a href="#">10</a>
<a href="#">1.1.2 - Le fichier /res/values/strings.xml.....</a>	<a href="#">11</a>
<a href="#">1.1.3 - Le fichier /res/values/arrays.xml.....</a>	<a href="#">11</a>
<a href="#">1.1.4 - Les layouts.....</a>	<a href="#">12</a>
<a href="#">1.1.4.1 - accueil_librairie.xml.....</a>	<a href="#">12</a>
<a href="#">1.1.4.2 - editeur_insert.xml.....</a>	<a href="#">13</a>
<a href="#">1.1.4.3 - auteur_insert.xml.....</a>	<a href="#">15</a>
<a href="#">1.1.5 - L'activité secondaire EditeurInsert.java.....</a>	<a href="#">17</a>
<a href="#">1.1.6 - L'activité secondaire AuteurInsert.java.....</a>	<a href="#">19</a>
<a href="#">1.1.7 - L'activité principale AccueilLibrairie.java.....</a>	<a href="#">20</a>
<a href="#">1.1.8 - Le fichier AndroidManifest.xml.....</a>	<a href="#">22</a>
<a href="#">1.1.9 - Exercice : gérer l'activité LivreInsert.....</a>	<a href="#">23</a>
<a href="#">1.2 - Multiples activités et cycle de vie.....</a>	<a href="#">24</a>
<a href="#">1.3 - Terminer une activité secondaire dans l'activité principale.....</a>	<a href="#">25</a>
<a href="#">1.4 - Modèle spécifique de page d'accueil : ListActivity.....</a>	<a href="#">26</a>
<a href="#">1.5 - Modèle spécifique de page d'accueil : Activity.....</a>	<a href="#">30</a>
<a href="#">1.6 - Modèle générique de page d'accueil.....</a>	<a href="#">32</a>
<a href="#">1.7 - Onglets et activités multiples.....</a>	<a href="#">35</a>
<a href="#">1.7.1 - Objectif.....</a>	<a href="#">35</a>
<a href="#">1.7.2 - Diagramme de navigation.....</a>	<a href="#">36</a>
<a href="#">1.7.3 - principal.xml.....</a>	<a href="#">37</a>
<a href="#">1.7.4 - expos.xml.....</a>	<a href="#">37</a>
<a href="#">1.7.5 - Principale.java.....</a>	<a href="#">38</a>
<a href="#">1.7.6 - Expos.java.....</a>	<a href="#">39</a>

## 1.1 - GÉRER UNE OU PLUSIEURS SOUS-ACTIVITÉS



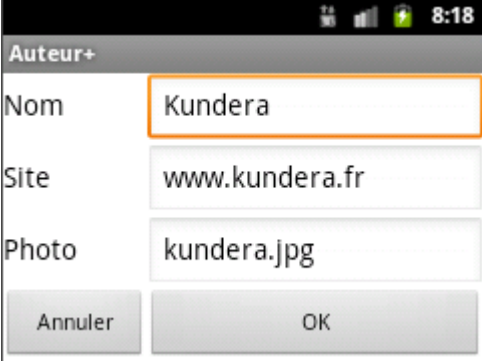
### 1.1.1 - Présentation

#### 1.1.1.1 - Objectif

Permettre de gérer des sous-activités.

L'intérêt est de diviser l'application en plusieurs activités soit pour des motifs de dimensions du terminal soit en fonction de la logique de l'application.

L'accueil, l'ajout d'un éditeur et l'ajout d'un auteur sont traités dans les paragraphes qui suivent. Vous traiterez l'activité Livre+ dans un exercice.

	
	
<p>Validation éditeur Gallimard</p>	<p>Annulation éditeur</p>

### 1.1.1.2 - Gestion des appels aux sous-activités et des retours

Les sous-activités sont gérées via des **Intents**.

Une intention est une description abstraite d'une opération à effectuer.

C'est un message ou un call entre un émetteur et un destinataire.

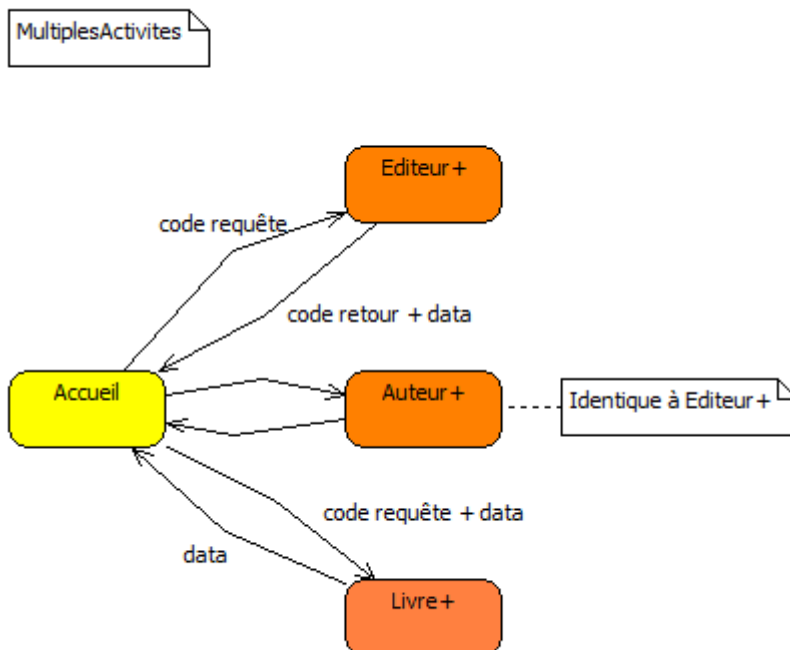
Les intentions permettent la communication entre les différentes activités de votre application mais aussi avec les activités du terminal Android (téléphonie, mail, ...) ainsi qu'avec les services et les broadcasters.

Elle peut être utilisée avec `startActivity()` pour lancer une activité, avec `broadcastIntent()` pour l'envoyer à un `BroadcastReceiver` (récepteur de message), et `startService()` ou `bindService()` pour communiquer avec un service d'arrière-plan.

L'objet de ce chapitre est de présenter la première utilisation.

Dans le cadre de la gestion des activités il existe plusieurs possibilités :

- ✓ Il n'y a qu'une seule activité secondaire,
- ✓ Il y a plusieurs activités secondaires et vous voulez connaître au retour vers l'activité principale la sous-activité qui a été sollicitée et éventuellement gérer un code retour (Validation/Annulation de l'opération),
- ✓ Vous voulez transmettre des données lors de l'appel d'une sous-activité et/ou récupérer des données au retour d'une sous-activité.



**Note :** gérer une activité de niveau 3 est possible (par exemple l'aide pour `AuteurInsert`) c'est exactement le même principe. L'activité de niveau 2 crée une `Intention` ...

#### 1.1.1.3 - Syntaxe pour le fichier Manifest

**Ajout d'un élément <activity>** à l'élément <application> dans le fichier AndroidManifest.xml.

```
<activity android:name=".Activité"></activity>
```

#### 1.1.1.4 - Syntaxes des appels

##### Déclaration d'un attribut Intent :

```
private Intent intention;
```

##### Instanciation d'une intention :

```
intention = new Intent(this.getContext(), ClasseActivité.class);
```

Les 2 paramètres du constructeur sont émetteur et récepteur.

##### La même chose en 2 temps :

```
Intent intention = new Intent();  
intention.setClass(this, ClasseActivité.class);
```

##### Appel d'une activité secondaire unique sans gestion de retour :

```
startActivity(intention);
```

##### Appel d'une activité secondaire avec gestion d'un code retour :

```
startActivityForResult(intention, code);
```

Le code permettra de connaître au retour l'activité secondaire qui a été appelée.  
La première intention aura le code 1, la deuxième le code 2, ...

##### Appel d'une activité secondaire avec envoi de données :

Ajout de données à une intention.

On ajoute une ou plusieurs paires de clé/valeur à l'objet Intent. Valeur peut être une String, un int, un parcelable, un tableau de String, un bundle, ...

```
Intention.putExtra("clé", valeur);
```

#### 1.1.1.5 - Syntaxes pour la récupération de data dans une activité secondaire

##### **Récupération dans l'appelé (activité secondaire) des données reçues.**

```
Bundle params = this.getIntent().getExtras();  
String valeur = params.getString("clé");
```

String ou tout autre type avec getXXX.

#### 1.1.1.6 - Syntaxe des retours dans l'activité secondaire

##### **Dans l'activité appelée (l'activité secondaire) :**

Renvoyer un résultat et terminer l'activité.

```
setResult(RESULT_OK | RESULT_CANCELED);  
finish();
```

Renvoyer un résultat, des données et terminer l'activité.

Ajout de données dans l'appelé pour les transmettre à l'appelant.  
On crée une intention "vide" pour pouvoir ajouter des données.

```
Intent intentionResultat = new Intent();  
intentionResultat.putExtra("clé", valeur);  
setResult(RESULT_OK | RESULT_CANCELED, intentionResultat);  
finish();
```



#### 1.1.1.7 - Syntaxe des retours dans l'activité principale

**Dans l'activité appelante (l'activité principale) :**

**Création d'une méthode nommée `onActivityResult()`**

Signature : `public void onActivityResult (int requestCode, int resultCode, Intent data)`

`requestCode` contient le code de l'activité appelante : 1, 2, ...

`resultCode` contient le code de retour : `RESULT_OK` ou `RESULT_CANCELED`.

`data` contient les données récupérables une à une avec `getStringExtra("clé")`.

#### 1.1.1.8 - Démarche générale

Il existe plusieurs possibilités (navigation explicite, navigation implicite).

Nous allons voir la navigation explicite dans ce paragraphe.

Ce qui est en rouge sera fait dans l'exercice.

- ✓ Création d'une nouvelle application nommée MultiplesActivites,
- ✓ La classe principale sera nommée AccueilLibrairie et le layout correspondant accueil\_librairie.xml; elle présente une liste (donc une ListView).
  
- ✓ Ajout des String pour les titres des activités dans le fichier strings.xml.
- ✓ Ajout des String pour les libellés des widgets,
- ✓ Ajout d'un tableau de String pour les items de la ListView de l'accueil.
  
- ✓ Création de 3 layouts : editeur\_insert.xml (\*), auteur\_insert.xml, livre\_insert.xml.
  
- ✓ Création d'une nouvelle classe de type Activity (EditeurInsert.java) (\*),
- ✓ Création d'une nouvelle classe de type Activity (AuteurInsert.java),
- ✓ Création d'une nouvelle classe de type Activity (LivreInsert.java),
  
- ✓ Référencement des activités dans le fichier AndroidManifest.xml (\*),
  
- ✓ Gestion des activités secondaires à partir de l'activité principale après avoir créé une Intention.
- ✓ Gestion des messages de retour et des data dans les activités secondaires.
- ✓ Gestion des messages de retour et des data dans l'activité principale.

(\*) cf note page suivante.

### 1.1.2 - Le fichier /res/values/strings.xml

Ajoutez les items en gras dans le fichier strings.xml pour personnaliser les titres de chaque activité.

```
<resources>
  <string name="app_name">MultiplesActivites</string>
  <string name="menu_settings">Settings</string>

  <string name="title_activity_accueil">Accueil</string>
  <string name="title_activity_editeur_insert">Editeur+</string>
  <string name="title_activity_auteur_insert">Auteur+</string>
  <string name="title_activity_livre_insert">Livre+</string>
  <string name="title_activity_aide_editeur_insert">Aide Editeur+</string>

</resources>
```

Ajoutez aussi les String suivantes au fichier strings.xml pour les libellés

```
<string name="annuler">Annuler</string>
<string name="ok">OK</string>
<string name="ko">KO</string>
<string name="nom">Nom</string>
<string name="site">Site</string>
<string name="editeur">Editeur</string>
<string name="auteur">Auteur</string>
<string name="livre">Livre</string>
<string name="titre">Titre</string>
<string name="photo">Photo</string>
<string name="aide">Aide</string>
<string name="message">Message</string>
```

### 1.1.3 - Le fichier /res/values/arrays.xml

Créez le fichier /res/arrays.xml s'il n'existe pas et ajoutez le tableau suivant dans le fichier arrays.xml pour les items de la ListView du menu d'accueil.

```
<string-array name="menu_general">
  <item>Editeur+</item>
  <item>Auteur+</item>
  <item>Livre+</item>
</string-array>
```

## 1.1.4 - Les layouts

### 1.1.4.1 - accueil\_librairie.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <ListView
        android:id="@android:id/list"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="60"
        android:entries="@array/menu_general" >
    </ListView>

    <TextView
        android:id="@+id/textViewMessageAccueil"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="20"
        android:text="@string/message"
        android:textAppearance="?android:attr/textAppearanceMedium" />

</LinearLayout>
```

## 1.1.4.2 - editeur\_insert.xml

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

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="30"
            android:text="@string/nom"
            android:textAppearance="?android:attr/textAppearanceMedium" />

        <EditText
            android:id="@+id/editTextNomEditeur"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="70"
            android:ems="10"
            android:inputType="textPersonName"
            android:text="Gallimard" >

            <requestFocus />
        </EditText>
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="30"
            android:text="@string/site"
            android:textAppearance="?android:attr/textAppearanceMedium" />

        <EditText
            android:id="@+id/editTextSiteEditeur"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="70"
            android:ems="10"
            android:text="www.gallimard.fr" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <Button
            android:id="@+id/buttonKOEditeur"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="30"
            android:text="@string/annuler" />

        <Button
```

```
        android:id="@+id/buttonOKEditeur"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="70"
        android:text="@string/ok" />
    </LinearLayout>

    <Button
        android:id="@+id/buttonAideEditeur"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/aide" />

</LinearLayout>
```

## 1.1.4.3 - auteur\_insert.xml

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

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="30"
            android:text="@string/nom"
            android:textAppearance="?android:attr/textAppearanceMedium" />

        <EditText
            android:id="@+id/editTextNomAuteur"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="70"
            android:ems="10"
            android:inputType="textPersonName"
            android:text="Kundera" >

            <requestFocus />
        </EditText>
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="30"
            android:text="@string/site"
            android:textAppearance="?android:attr/textAppearanceMedium" />

        <EditText
            android:id="@+id/editTextSiteAuteur"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="70"
            android:ems="10"
            android:text="www.kundera.fr" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <TextView
            android:id="@+id/textView3"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="30"
            android:text="@string/photo"
            android:textAppearance="?android:attr/textAppearanceMedium" />
```

```
<EditText
    android:id="@+id/editTextPhotoAuteur"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="70"
    android:ems="10"
    android:text="kundera.jpg" />
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >

    <Button
        android:id="@+id/buttonKOAuteur"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="30"
        android:text="@string/annuler" />

    <Button
        android:id="@+id/buttonOKAuteur"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="70"
        android:text="@string/ok" />
</LinearLayout>
</LinearLayout>
```



### 1.1.5 - L'activité secondaire EditeurInsert.java

La saisie d'un éditeur avec en retour le nom de l'éditeur saisi et le code retour correspondant à une validation ou à une annulation.

```
package fr.pb.multiplesactivites;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;

// -----
public class EditeurInsert extends Activity implements OnClickListener {
    private EditText editTextNomEditeur;
    private EditText editTextSiteEditeur;

    private Button buttonOKEditeur;
    private Button buttonKOEdeur;

    @Override
    // -----
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.editeur_insert);

        editTextNomEditeur = (EditText)findViewById(R.id.editTextNomEditeur);
        editTextSiteEditeur = (EditText)findViewById(R.id.editTextSiteEditeur);

        buttonOKEditeur = (Button)findViewById(R.id.buttonOKEditeur);
        buttonKOEdeur = (Button)findViewById(R.id.buttonKOEdeur);

        buttonOKEditeur.setOnClickListener(this);
        buttonKOEdeur.setOnClickListener(this);
    } // onCreate

    @Override
    // -----
    public void onClick(View v) {

        if(v == buttonOKEditeur) {

            // --- Une Intention vide
            Intent intentionResultat = new Intent();

            // --- L'ajout des valeurs a renvoyer
            intentionResultat.putExtra("nomEditeur",
editTextNomEditeur.getText().toString());

            // --- Affectation du code retour
            setResult(RESULT_OK, intentionResultat);

            // --- On clot l'activite
            finish();
        }

        if(v == buttonKOEdeur) {
            setResult(RESULT_CANCELED);
            finish();
        }
    }
}
```

```
        }  
    } /// onClick()  
} /// class EditeurInsert
```

### 1.1.6 - L'activité secondaire AuteurInsert.java

C'est quasiment le même code que pour l'activité EditeurInsert.

```
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;

// -----
public class AuteurInsert extends Activity implements OnClickListener {
    private EditText editTextNomAuteur;
    private EditText editTextSiteAuteur;
    private EditText editTextPhotoAuteur;

    private Button buttonOKAuteur;
    private Button buttonKOAuteur;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.auteur_insert);

        editTextNomAuteur = (EditText)findViewById(R.id.editTextNomAuteur);
        editTextSiteAuteur = (EditText)findViewById(R.id.editTextSiteAuteur);
        editTextPhotoAuteur = (EditText)findViewById(R.id.editTextPhotoAuteur);

        buttonOKAuteur = (Button)findViewById(R.id.buttonOKAuteur);
        buttonKOAuteur = (Button)findViewById(R.id.buttonKOAuteur);

        buttonOKAuteur.setOnClickListener(this);
        buttonKOAuteur.setOnClickListener(this);
    } /// onCreate

    @Override
    public void onClick(View v) {

        if(v == buttonOKAuteur) {
            // --- Une Intention vide
            Intent intentionResultat = new Intent();

            // --- L'ajout des valeurs a renvoyer
            intentionResultat.putExtra("nomAuteur",
editTextNomAuteur.getText().toString());

            // --- Affectation du code retour
            setResult(RESULT_OK, intentionResultat);

            // --- On clot l'activite
            finish();
        }
        if(v == buttonKOAuteur) {
            setResult(RESULT_CANCELED);
            finish();
        }
    } /// onClick()
} /// class AuteurInsert
```

### 1.1.7 - L'activité principale AccueilLibrairie.java

L'activité principale avec le menu (ListView) qui envoie des codes-paramètres et reçoit des codes et des data.

```
import android.os.Bundle;
import android.app.ListActivity;
import android.content.ActivityNotFoundException;
import android.content.Intent;
import android.view.View;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.TextView;

/*
 * AccueilLibrairie
 *
 */
public class AccueilLibrairie extends ListActivity {

    private TextView textViewMessageAccueil;

    private Intent intentionEditeurInsert;
    private Intent intentionAuteurInsert;

    @Override
    // -----
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.accueil_librairie);

        textViewMessageAccueil = (TextView)
        findViewById(R.id.textViewMessageAccueil);
    } // / onCreate

    // -----
    public void onItemClick(ListView parent, View v, int position, long id)
    {

        try {
            if (position == 0) {
                intentionEditeurInsert = new Intent(this,
                EditeurInsert.class);
                startActivityForResult(intentionEditeurInsert, 1);
            }

            if (position == 1) {
                intentionAuteurInsert = new Intent(this,
                AuteurInsert.class);
                startActivityForResult(intentionAuteurInsert, 2);
            }
        } // / try

        catch (ActivityNotFoundException e) {
            textViewMessageAccueil.setText(e.getMessage());
        }

    } // / onItemClick()

    // -----
    public void onActivityResult(int requestCode, int resultCode, Intent data) {
        switch (requestCode) {
            case 1: // Editeur
                switch (resultCode) {
                    case RESULT_OK:
```

```
        // --- Recuperation des donnees recues
        textViewMessageAccueil.setText("Validation éditeur " +
data.getStringExtra("nomEditeur"));
        return;
        case RESULT_CANCELED:
            textViewMessageAccueil.setText("Annulation éditeur");
            return;
        } // / switch (resultCode)

    case 2: // Auteur
        switch (resultCode) {
            case RESULT_OK:
                // --- Recuperation des donnees recues
                textViewMessageAccueil.setText("Validation auteur " +
data.getStringExtra("nomAuteur"));
                return;
            case RESULT_CANCELED:
                textViewMessageAccueil.setText("Annulation auteur");
                return;
        }
    } // / switch (requestCode)

} // / onActivityResult
} // / class AccueilLibrairie
```

### 1.1.8 - Le fichier AndroidManifest.xml

L'ajout des éléments <activity>.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="fr.pb.multiplesactivites"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="7"
        android:targetSdkVersion="15" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >

        <activity
            android:name=".AccueilLibrairie"
            android:label="@string/title_activity_accueil_multiple" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <activity
            android:name=".EditeurInsert"
            android:label="@string/title_activity_editeur_insert" >
        </activity>
        <activity
            android:name=".AuteurInsert"
            android:label="@string/title_activity_auteur_insert" >
        </activity>
    </application>

</manifest>
```

### 1.1.9 - Exercice : gérer l'activité LivreInsert

#### Objectif :

Ajouter un livre.

Après avoir éventuellement saisi un nouvel éditeur et/ou un nouvel auteur vous ajoutez un nouveau livre.

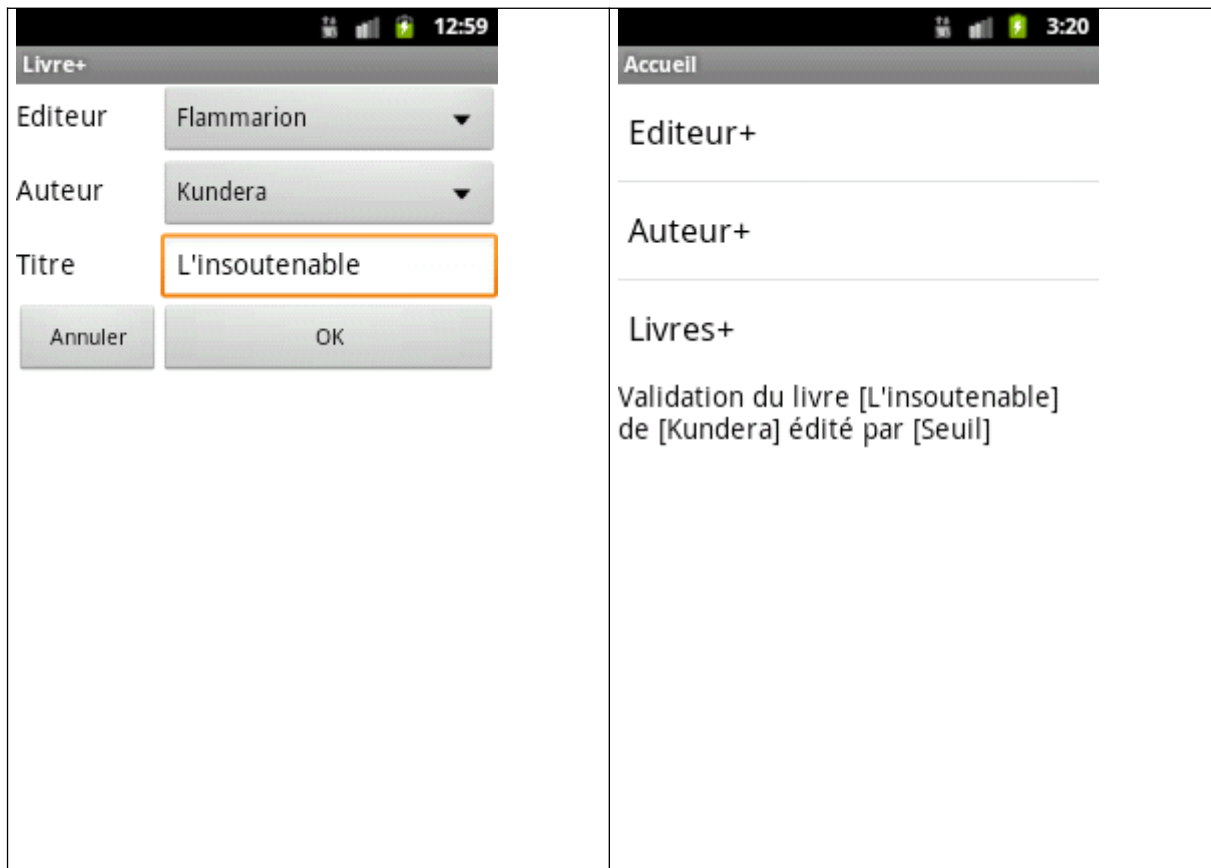
Les listes des éditeurs et des auteurs sont triées. Pour faire vos tests vous considérez que vous avez déjà trois auteurs et trois éditeurs.

Si un éditeur a été ajouté au préalable via l'activité secondaire EditeurInsert il est sélectionné.

Si un auteur a été ajouté au préalable via l'activité secondaire AuteurInsert il est sélectionné.

Au retour, dans l'activité Accueil, le titre du livre est affiché avec le nom de l'auteur et le nom de l'éditeur.

Dans un premier temps l'exercice peut être réalisé avec des EditText à la place des Spinners.



Au niveau applicatif il existe donc 4 cas :

- +Livre,
- +Auteur, +Livre,
- +Editeur, +Livre,
- +Editeur, +Auteur, +Livre.

## 1.2 - MULTIPLES ACTIVITÉS ET CYCLE DE VIE

Cf intro\_android\_java.odt pour la présentation du cycle de vie.

Il faut remarquer que, lorsque l'on passe de l'activité principale à une activité secondaire et vice-versa, par exemple lorsque l'on revient à l'activité principale, les méthodes `onStart()` et `onResume()` sont exécutées.



### 1.3 - TERMINER UNE ACTIVITÉ SECONDAIRE DANS L'ACTIVITÉ PRINCIPALE

Si vous appelez une activité secondaire personnelle qui n'a pas de code finish() vous devez terminer l'activité secondaire dans l'activité principale.

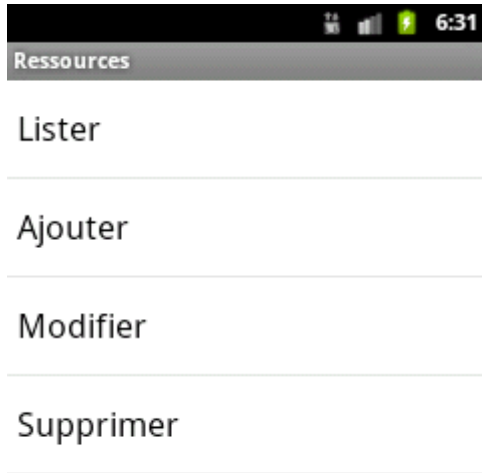
C'est grâce au requestCode que vous pouvez le faire.

Il en est de même avec les Intentions d'Android (RecognizerIntent, Dial, Phone, ...).

```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    // L'appel a été du type : startActivityForResult(intent, 1);
    if(requestCode == 1) {
        finishActivity(requestCode);
    }
    // Autre code
} /// onActivityResult
```

## 1.4 - MODÈLE SPÉCIFIQUE DE PAGE D'ACCUEIL : LISTACTIVITY

Une ListView



Toutes les classes utilisées sont référencées dans le Manifest.

Le tableau de Strings – dans /res/values/arrays.xml - est le suivant :

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string-array name="items_main">
        <item>Lister</item>
        <item>Ajouter</item>
        <item>Modifier</item>
        <item>Supprimer</item>
    </string-array>

</resources>
```

Le layout – main.xml - est le suivant :

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <ListView
        android:id="@android:id/list"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:entries="@array/items_main">
    </ListView>

</LinearLayout>
```

La classe Main.java est la suivante :

```
import android.os.Bundle;
import android.app.ListActivity;
import android.content.Intent;
import android.view.View;
import android.widget.ListView;
import android.widget.Toast;

public class Main extends ListActivity {

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

    @Override
    public void onListItemClick(ListView parent, View v, int position, long id)
    {

        Intent intention = null;

        switch (position) {
            case 0:
                intention = new Intent(this, Lister.class);
                break;
            case 1:
                intention = new Intent(this, Ajouter.class);
                break;
            case 2:
                intention = new Intent(this, Modifier.class);
                break;
            case 3:
                intention = new Intent(this, Supprimer.class);
                break;
            default:
                Toast.makeText(getBaseContext(), "Bizarre!!! ",
                    Toast.LENGTH_LONG).show();
        }

        if (position >= 0 && position <= 3) {
            startActivityForResult(intention, 1);
        }

    } // onListItemClick()

    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data) {
        finishActivity(requestCode);
    } // onActivityResult

} // Main
```

**Notes :**

La méthode `finishActivity()` est utilisée pour tuer l'activité appelée par le menu et qui ne possède pas de code de fin (pas d'appel à la méthode `finish()`).

Le code 1 utilisé avec la méthode `startActivityForResult()` est unique puisqu'il n'y a pas de code spécifique à exécuter sauf à tuer l'activité.

## 1.5 - MODÈLE SPÉCIFIQUE DE PAGE D'ACCUEIL : ACTIVITY

Une ListView



### arrays.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="menu">
        <item>Déplacement</item>
        <item>Rotation</item>
        <item>Dimensionnement</item>
        <item>Fondu</item>
        <item>Mélanges</item>
    </string-array>
</resources>
```

### main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <ListView
        android:id="@+id/listMenu"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:entries="@array/menu">
    </ListView>

</LinearLayout>
```

```
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.*;

public class Main extends Activity implements AdapterView.OnItemClickListener {

    private ListView listMenu;

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

        listMenu = (ListView) findViewById(R.id.listMenu);
        listMenu.setOnItemClickListener(this);

    } /// onCreate

    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long
id) {
        Intent intention = null;

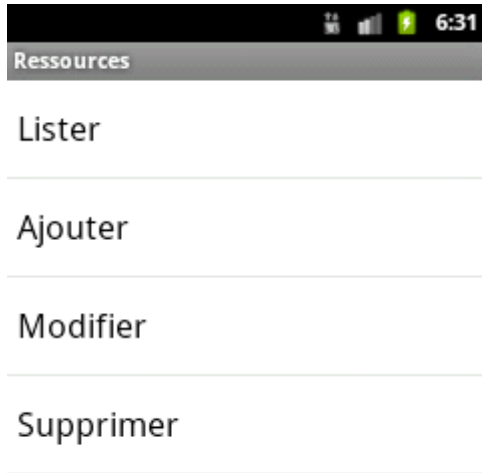
        switch (position) {
            case 0:
                intention = new Intent(this, Animations.class);
                intention.putExtra("type", "deplacement");
                break;
            case 1:
                intention = new Intent(this, Animations.class);
                intention.putExtra("type", "rotation");
                break;
            case 2:
                intention = new Intent(this, Animations.class);
                intention.putExtra("type", "dimensionnement");
                break;
            case 3:
                intention = new Intent(this, Animations.class);
                intention.putExtra("type", "fondu");
                break;
            case 4:
                intention = new Intent(this, Animations.class);
                intention.putExtra("type", "melanges");
                break;
        }
        startActivity(intention);
    } /// onItemClick

    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data) {
        finishActivity(requestCode);
    } /// onActivityResult

} /// class
```

## 1.6 - MODÈLE GÉNÉRIQUE DE PAGE D'ACCUEIL

Une ListView





Les items du menu **doivent être** identiques aux noms des classes (il serait possible de passer par un map pour avoir des items de menus plus « élégants » cf le support listes\_android\_java.odt). Les clés seraient les noms des classes et les valeurs les items affichés.

Toutes les classes utilisées sont référencées dans le Manifest.  
Toutes les classes sont dans le même package, le package principal.

Le tableau de Strings – dans /res/values/arrays.xml - est le suivant :

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string-array name="items_main">
    <item>Lister</item>
    <item>Ajouter</item>
    <item>Modifier</item>
    <item>Supprimer</item>
  </string-array>
</resources>
```

Le layout – main.xml - est le suivant :

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >

  <ListView
    android:id="@android:id/list"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:entries="@array/items_main">
  </ListView>

</LinearLayout>
```

La classe Main.java est la suivante :

```
import android.os.Bundle;
import android.app.ListActivity;
import android.content.Intent;
import android.view.View;
import android.widget.ListView;
import android.widget.Toast;

public class Main extends ListActivity {

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

    } // / onCreate

    @Override
    public void onItemClick(ListView parent, View v, int position, long id)
    {

        String lsSelection = parent.getItemAtPosition(position).toString();

        Intent intention = null;

        Class c = null;

        try {
            c = Class.forName(this.getPackageName() + "." + lsSelection);
            intention = new Intent(this, c);
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
            Toast.makeText(this, "Erreur : " + e.getMessage(),
                           Toast.LENGTH_LONG).show();
        }

        if (c != null) {
            this.startActivityForResult(intention, 1);
        }
    } // / onItemClick()

    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data) {
        finishActivity(requestCode);
    } /// onActivityResult

} // / Accueil
```

#### Notes :

La méthode finishActivity() est utilisée pour tuer l'activité appelée par le menu et qui ne possède pas de code de fin (pas d'appel à la méthode finish()).

Le code 1 utilisé avec la méthode startActivityForResult() est unique puisqu'il n'y a pas de code spécifique à exécuter sauf à tuer l'activité.

## 1.7 - ONGLETS ET ACTIVITÉS MULTIPLES

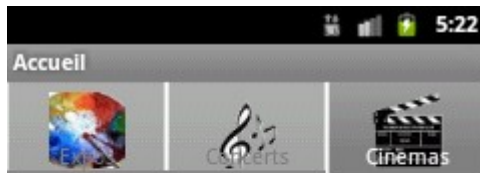
<http://developer.android.com/reference/android/app/TabActivity.html>

This class was **deprecated** in API level 13.

New applications should use **Fragments** instead of this class; to continue to run on older devices, you can use the v4 support library which provides a version of the Fragment API that is compatible down to DONUT.

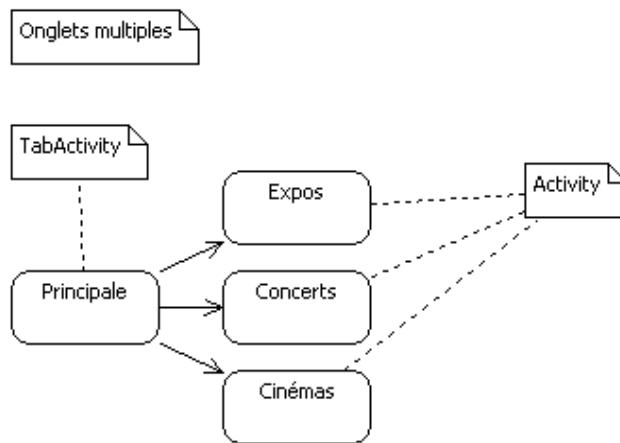
### 1.7.1 - Objectif

Pour ne pas coder le code de toutes les pages d'onglet dans une seule activité et un seul layout, nous allons coder dans plusieurs activités.



CINEMAS

### 1.7.2 - Diagramme de navigation



### 1.7.3 - principal.xml

```
<?xml version="1.0" encoding="utf-8"?>
<TabHost xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@android:id/tabhost"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <TabWidget
            android:id="@android:id/tabs"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

        <FrameLayout
            android:id="@android:id/tabcontent"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"/>
        <!-- Dans ce cas, avec les onglets en bas il faut dimensionner la Frame
        layout autrement (en dp, en %, ...) -->

    </LinearLayout>

</TabHost>
```

### 1.7.4 - expos.xml

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

    <!-- expos -->
    <TextView android:text="EXPOS"
        android:padding="15dip"
        android:textSize="18dip"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

</LinearLayout>
```

Idem pour concerts.xml et cinemas.xml.

### 1.7.5 - Principale.java

```
import android.app.TabActivity;
import android.content.Intent;
import android.os.Bundle;
import android.widget.TabHost;
import android.widget.TabHost.TabSpec;

/*
 * TabHosts et multiples activites
 */
public class Principale extends TabActivity {

    // -----
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.principal);

        // Les autres activites
        Intent exposIntent = new Intent(this, Expos.class);
        Intent concertsIntent = new Intent(this, Concerts.class);
        Intent cinemasIntent = new Intent(this, Cinemas.class);

        // L'onglet
        TabHost tabHost = getTabHost();

        // Les pages d'onglet
        // Tab Expos
        // Un id
        TabSpec exposSpec = tabHost.newTabSpec("expos");
        // Le titre et l'icone
        exposSpec.setIndicator("Expos",

getResources().getDrawable(R.drawable.ic_action_search));
        // Le contenu
        exposSpec.setContent(exposIntent);

        // Tab Concerts
        TabSpec concertsSpec = tabHost.newTabSpec("concerts");
        concertsSpec.setIndicator("Concerts",

getResources().getDrawable(R.drawable.ic_action_search));
        concertsSpec.setContent(concertsIntent);

        // Tab Cinemas
        TabSpec cinemasSpec = tabHost.newTabSpec("cinemas");
        cinemasSpec.setIndicator("Cinemas",

getResources().getDrawable(R.drawable.ic_action_search));
        cinemasSpec.setContent(cinemasIntent);

        // Ajout des TabSpec au TabHost donc des pages d'onglet a l'onglet
        tabHost.addTab(exposSpec); // Ajout tab
        tabHost.addTab(concertsSpec);
        tabHost.addTab(cinemasSpec);
    } // / on Create
} // / class Principale
```

### 1.7.6 - Expos.java

```
import android.app.Activity;
import android.os.Bundle;

// -----
public class Expos extends Activity
{
    // -----
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.expos);
    } /// onCreate
} /// class
```

Idem pour Concerts.java et Cinemas.java.

Et sans oublier les références aux activités dans AndroidManifest.xml

```
<activity android:name=".Expos" ></activity>
<activity android:name=".Concerts" ></activity>
<activity android:name=".Cinemas" ></activity>
```