

Activités

Chantal Keller

4 septembre 2015

Plan

1 Questions

2 Activités

3 Théorie

CyanogenMod

- Fork d'Android
- Extension de la partie open source
- Supporte les applications Android n'utilisant pas les parties propriétaires (pilotes)

Plan

1 Questions

2 Activités

3 Théorie

Rappels

Application = ensemble d'*activités* :

- Une activité par page
- Interface en XML
- Interactions en Java

Interaction :

- Utilisateur : entrée de texte, clic, ...
- Environnement : création et destruction d'activité, mise en pause...

Activité, côté Java

La classe *Activity* :

- Toutes les activités en héritent (éventuellement, via plusieurs niveaux d'héritage)
- Définit des méthodes appelées lors de la création, la destruction, ... d'une activité
- Méthodes à surcharger pour chaque activité

Création d'une activité

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_td1);  
    System.out.println("Activity created");  
}
```

- Appel de la méthode de la super-classe
- Création de l'interface graphique
- Autres calculs (notamment, récupération des objets de l'interface)

Activité, côté XML

Description de l'interface en XML :

- Mieux adapté que Java
- Partie statique

Extensible Markup Language

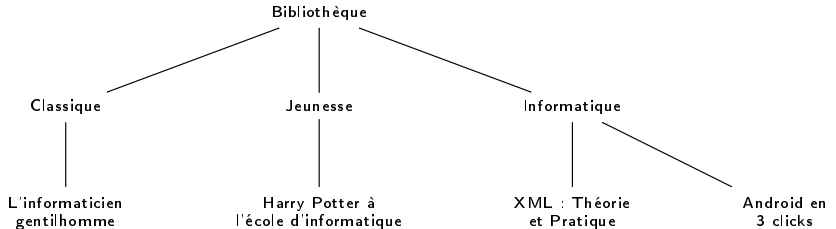
XML :

- Langage extensible
- Description d'objets structurés, notamment d'arbres
- Ex : XHTML

Principe :

- Chaque nœud est représenté par des balises (ouvrante et fermante)
- Les fils sont des nœuds entre la balise ouvrante et la balise fermante
- Chaque nœud peut avoir des attributs

Exemple



```
<bibliothèque genre="geek">
  <theme genre="Classique">
    <livre titre="L'informaticien gentilhomme" auteur="Lomière" />
  </theme>
  <theme genre="Jeunesse">
    <livre titre="Harry Potter à l'école d'informatique" auteur="K.J. Lowring" />
  </theme>
  <theme genre="Informatique">
    <livre titre="XML : Théorie et Pratique" auteur="Foobar" />
    <livre titre="Android en 3 clicks" auteur="T. Toto" />
  </theme>
</bibliothèque>
```

Une interface est un arbre

Interface :

- “Boîtes” imbriquées les unes dans les autres
- Pour Android, ces boîtes sont appelées des *vues*

Exemple

```
<RelativeLayout
    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:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".TD1Activity">

    <TextView
        android:text="@string/hello_world"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

RelativeLayout

TextView

Vues, côté XML

Les boîtes :

- *ViewGroup (layout)* : contiennent d'autres boîtes ; le choix du *layout* détermine la position de ses fils
- Autres : zones de texte, boutons, images, ...

LinearLayout

Aligne les fils dans une même direction

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

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

    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="A" />

    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="B" />
  </LinearLayout>

  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="C" />
</LinearLayout>
```



RelativeLayout

Place les fils les uns par rapport aux autres

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_alignParentTop="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="A"
        android:id="@+id/a" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="B"
        android:id="@+id/b"
        android:layout_toRightOf="@id/a"
        android:layout_toEndOf="@id/a" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="C"
        android:id="@+id/c"
        android:layout_below="@id/a" />
</RelativeLayout>
```



GridLayout

Place les objets sur une grille

```
<GridLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="A"
        android:layout_row="0"
        android:layout_column="0" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="B"
        android:layout_row="0"
        android:layout_column="1" />

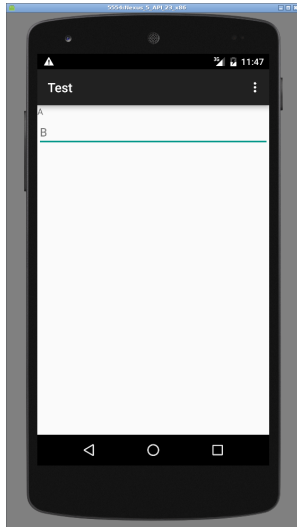
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="C"
        android:layout_row="1"
        android:layout_column="0" />
</GridLayout>
```



Zone de texte

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="A" />

<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:hint="B" />
```



Boutons

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="OK" />

<RadioButton
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Un et un seul" />

<CheckBox
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Facultatif" />

<Switch
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Switch" />
```



Boutons

```
<Button  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="OK"  
    android:onClick="whatToDoOnClick" />
```

```
<RadioButton  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Un et un seul" />
```

```
<CheckBox  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Facultatif" />
```

```
<Switch  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Switch" />
```



Vues, côté Java

```
public void whatToDoOnClick(View view) {  
    System.out.println("Clic!");  
}
```

Interaction :

- Méthode à exécuter lorsque l'utilisateur fait une action
- Prend comme argument la vue à laquelle l'action s'applique
- La classe *View* : toutes les vues en hérite

Objets de l'interface

```
private TextView a, b;

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_td1);
    a = (TextView) findViewById(R.id.a);
    b = (TextView) findViewById(R.id.b);
}

public void whatToDoOnClick(View view) {
    a.setText("B");
    b.setText("A");
}
```

- Récupération des objets de l'interface au moment de la création de l'activité, **après avoir lancé la création de l'interface**
- Utilisation quand on veut

Plan

1 Questions

2 Activités

3 Théorie

Historique

En Java “standard” :

- Dans le J2SE (Java Standard Edition) : AWT (Abstract Window Toolkit) et Swing
- Autres : SWT (Standard Widget Toolkit) et JFace

Android remplace AWT et Swing par ses propres bibliothèques.

Modèle-vue-contrôleur

Architecture MVC :

- Modèle : gestion des données
- Vue : gestion de l'interface
- Contrôleur : interaction avec l'utilisateur

En Android :

- Modèle : `strings.xml`, voir le cours sur la persistance des données
- Vue : interface XML
- Contrôleur : interaction en Java

À vous

- Création d'interfaces utilisant les différents *layouts*, des zones de texte et des boutons
- Interaction avec l'utilisateur