Android: Widgets & Layout

Assane SECK
Ingénieur-Informaticien

Sommaire

Widgets & Layout

Utiliser l'XML pour le layout
Labels, boutons & champs
LinearLayout
RelativeLayout
TableLayout
ScrollView

UTILISER DE L'XML POUR LE LAYOUT

- Il est possible d'instancier ses composants depuis Java, mais l'XML est une approche commune.
- Les layouts sont définis dans res/layout/
- L'XML contient un arbre d'éléments, qui spécifient le layout et les widgets qui forment la View.
- Les éléments XML contiennent des attributs, souvent avec le namespace android
- Exemple: **android:textStyle="bold"** sur une View **Button**

POURQUOI DES LAYOUTS XML?

- GUI builder dans Eclipse
- Les données sont structurées et faciles à lire : maintenance facile
- Separation de IHM et code
- L'XML est souvent utilisé pour définir des IHM
- Microsoft XAML, Mozilla XUL, Adobe Flex MXML,...
- Recommandation du W3C, Facilite l'interopérabilité des OS

CHOSES SPECIFIQUES XML

- **android:id="@+id/myId"** assigne myId comme id d'un widget ou layout
- References aux resources, strings (chaines) et autres id:

  @drawable/myImage pointera vers myImage.png dans res/drawable actuellement **@mipmap/myImage**
- **@string/app_name** pointera vers **app_name** dans res/values/strings.xml
- **@id/otherId** met une référence vers **otherId**

mipmap disponible depuis l'API 17

Sommaire

Widgets & Layout

Utiliser l'XML pour le layout
Labels, boutons & champs
LinearLayout
RelativeLayout
TableLayout

LABELS: TEXTVIEW

- **TextView** widget
- propriétés souvent utilisées:
- **android:text**
- **android:typeFace (**normal, sans, serif, monospace)
- **android:textStyle** (**bold**, **italic** or **bold_italic**)

```
<TextView
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Bonjour tout le monde !"
   />
```

Depuis l'API 8 arrive match_parent jouant le même role que fill_parent

BUTTONS

- **Button** widget
- Hérite de **TextView**

```
<Button
xmlns:android="http://schemas.android.com/apk/res/android"
android:id="@+id/button"
android:text="Valider"
android:layout_width="fill_parent"
android:layout_height="wrap_content"/>
```

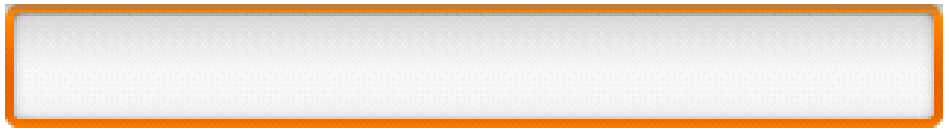
CHAMPS : EDITTEXT

- Widget **EditText** Hérite de **TextView**
- Propriétés:
- android:autoText
- android:capitalize
- android:maxLength
- android:minLine
- android:maxLine
- android:maxLength
- android:password
- android:inputType (number, phone,...)
- android:hint

```
<EditText

xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/field"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:inputType="text"

/>
```



EditText: Valeur inputType

- Pour le widget "**EditText**", avec l'option **android:inputType**, voici les différentes valeurs disponibles :
- **text** (par défaut) : clavier normal
- **textCapCharacters** : Clavier tout en majuscule
- **textCapWords** : Première lettre automatiquement en majuscule
- **textCapSentences** : Phrase en majuscule
- **textAutoCorrect** : Activer la correction automatique
- **textAutoComplete** : Activer la correction automatique forcée
- **textMultiLine** : Texte sur plusieurs lignes
- **textNoSuggestions** : Pas de suggestion de correction
- **textUri** : Saisie d'une url web
- **textEmailAddress** : Adresse mail
- **textEmailSubject** : Sujet de mail
- **textShortMessage** : Active le raccourci smiley sur le clavier
- **textPersonName** : Saisie du nom d'une personne (affichage de speach to text en bas a gauche du clavier)
- **textPostalAddress** : Saisie d'une adresse postale (affichage de speach to text en bas a gauche du clavier)
- **textPassword** : Saisie de mot de passe
- **textVisiblePassword** : Saisie d'un mot de passe visible
- **textWebEditText** : Text Web (activation de raccourci tabulation et speach to text)
- **number** / **numberSigned** / **numberDecimal** / **phone** / **datetime** / **date** / **time** : Clavier numérique

CHECKBOX

- Widget **CheckBox**
- Hérite de **CompoundButton**
- Possible d'alterer la valeur depuis le code:
- **isChecked()**
- **setChecked()**
- **toggle()** (Contraire de l'état actuel du widget)
- Possible d'enregistrer un event listener **OnCheckedChangeListener**

EXEMPLE DE CHECKBOX

```
<CheckBox
xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/check"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Cochez Moi"
/>
```

RADIOBUTTON

- Widget **RadioButton**
- Hérite de **CompoundButton**
- Les radiobuttons doivent toujours être mis dans un **RadioGroup** ???
- Fonctions sur un **RadioGroup**
- **check()**
- **clearCheck()**
- **getCheckedRadioButtonId()**
- **setOnCheckedChangeListener(RadioGroup.OnCheckedChangeListener listener)**

EXEMPLE RADIOBUTTON

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

    <RadioButton
        android:id="@+id/radio1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Or" />

    <RadioButton
        android:id="@+id/radio2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Argent" />

    <RadioButton
        android:id="@+id/radio3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Diamant" />

</RadioGroup>
```

Sommaire

Widgets & Layout

Utiliser l'XML pour le layout
Labels, boutons & champs
LinearLayout
RelativeLayout
TableLayout
ScrollView

CONTENEUR

- Utilisé pour mettre des widgets ensembles d'après un layout spécifique
- Des conteneurs peuvent avoir des conteneurs comme enfants
- Exemples :
- **LinearLayout** : «box model»
- **RelativeLayout**: basé sur des règles
- **TableLayout**: basé sur une grille

LINEARLAYOUT

- Similaire au FlowLayout en Java et HBox/VBox en Adobe Flex
- Les widgets sont alignés en colonnes ou rangées

Conteneurs: 5 points de contrôle:

- Orientation (Alignement des éléments, Vertical, Horizontal)
- Modèle de remplissage (Au besoin, Au maximum)
- Poids (% d'occupation)
- Gravité (centré)
- Espacement (les marges)

CONTENEURS: ORIENTATION

- android:orientation="vertical|horizontal"
- via code:
- setOrientation()
- Paramètres:
- LinearLayout.HORIZONTAL
- LinearLayout.VERTICAL

CONTENEURS: MODELE DE REMPLISSAGE

- android:layout_width et android:layout_height
- Dimensions
- Fixes: 125px basé sur la taille de l'ecran, 300dip basée sur la densité
- Wrapper le contenu: **wrap_content**
- Prendre toute la place disponible: **fill_parent/match_parent**

CONTENEURS: EXEMPLE DE MODELE DE REMPLISSAGE

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="horizontal"
    android:background="#111177" >

    <TextView
        android:id="@+id/field"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#FFFFFF"
        android:text="LinearLayout avec fill_parent" />

</LinearLayout>
```

CONTENEURS : POIDS

- Comment partager l'espace disponible entre widgets
- android:layout_weight

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

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.5"
        android:text="Button 1" />

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.5"
        android:text="Button 2" />
</LinearLayout>
```

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

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.66"
        android:text="Button 1" />

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.33"
        android:text="Button" />

</LinearLayout>
```

Possitionnement Externe et Interne

- android:layout_marginBottom="*50px*"
- android:layout_marginTop="*50px*"
- android:paddingBottom="*50px*"

android:padding (Pour un layout décale les widgets)
android:padding (Pour un widget décale le texte)

CONTENEURS: MODELE DE REMPLISSAGE

- android:layout_width et android:layout_height
- Dimensions
- Fixes: 125px, 300dip,...
- Wrapper le contenu: **wrap_content**
- Prendre toute la place disponible: **fill_parent/match_parent**

CONTENEURS : GRAVITE

- android:layout_gravity
- Comment aligner le widget vis-à-vis de l'écran
- center_horizontal
- center_vertical
- right
- ...
- setGravity() via le code
  - Expl: leWidget.setGravity(Gravity.CENTER_VERTICAL);

android:gravity (suivant le texte du widget)
android:layout_gravity (suivant le parent)

EXEMPLE DE LAYOUT DE CONTENEUR

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

    <RadioGroup
        android:id="@+id/orientation"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:padding="5px" >

        <RadioButton
            android:id="@+id/esmt"
            android:text="ESMT" />

        <RadioButton
            android:id="@+id/ucad"
            android:text="UCAD" />
    </RadioGroup>
```

```
    <RadioGroup
        android:id="@+id/gravity"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="5px" >

        <RadioButton
            android:id="@+id/or"
            android:text="OR" />

        <RadioButton
            android:id="@+id/argent"
            android:text="ARGENT" />

        <RadioButton
            android:id="@+id/bronze"
            android:text="BRONZE" />
    </RadioGroup>

</LinearLayout>
```

RELATIVELAYOUT

- Conteneur **RelativeLayout**
- Permet le positionnement par rapport:
  - Au conteneur
  - Aux autres widgets

RELATIF AU LAYOUT

Possible en mettant une des propriétés suivantes:

- **android:layout_alignParentTop (le haut)**
- **android:layout_alignParentBottom (le bas)**
- **android:layout_alignParentLeft (gauche)**
- **android:layout_alignParentRight (Droite)**
- **android:layout_centerHorizontal**
- **android:layout_centerVertical**
- **android:layout_centerInParent**

Chacune de ces propriétés a **true** ou **false** comme valeur

RELATIF AUX AUTRES WIDGETS

Propriétés qui contrôlent le **positionnement** relatif aux autres widgets:

- **android:layout_above (En haut)**
- **android:layout_below (En bas)**
- **android:layout_toLeftOf (Juste à Gauche de)**
- **android:layout_toRightOf (Juste à Droite de)**

Chacune de ces propriétés prend une référence vers un autre id en valeur.

Exemple: **android:layout_toRightOf="@id/myButton"**

Elément **A** à droite, à gauche, en haut, en bas de Elément **B**

RELATIF AUX AUTRES WIDGETS

- Propriétés qui contrôlent **l'alignement** des widgets par rapport aux autres widgets:
- **android:layout_alignTop**
- **android:layout_alignBottom**
- **android:layout_alignLeft**
- **android:layout_alignRight**
- **android:layout_alignBaseline**

Chacune de ces propriétés prend une référence vers un autre id en valeur

Exemple: **android:layout_toRightOf="@id/myButton"**

Un élement à 5 niveau, haut, bas coté gauche, droite, milieu

Le haut, le bas, la gauche, la droite, le milieu de A est au même niveau que B

RELATIVELAYOUT : POINTS D'ATTENTION

- Il est seulement possible de référencer des widgets qui ont été définis avant celui que vous utilisez.

- Attention à **fill_parent / match_parent** Celui-ci pourrait utiliser l'espace entier et rendre invisible d'autres widgets

Exemple

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <EditText
        android:id="@+id/nomEdit"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="Entrez votre nom ici" />
    <EditText
        android:id="@+id/prenomEdit"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/nomEdit"
        android:hint="Entrez votre prenom ici" />
    <Button
        android:id="@+id/valider"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@id/prenomEdit"
        android:layout_below="@id/prenomEdit"
        android:text="valider" />
    <Button
        android:id="@+id/annuler"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignTop="@id/valider"
        android:layout_toLeftOf="@id/valider"
        android:text="annuler" />
</RelativeLayout>
```

Sommaire

Widgets & Layout

Utiliser l'XML pour le layout
Labels, boutons & champs
LinearLayout
RelativeLayout
TableLayout
ScrollView

TABLELAYOUT

- Conteneur **TableLayout**
- Similaire aux tables en HTML
- Les rangs sont contrôlés par vous via le conteneur **TableRow**
- Les colonnes sont contrôlées par Android
- Une colonne = un widget
- Les widgets peuvent s'étendre sur plusieurs colonnes :
- **android:layout_span**
- L'assignement des colonnes est automatique
- Mais peut aussi être assigné manuellement : **android:layout_column**

TABLELAYOUT

- Les colonnes peuvent avoir des tailles différentes
- La taille peut être contrôlée en utilisant diverses propriétés sur **TableLayout**:
- **android:stretchColumns** (max lengt for all elmts clmn)
- via code: **setColumnStretchable(int index,bool true/false)**
- **android:shrinkColumns** (wrap lengt for all elmts clmn)
- via code: **setColumnShrinkable(int index,bool true/false)**
- **android:collapseColumns**
- via code: **setColumnCollapsed(intindex,booltrue/false)**

android:stretchColumns: 0,1,..

EXEMPLE DE TABLELAYOUT

```
<TableLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:stretchColumns="1" >
    <TableRow>
        <TextView android:text="URL:" />
        <EditText
            android:id="@+id/entry"
            android:layout_span="3" />
    </TableRow>
    <View
        android:layout_height="2px"
        android:background="#0000FF" />
    <TableRow>
        <Button
            android:id="@+id/annuler"
            android:layout_column="2"
            android:text="Annuler" />
        <Button
            android:id="@+id/ok"
            android:text="OK" />
    </TableRow>
</TableLayout>
```

Sommaire

Widgets & Layout

Utiliser l'XML pour le layout
Labels, boutons & champs
LinearLayout
RelativeLayout
TableLayout
ScrollView

SCROLLVIEW

- **TableLayout**, **LinearLayout** or **RelativeLayout** peuvent prendre plus de place que disponible à l'écran.
- Il est possible d'utiliser un **ScrollView** pour du scrolling automatique
- **ScrollView** pour scrolling vertical
- **HorizontalScrollView** pour scrolling horizontal
- Pas de scrolling diagonal ??????

EXEMPLE AVEC SCROLLVIEW

```
<ScrollView
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content" >
    <TableLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:stretchColumns="0" >
        <TableRow>
            <View
                android:layout_height="80px"
                android:background="#000000" />
            <TextView
                android:layout_gravity="center_vertical"
                android:paddingLeft="4px"
                android:text="#000000" />
        </TableRow>
        <TableRow>
            <View
                android:layout_height="80px"
                android:background="#440000" />
            <TextView
                android:layout_gravity="center_vertical"
                android:paddingLeft="4px"
                android:text="#440000" />
        </TableRow>
    </TableLayout>
</ScrollView>
```

FIN