

Interfaces graphiques utilisateur

IUT - Université du Havre

frederic.guinand@univ-lehavre.fr

Plan

- Couleurs - Positions - Fontes
- Layout :
 - LinearLayout
 - RelativeLayout
 - GridLayout
- Composition de layout

Couleurs - Positions - Fontes

background

```
android:background="#ff0000"
```

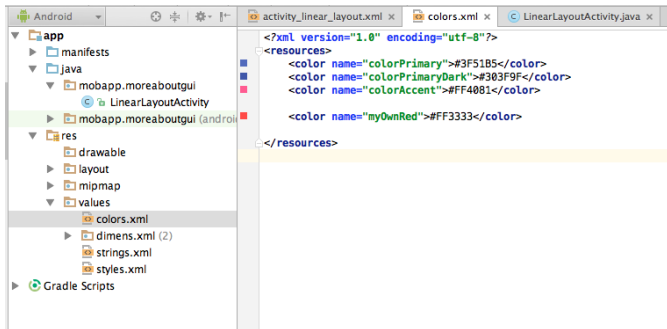
The screenshot shows an IDE with two tabs: 'activity_linear_layout.xml' and 'LinearLayoutActivity.java'. The XML code is as follows:

```
<?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="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"
    android:background="#ff0000"
    tools:context=".LinearLayoutActivity">
    <TextView
        android:text="Hello World!"
        android:background="#00ff00"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</LinearLayout>
```

The 'Preview' window on the right shows a Nexus 4 smartphone with a red background and a green text box containing 'Hello World!'.

background

- ajouter des couleurs dans les ressources
- répertoire `res/values/colors.xml`



background

- le résultat...

The image shows an IDE window with two panes. The left pane displays the XML code for an activity layout, and the right pane shows a preview of the resulting UI on a Nexus 4 device.

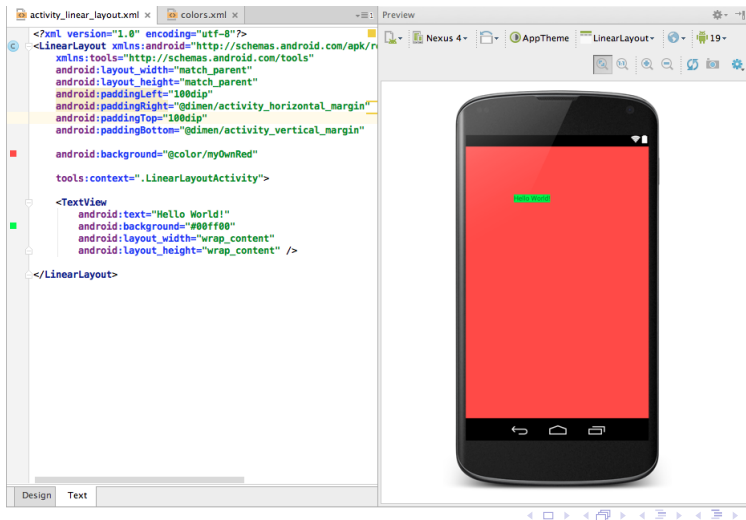
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:background="@color/myOwnRed"
    tools:context=".LinearLayoutActivity">
    <TextView
        android:text="Hello World!"
        android:background="#00ff00"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</LinearLayout>
```

The preview shows a black Nexus 4 smartphone with a red background and a small green rectangle at the top, representing the text view.

aérer votre UI : padding et margin

```
android:paddingLeft="100dip"
```

```
android:paddingTop="100dip"
```



The screenshot displays the Android Studio interface. On the left, the XML editor shows the following code for `activity_linear_layout.xml`:

```
<?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="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="100dip"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="100dip"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:background="@color/myOwnRed"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="Hello World!"
        android:background="#00ff00"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</LinearLayout>
```

On the right, the Preview window shows a Nexus 4 smartphone with a red background and a green text box containing "Hello World!". The interface includes a toolbar with various icons and a status bar at the top.

Unité de mesure

- **dip** ? signifie *density-independent pixel*
- un **dip ou dp** est un **pixel virtuel** équivalent à un **pixel physique sur un écran de densité 160 dpi** (dpi = dots per inch).
⇒ 10 dip sur un 240 dpi est équivalent à
 $10 \times 240 / 160 = 15$ pixels physiques
- il est **fortement recommandé** d'utiliser des **dip** (ou dp) comme unité pour exprimer les dimensions dans les GUI, parce qu'ils permettent de redimensionner automatiquement les éléments selon la taille réelle de l'écran.

écrans pris en charge

- ldpi (low) ~120dpi
- mdpi (medium) ~160dpi
- hdpi (high) ~240dpi
- xhdpi (extra-high) ~320dpi
- xxhdpi (extra-extra-high) ~480dpi
- xxxhdpi (extra-extra-extra-high) ~640dpi

référence

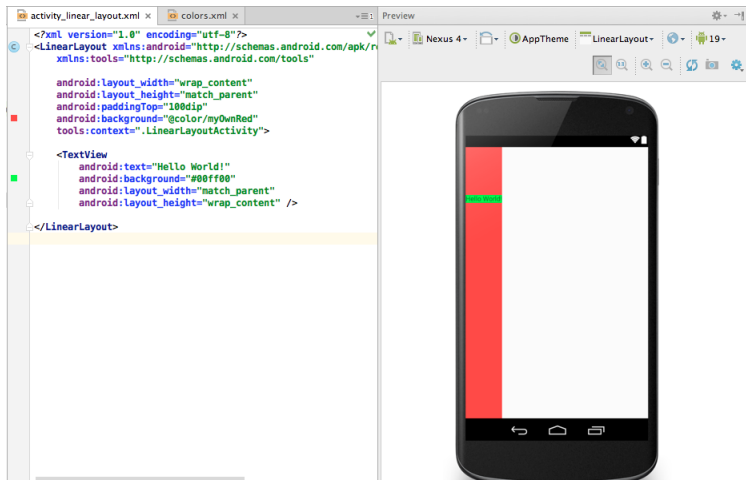
http://developer.android.com/guide/practices/screens_support.html

attributs layout_width et height of the View

- requis

```
android:layout_width="wrap_content"
```

```
android:layout_height="match_parent"
```



aérer votre UI : padding et margin

- ..au sein d'un View: `android:padding="10dip"`
- ...entre les Views: `android:layout_margin="10dip"`

The screenshot displays the Android Studio interface. On the left, the XML editor shows the following code for `activity_linear_layout.xml`:

```
<?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="wrap_content"
    android:layout_height="match_parent"
    android:paddingTop="100dip"
    android:background="@color/myOwnRed"
    android:orientation="vertical"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="Hello World!"
        android:background="#00ff00"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <TextView
        android:text="Ciao a tutti, piacere di essere qui"
        android:background="#0000ff"
        android:layout_margin="10dip"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

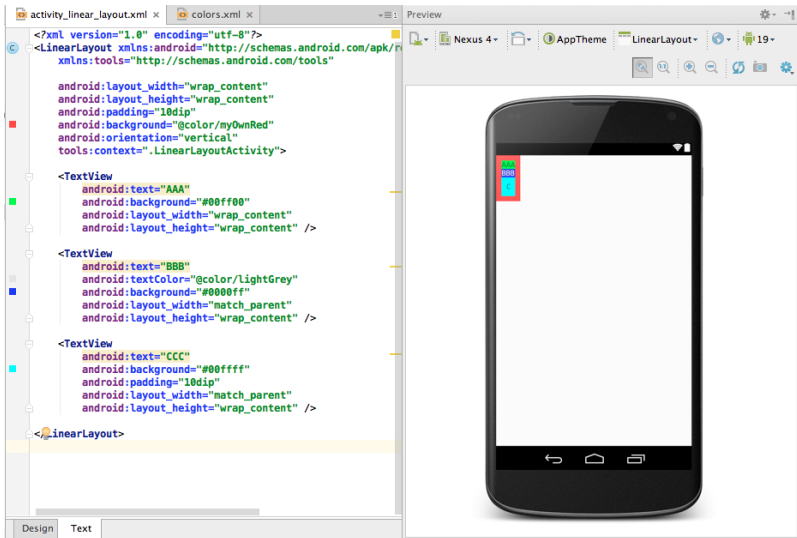
    <TextView
        android:text="Ciao"
        android:background="#00ffff"
        android:padding="10dip"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>
```

On the right, the Preview window shows a Nexus 4 device with a vertical red background. Three text views are stacked vertically: a green one with "Hello World!", a blue one with "Ciao a tutti, piacere di essere qui", and a cyan one with "Ciao". The spacing between the views is visually adjusted to demonstrate padding and margin.

centrer les éléments

- rien n'est centré



```
<?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="wrap_content"
    android:layout_height="wrap_content"
    android:padding="10dip"
    android:background="@color/myOwnRed"
    android:orientation="vertical"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="AAA"
        android:background="#00ff00"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <TextView
        android:text="BBB"
        android:textColor="@color/LightGrey"
        android:background="#0000ff"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

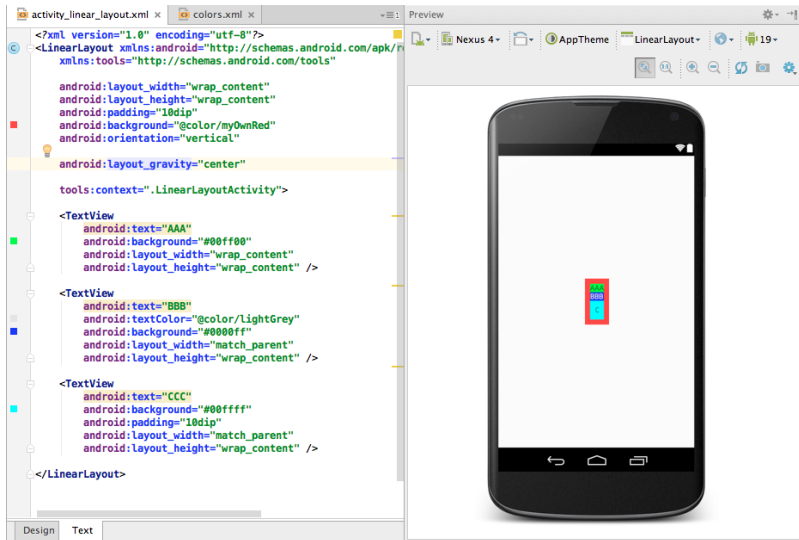
    <TextView
        android:text="CCC"
        android:background="#00ffff"
        android:padding="10dip"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>
```

The screenshot shows the Android Studio interface. On the left, the XML code for `activity_linear_layout.xml` is displayed. It defines a vertical `LinearLayout` with a red background and 10dp padding. It contains three `TextView` elements: the first has text "AAA" and a green background; the second has text "BBB", grey text color, and a blue background; the third has text "CCC" and a cyan background. The preview window on the right shows a Nexus 4 device with these three text views stacked vertically. The text views are not horizontally centered, illustrating the "rien n'est centré" (nothing is centered) state.

centrer les éléments

● centrage du ViewGroup



The screenshot displays the Android Studio interface. On the left, the XML code for `activity_linear_layout.xml` is shown. The root `LinearLayout` is configured with `android:layout_width="wrap_content"`, `android:layout_height="wrap_content"`, `android:padding="10dip"`, `android:background="@color/myOwnRed"`, `android:orientation="vertical"`, and `android:layout_gravity="center"`. The `tools:context` attribute is set to `".LinearLayoutActivity"`. Three `TextView` elements are nested within the `LinearLayout`, each with `android:layout_width="wrap_content"` and `android:layout_height="wrap_content"`. The first `TextView` has `android:text="AAA"` and `android:background="#00ff00"`. The second `TextView` has `android:text="BBB"`, `android:textColor="@color/LightGrey"`, and `android:background="#0000ff"`. The third `TextView` has `android:text="CCC"`, `android:background="#00ffff"`, and `android:padding="10dip"`. On the right, the 'Preview' window shows a Nexus 4 smartphone displaying the rendered layout. The three text elements are vertically and horizontally centered on the screen.

```
<?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="wrap_content"
    android:layout_height="wrap_content"
    android:padding="10dip"
    android:background="@color/myOwnRed"
    android:orientation="vertical"
    android:layout_gravity="center"
    tools:context=".LinearLayoutActivity">
    <TextView
        android:text="AAA"
        android:background="#00ff00"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
    <TextView
        android:text="BBB"
        android:textColor="@color/LightGrey"
        android:background="#0000ff"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <TextView
        android:text="CCC"
        android:background="#00ffff"
        android:padding="10dip"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

centrer les éléments

● centrer un View

```
<?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="match_parent"
    android:layout_height="wrap_content"
    android:padding="10dip"
    android:background="@color/myOwnRed"
    android:orientation="vertical"
    android:layout_gravity="center"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="AAA"
        android:background="#00ff00"
        android:layout_gravity="center"

        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <TextView
        android:text="BBB"
        android:textColor="@color/lightGrey"
        android:background="#0000ff"
        android:layout_margin="20dip"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <TextView
        android:text="CCC"
        android:background="#00ffff"
        android:padding="10dip"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>
```

centrer les éléments

- centrer le contenu d'un View

```
<?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="match_parent"
    android:layout_height="wrap_content"
    android:padding="10dip"
    android:background="@color/myOwnRed"
    android:orientation="vertical"
    android:layout_gravity="center"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="AAA"
        android:background="#00ff00"
        android:layout_gravity="center"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <TextView
        android:text="BBB"
        android:textColor="@color/LightGrey"
        android:background="#0000ff"
        android:layout_margin="20dip"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <TextView
        android:text="CCC"
        android:background="#00ffff"
        android:padding="10dip"
        android:gravity="center"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>
```

The preview shows a Nexus 4 device with a red background. Three text views are stacked vertically and centered: a green one with "AAA", a blue one with "BBB", and a cyan one with "CCC".

gravity vs layout_gravity

- **gravity** est utilisé pour positionné le contenu dans un View
- **layout_gravity** pour positionner un View dans son parent

The screenshot displays the Android Studio interface. On the left, the XML editor shows the following code:

```
<?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="wrap_content"
    android:layout_height="match_parent"
    android:padding="10dp"
    android:background="@color/myOwnRed"
    android:orientation="vertical"
    android:gravity="center"
    android:layout_gravity="center"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="BBB"
        android:textColor="@color/LightGrey"
        android:background="#0000ff"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>
```

On the right, the Preview window shows a Nexus 4 device. The screen displays a vertical red bar with the text "BBB" centered on it. The text is blue, and the bar is red, matching the attributes in the XML code.

jouer avec les fontes

- vous pouvez spécifier la `fontFamily`, la `size` (en dp) et le `style`

Fontes

```
android:fontFamily="sans-serif" // roboto regular
```

- "sans-serif-light" // roboto light
- "sans-serif-condensed" // roboto condensed
- "sans-serif-thin" // roboto thin (android 4.2)
- "sans-serif-medium" // roboto medium (android 5.0)

Style

```
android:textStyle="normal | italic | bold"
```

jouer avec les fontes

The image shows the Android Studio interface with two tabs: 'activity_linear_layout.xml' and 'colors.xml'. The XML code defines a vertical linear layout with a red background and two text views. The first text view has a blue background and displays 'this is a test' in a bold, sans-serif font. The second text view also has a blue background and displays 'this is a test' in an italicized, sans-serif font. The preview window on the right shows a smartphone with a red screen displaying the two text views as described in the code.

```
<?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="match_parent"
    android:layout_height="match_parent"
    android:padding="10dp"
    android:background="@color/myOwnRed"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="this is a test"
        android:textColor="@color/LightGrey"

        android:fontFamily="sans-serif-medium"
        android:textStyle="bold"
        android:textSize="30dp"

        android:background="#0000ff"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <TextView
        android:text="this is a test"
        android:textColor="@color/LightGrey"

        android:fontFamily="sans-serif-thin"
        android:textStyle="italic"
        android:textSize="50dp"

        android:background="#0000ff"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>
```

LinearLayout

android:orientation (requis)

```
android:orientation="vertical"
```

The screenshot displays the Android Studio interface. On the left, the XML editor shows the following code for `activity_linear_layout.xml`:

```
<?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="match_parent"
    android:layout_height="match_parent"
    android:padding="10dip"
    android:background="@color/myOwnRed"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="this is a test"
        android:textColor="@color/LightGrey"
        android:background="#0000ff"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <TextView
        android:text="this is a test"
        android:background="#00ffff"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <TextView
        android:text="this is a test"
        android:background="#ffff00"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

On the right, the Preview window shows a Nexus 4 device with a red background. Three lines of text are displayed at the bottom: "this is a test" in light grey on a blue background, "this is a test" on a cyan background, and "this is a test" on a yellow background. The device is oriented vertically.

orientation

```
android:orientation="horizontal"
```

The screenshot displays the Android Studio interface. On the left, the XML editor shows the following code for `activity_linear_layout.xml`:

```
<?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="match_parent"
    android:layout_height="match_parent"
    android:padding="10dip"
    android:background="@color/myOwnRed"
    android:orientation="horizontal"
    android:gravity="center"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="this is a test"
        android:textColor="@color/LightGrey"
        android:background="#0000ff"
        android:layout_width="wrap_content"
        android:layout_height="match_parent" />

    <TextView
        android:text="this is a test"
        android:background="#00ffff"
        android:layout_width="wrap_content"
        android:layout_height="match_parent" />

    <TextView
        android:text="this is a test"
        android:background="#ffff00"
        android:layout_width="wrap_content"
        android:layout_height="match_parent" />
</LinearLayout>
```

On the right, the Preview window shows a Nexus 4 device with the following visual output: a red background with three vertical bars in the center. From left to right, the bars are blue, cyan, and yellow. The text "this is a test" is visible at the top of the screen.

Attention

`android:orientation="horizontal"` (LinearLayout) et
`android:layout_width="match_parent"` (TextView)

The screenshot displays the Android Studio interface. On the left, the XML code for `activity_linear_layout.xml` is shown. The `LinearLayout` root element is highlighted, with the `android:orientation="horizontal"` attribute selected. Below it, three `TextView` elements are defined, each with a different background color and the `android:layout_width="match_parent"` attribute. The preview window on the right shows a mobile device with a red background and a blue horizontal bar containing the text "this is a test".

```
<?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="match_parent"
    android:layout_height="match_parent"
    android:padding="10dip"
    android:background="@color/myOwnRed"
    android:orientation="horizontal"
    android:gravity="center"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="this is a test"
        android:textColor="@color/LightGrey"
        android:background="#0000ff"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <TextView
        android:text="this is a test"
        android:background="#00ffff"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <TextView
        android:text="this is a test"
        android:background="#ffff00"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

Combiner les Layout

- pas plus de deux niveaux d'imbrication recommandé

The image shows an IDE window with two panes. The left pane displays the XML code for an activity layout, and the right pane shows a preview of the app on a Nexus 4 device.

```
<?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="match_parent" android:layout_height="match_parent"
    android:padding="10dip" android:background="@color/myOwnRed"
    android:orientation="vertical"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="this is a test" android:textColor="@color/LightGrey"
        android:background="#0000ff" android:gravity="center"
        android:layout_width="match_parent" android:layout_height="wrap_content" />

    <LinearLayout
        android:layout_width="match_parent" android:layout_height="wrap_content"
        android:gravity="center"
        android:background="#00ff00" android:padding="20dip">

        <TextView
            android:text="this is a test" android:textColor="@color/LightGrey"
            android:background="#ff0000"
            android:layout_width="wrap_content" android:layout_height="wrap_content" />

        <TextView
            android:text="this is a test" android:background="#ffff00"
            android:layout_width="wrap_content" android:layout_height="wrap_content" />

    </LinearLayout>
</LinearLayout>
```

The preview shows a smartphone with a red background. At the top, there is a blue bar with the text "this is a test" in white. Below that is a green bar with the text "this is a test" in red. The rest of the screen is red.

RelativeLayout

RelativeLayout

- chaque View ou ViewGroup (les layout) au sein d'un RelativeLayout doit indiquer sa position par rapport aux autres éléments
- si aucune indication n'est donnée, le dernier élément ajouté sera positionné sur les précédents

The screenshot displays the Android Studio interface. On the left, the XML editor shows the following code for `activity_linear_layout.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<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:padding="10dp" android:background="@color/LightGrey"
    tools:context=".LinearLayoutActivity">

    <TextView
        android:text="very long test" android:textColor="#000000"
        android:background="#00ff00" android:textSize="20dp"
        android:layout_width="wrap_content" android:layout_height="wrap_content" />

    <TextView
        android:text="long test" android:textColor="@color/LightGrey"
        android:background="#ff0000" android:textSize="20dp"
        android:layout_width="wrap_content" android:layout_height="wrap_content" />

    <TextView
        android:text="test" android:textColor="#000000"
        android:background="#ffff00" android:textSize="20dp"
        android:layout_width="wrap_content" android:layout_height="wrap_content" />

</RelativeLayout>
```

On the right, the Preview window shows a Nexus 4 device with the rendered UI. It features three text views stacked vertically: a green background with black text "test test tes", a red background with grey text "long test", and a yellow background with black text "test".

RelativeLayout

```
android:layout_toRightOf="@id/tv1"  
android:layout_below="@id/tv2"
```

The image shows an IDE window with two panes. The left pane displays the XML code for an activity layout, and the right pane shows a preview of the layout on a Nexus 4 device.

```
<?xml version="1.0" encoding="utf-8"?>  
<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:padding="10dip" android:background="@color/LightGrey"  
    tools:context=".LinearLayoutActivity">  
  
    <TextView  
        android:id="@+id/tv1" android:padding="10dip"  
        android:text="very long test" android:textColor="#000000"  
        android:background="#00ff00" android:textSize="20dip"  
        android:layout_width="wrap_content" android:layout_height="wrap_content" />  
  
    <TextView  
        android:id="@+id/tv2" android:padding="10dip"  
        android:text="long test" android:textColor="@color/LightGrey"  
        android:background="#fff000" android:textSize="20dip"  
        android:layout_width="wrap_content" android:layout_height="wrap_content"  
        android:layout_toRightOf="@id/tv1"/>  
  
    <TextView  
        android:id="@+id/tv3" android:padding="10dip"  
        android:text="test" android:textColor="#000000"  
        android:background="#ffff00" android:textSize="20dip"  
        android:layout_width="wrap_content" android:layout_height="wrap_content"  
        android:layout_below="@id/tv1" />  
  
    <TextView  
        android:id="@+id/tv4" android:padding="10dip"  
        android:text="test" android:textColor="@color/LightGrey"  
        android:background="#0000ff" android:textSize="20dip"  
        android:layout_width="wrap_content" android:layout_height="wrap_content"  
        android:layout_below="@id/tv1" android:layout_toRightOf="@id/tv3"/>  
  
</RelativeLayout>
```

The preview pane shows a Nexus 4 device displaying the layout. The text is arranged as follows:

- A yellow box containing "very long test" and a red box containing "long test" are positioned horizontally next to each other.
- A green box containing "test" and a blue box containing "test" are positioned horizontally next to each other below the yellow and red boxes.

RelativeLayout

- les éléments peuvent être positionnés relativement à différents repères particuliers d'autres éléments comme leur extrémité gauche ou droite

```
android:layout_alignLeft="@id/tv2"
```

- cet attribut indique que l'extrémité gauche de la vue courante sera alignée avec celle de la View d'identifiant tv2
- parmi les attributs les plus communément utilisés

```
android:layout_  
    above  
    below  
    toRightOf  
    alignLeft
```

- ils peuvent également être combinés pour obtenir des configurations particulières

activity_linear_layout.xml x colors.xml x LinearLayoutActivity.java x Preview

```
tools:context=".LinearLayoutActivity">

<TextView
    android:id="@+id/tv1" android:padding="10dip"
    android:text="very long test" android:textColor="#000000"
    android:background="#00ff00" android:textSize="20dip"
    android:layout_width="wrap_content" android:layout_height="wrap_content" />

<TextView
    android:id="@+id/tv2" android:padding="10dip"
    android:text="long test" android:textColor="@color/LightGrey"
    android:background="#ff0000" android:textSize="20dip"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:layout_toRightOf="@id/tv1"/>

<TextView
    android:id="@+id/tv3" android:padding="10dip"
    android:text="test" android:textColor="#000000"
    android:background="#ffff00" android:textSize="20dip"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:layout_below="@id/tv1" />

<TextView
    android:id="@+id/tv4" android:padding="10dip"
    android:text="test" android:textColor="@color/LightGrey"
    android:background="#0000ff" android:textSize="20dip"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:layout_below="@id/tv1" android:layout_toRightOf="@id/tv3"/>

<TextView
    android:id="@+id/tv5" android:padding="10dip"
    android:text="test" android:textColor="@color/LightGrey"
    android:background="#000000" android:textSize="20dip"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:layout_below="@id/tv4"
    android:layout_alignRight="@id/tv1" android:layout_alignLeft="@id/tv4"/>

</RelativeLayout>
```

Design Text

Preview

Nexus 4 v AppTheme



activity_linear_layout.xml x colors.xml x LinearLayoutActivity.java x Preview

```
tools:context=".LinearLayoutActivity">
<TextView
    android:id="@+id/tv1" android:padding="10dip"
    android:text="very long test" android:textColor="@#000000"
    android:background="@#00ff00" android:textSize="20dip"
    android:layout_width="wrap_content" android:layout_height="wrap_content" />
<TextView
    android:id="@+id/tv2" android:padding="10dip"
    android:text="tv2" android:textColor="@color/lightGrey"
    android:background="@#ff0000" android:textSize="20dip"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:layout_toRightOf="@id/tv1"/>
<TextView
    android:id="@+id/tv3" android:padding="10dip"
    android:text="tv3" android:textColor="@#000000"
    android:background="@#ffff00" android:textSize="20dip"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:layout_below="@id/tv2" />
<TextView
    android:id="@+id/tv4" android:padding="10dip"
    android:text="tv4" android:textColor="@color/lightGrey"
    android:background="@#0000ff" android:textSize="20dip"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:layout_below="@id/tv2" android:layout_toRightOf="@id/tv3"/>
<TextView
    android:id="@+id/tv5" android:padding="10dip"
    android:text="tv5" android:textColor="@color/lightGrey"
    android:background="@#000000" android:textSize="20dip"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:layout_below="@id/tv4"
    android:layout_alignRight="@id/tv2" android:layout_toRightOf="@id/tv4"/>
</RelativeLayout>
```

Design Text

Preview

Nexus 4+ AppTheme



Autre exemple

```
<TextView android:id="@+id/tv1"
  android:layout_width="wrap_content" android:layout_height="wrap_content"
  android:text="tv1" android:textSize="@dimen/mediumLetters"
  android:background="@color/white"
  android:layout_below="@+id/tv3" android:layout_toRightOf="@id/tv3" />

<TextView android:id="@+id/tv2" android:padding="@dimen/smallSpace"
  android:layout_width="wrap_content" android:layout_height="wrap_content"
  android:text="tv2" android:textSize="@dimen/mediumLetters"
  android:background="@color/blue" android:textColor="@color/white"
  android:layout_below="@id/tv1" android:layout_toLeftOf="@id/tv1"/>

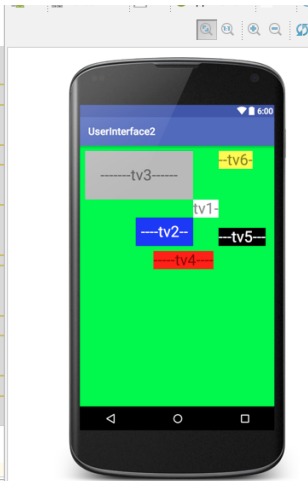
<TextView android:id="@id/tv3" android:padding="@dimen/someSpace"
  android:layout_width="wrap_content" android:layout_height="wrap_content"
  android:text="tv3" android:textSize="@dimen/mediumLetters"
  android:background="@color/Lightgrey"/>

<TextView android:id="@+id/tv4" android:layout_margin="@dimen/smallSpace"
  android:layout_width="wrap_content" android:layout_height="wrap_content"
  android:text="tv4" android:textSize="@dimen/mediumLetters"
  android:background="#ff0000" android:layout_alignRight="@id/tv1"
  android:layout_below="@id/tv2"/>

<TextView android:id="@+id/tv5"
  android:layout_width="wrap_content" android:layout_height="wrap_content"
  android:text="tv5" android:textSize="@dimen/mediumLetters"
  android:background="#000000" android:textColor="#ffffff"
  android:layout_toRightOf="@id/tv1" android:layout_alignBottom="@id/tv2"/>

<TextView android:id="@+id/tv6"
  android:layout_width="wrap_content" android:layout_height="wrap_content"
  android:text="tv6" android:textSize="@dimen/mediumLetters"
  android:background="#ffff00"
  android:layout_toRightOf="@id/tv1" android:layout_alignTop="@id/tv3"/>

</RelativeLayout>
```





la prévisualisation n'est pas une garantie

- une bonne prévisualisation ne signifie pas que le code est correct
- considérons l'interface graphique suivante

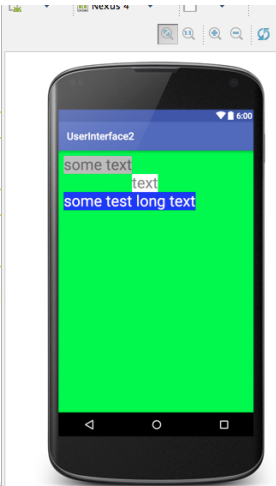
Prévisualisation ok

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent"
    android:padding="10dp"
    android:background="#00ff00">

    <TextView android:id="@+id/tv1"
        android:layout_width="wrap_content" android:layout_height="wrap_content"
        android:text="text" android:textSize="30dp"
        android:background="@color/white"
        android:layout_below="@id/tv3" android:layout_toRightOf="@id/tv3" />

    <TextView android:id="@+id/tv2"
        android:layout_width="wrap_content" android:layout_height="wrap_content"
        android:text="some test long text" android:textSize="30dp"
        android:background="@color/blue" android:textColor="@color/white"
        android:layout_below="@id/tv1" android:layout_toLeftOf="@id/tv2" />

    <TextView android:id="@+id/tv3"
        android:layout_width="wrap_content" android:layout_height="wrap_content"
        android:text="some text" android:textSize="30dp"
        android:background="@color/ligthgrey"/>
</RelativeLayout>
```



...à l'exécution...

```
app/src/main/res/layout/relativelayout_example.xml
No resource found that matches the given name (at 'layout_below' with value '@id/tv3').
No resource found that matches the given name (at 'layout_toRightOf' with value '@id/tv3').
```

- en effet, nous utilisons une référence à un View `@id/tv3` (dans `TextView tv1`) avant sa définition
- ce qui doit être fait est de changer la première référence à `tv3` en un `@+id/tv3` et passer l'autre référence de `@+id/tv3` à `@id/tv3`.
- ce doit être fait pour toutes les références qui sont dans la même situation

Dépendances circulaires

- si vous définissez la position de l'une de votre View v_1 en fonction de View v_2 et si la position de v_2 dépend également de la position de v_1 l'éditeur vous avertit

```
<TextView android:id="@+id/tv1"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:text="tv1" android:textSize="@dimen/mediumletters"
    android:background="@color/white"
    android:layout_below="@+id/tv3" android:layout_toRightOf="@id/tv3" />

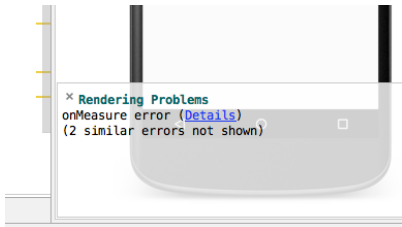
<TextView android:id="@+id/tv2" android:padding="@dimen/smallspace"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:text="-----tv2-----" android:textSize="@dimen/mediumletters"
    android:background="@color/blue" android:textColor="@color/white"
    android:layout_below="@id/tv1" android:layout_toLeftOf="@id/tv1"/>

<TextView android:id="@id/tv3" android:padding="@dimen/smallspace"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:text="-----tv3-----" android:textSize="@dimen/mediumletters"
    android:background="@color/ligthgrey" android:layout_alignRight="@id/tv1"/>

<TextView android:id="@+id/tv4" android:layout_margin="@dimen/smallspace"
```

Dépendances circulaires

- l'éditeur vous avertit et si vous cliquez sur l'avertissement vous obtenez le message complet



```
Stack trace
java.lang.IllegalStateException: Circular dependencies cannot exist in RelativeLayout
    at android.widget.RelativeLayout$DependencyGraph.getSortedViews(RelativeLayout.java:1724)
    at android.widget.RelativeLayout.sortChildren(RelativeLayout.java:382)
    at android.widget.RelativeLayout.onMeasure(RelativeLayout.java:389)
```

GridLayout

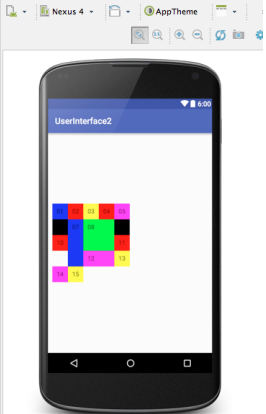
Particularités

- les éléments sont arrangés comme un tableau
- le nombre de colonnes est fixé par un attribut `android:columnCount`
- à chaque ajout d'un nouvel élément (View ou ViewGroup) dans ce layout, sa position dépend à la fois du nombre d'éléments qui ont été ajoutés et de la valeur de `columnCount`.
- si vous avez déjà ajouté $n - 1$ éléments à ce layout et si le nombre de colonnes est k , alors le nouvel élément sera positionné à la colonne numéro $(n \bmod (k + 1)) + 1$

GridLayout : exemple

```
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="wrap_content"
    android:columnCount="5" android:padding="@dimen/smallspace" android:layout_gravity="center">

<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="01" android:background="@color/blue"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="02" android:background="@color/red"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="03" android:background="@color/yellow"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="04" android:background="@color/red"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="05" android:background="@color/purple"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="06" android:background="@color/black"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="07" android:background="@color/blue"
    android:layout_rowSpan="3" android:layout_gravity="fill"/&>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="08" android:background="@color/green"
    android:layout_rowSpan="2" android:layout_columnSpan="2" android:layout_gravity="fill"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="09" android:background="@color/black"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="10" android:background="@color/red"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="11" android:background="@color/red"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="12" android:background="@color/purple"
    android:layout_columnSpan="2" android:layout_gravity="fill"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="13" android:background="@color/yellow"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="14" android:background="@color/purple"/>
<TextView android:layout_height="wrap_content" android:layout_width="wrap_content" android:padding="@dimen/smallspace"
    android:text="15" android:background="@color/yellow" />
```



Exercice

- sans utiliser l'éditeur graphique, pouvez-vous reproduire les dispositions suivantes (RelativeLayout à gauche et GridLayout à droite) ?



Réutilisation de Layouts

- si dans votre application différentes activités partagent une disposition commune des éléments ou d'une partie de leur interface graphique, vous pouvez utiliser :
 - la balise `<include />` et
 - la balise `<merge />`
- la balise `include` à l'intérieur d'un fichier `layout.xml` permet de faire référence à un autre fichier xml contenant un autre layout

```
<?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="match_parent"
    android:layout_height="match_parent" android:padding="@dimen/activity_vertical_margin"
    android:orientation="vertical" tools:context=".MainActivity">

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

    <include layout="@layout/included_layout" />

</LinearLayout>
```

- la layout ajouté est également situé dans le répertoire `res/layout/`
- il prend la forme de n'importe quel autre fichier de type `layout.xml` file avec un élément `ViewGroup` contenant d'autres Views, ou...

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal" android:gravity="center"
    android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:background="@color/red">

    <TextView android:layout_margin="16dp"
        android:text="in the included layout" android:background="@color/green"
        android:layout_width="wrap_content" android:layout_height="wrap_content" />

    <TextView
        android:text="me too" android:background="@color/blue"
        android:layout_width="wrap_content" android:layout_height="wrap_content" />
</LinearLayout>
```

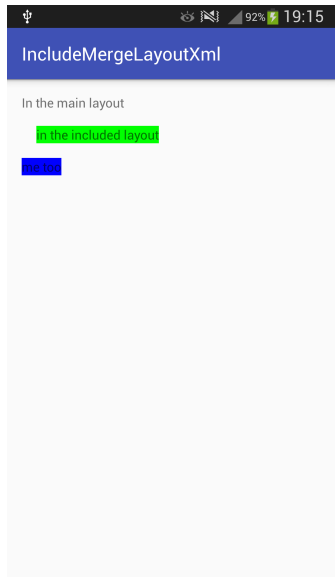
- il peut prendre une forme plus simple en utilisant la balise `merge`
- dans ce cas, les éléments ajoutés se conformeront à la mécanique de disposition du ViewGroup englobant

```
><merge xmlns:android="http://schemas.android.com/apk/res/android">  
  <TextView android:layout_margin="@dimen/activity_horizontal_margin"  
    android:text="@string/tvlincluded" android:background="@color/green"  
    android:layout_width="wrap_content" android:layout_height="wrap_content" />  
  <TextView  
    android:text="@string/tv2included" android:background="@color/blue"  
    android:layout_width="wrap_content" android:layout_height="wrap_content" />  
</merge>
```

without merge



with merge



Exercice

- Nous souhaitons proposer à l'utilisateur deux dispositions différentes selon que l'écran soit en mode portrait ou paysage
- Comment peut-on faire ça ?
- Reproduisez (sans utiliser l'éditeur graphique) les deux écrans ci-dessous.



Portrait screen layout showing text elements stacked vertically. The first row contains 'Texte 1', 'Texte 4', 'Texte 5', and 'Texte 6'. The second row contains 'Texte 2'. The third row contains 'Texte 3'. A green bar is positioned to the right of the text.



Landscape screen layout showing text elements arranged horizontally. The first row contains 'Texte 1'. The second row contains 'Texte 2'. The third row contains 'Texte 3'. The fourth row contains 'Texte 4', 'Texte 5', and 'Texte 6'. A green bar is positioned to the right of the text.

- pour obtenir l'orientation de l'écran :

```
this.getResources().getConfiguration().orientation
```

qui vaut **ORIENTATION_PORTRAIT** ou

ORIENTATION_LANSCAPE (this correspond à l'Activity)