

Département d'Informatique, Licence S06
Cours développement mobile

Création des interfaces utilisateur
Dr. Sabri GHAZI



Université Badji Mokhtar Annaba

E-mail: sabri.ghazi@univ-annaba.dz



Rappel

- Support du cours :
- <http://alif.univ-annaba.org>
- Clé : **DevMob16**
- Série de TP disponible sur le site.
- Evaluation.
- Contact : sabri.ghazi@univ-annaba.dz

Dr. Sabri GHAZI

L'interface utilisateur



✓ **Ecran, Interface, Vue, I.H.M., U.I, G.U.I.**

- ✓ C'est la partie visible de l'application.
- ✓ Elle permet l'interaction avec l'utilisateur:
 - Lui **permettre** de **saisir** des informations.
 - **Visualiser** des informations.
 - **Lancer** des traitements.



- Respecter : **l'ergonomie, la charge cognitive,** bref toutes les **recommandations** des **IHM.**

Dr. Sabri GHAZI

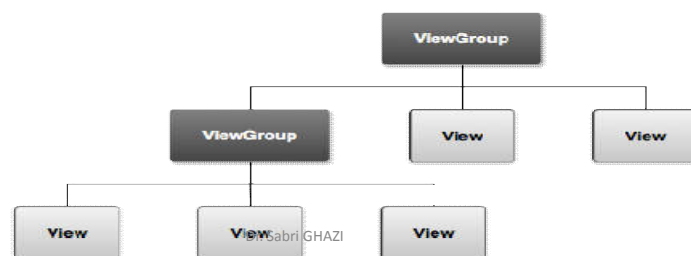
L'interface utilisateur

- Une partie **dynamique** contenant le code **Java**.
- Une partie **statique** contenant la description de l'interface utilisateur en **XML**.
 - Cette séparation permet de
 - ✓ Réaliser des **maquettes(prototypes)** de l'application .
 - ✓ Concevoir les interfaces **indépendamment** de la programmation.
 - ✓ Collaborer entre plusieurs développeurs sur une même fonctionnalité.
 - ✓ Faciliter de modifier l'interface **sans changer** le code source de l'application (**externalisation**).
 - ✓ Faciliter l'**internationalisation** de l'application.

Dr. Sabri GHAZI

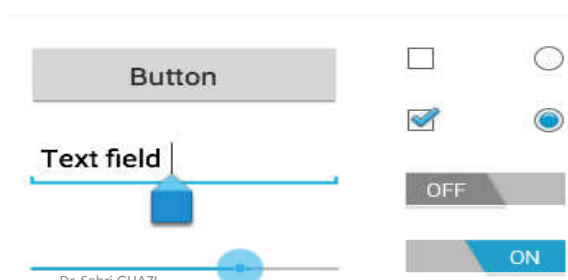
Les interfaces utilisateur

- Le package des classes qui permettent la création des interfaces utilisateurs sous Android sont organisés comme suit :
- Chaque composant de l'interface utilisateur est : Un **View**
 - Un **ViewGroup** est un composant qui peut contenir d'autre **View** ou d'autre **ViewGroup**



Les activités et l'interface utilisateur

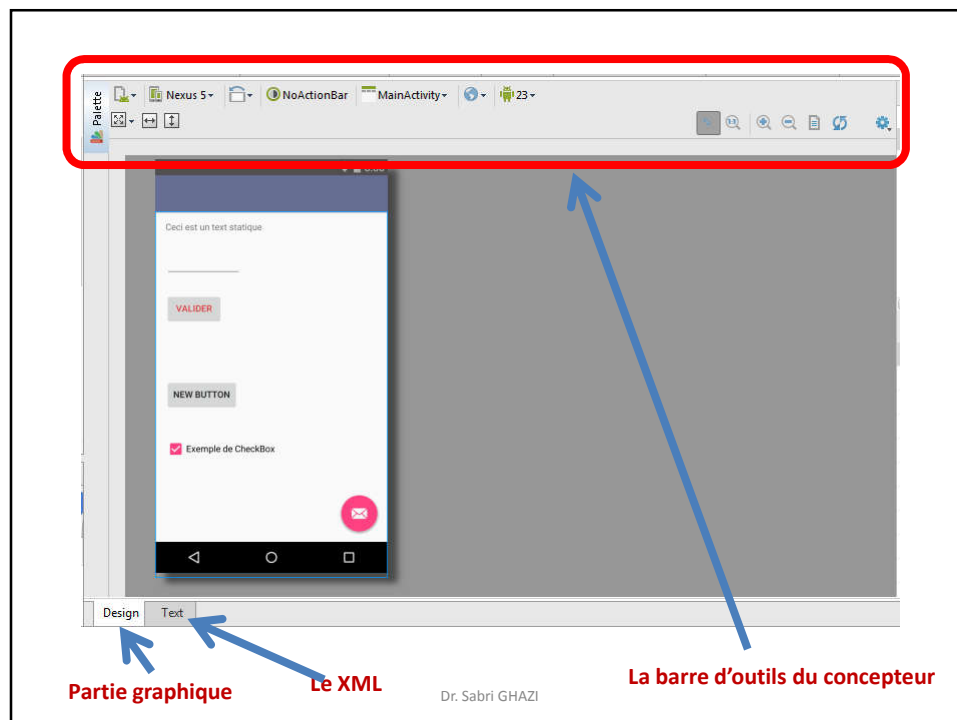
- Comme vous avez remarqué, chaque activité possède sa propre **IHM**.
- Les composants de base (les **widgets**).
 - Les Zones de texte statique **TextView**
 - Les Zones de saisi de texte **EditText**
 - Les Boutons radio **RadioButton**



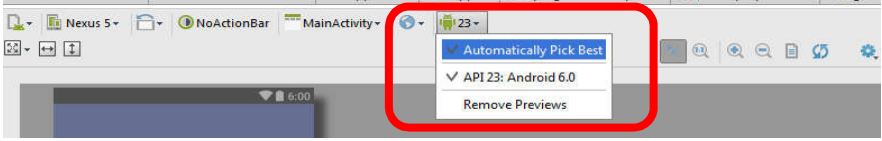
Utilisation du concepteur d'IHM

- Le concepteur des **IHM** permet de décrire d'une façon graphique les écrans de notre application.
- La description **XML** est générée automatiquement.

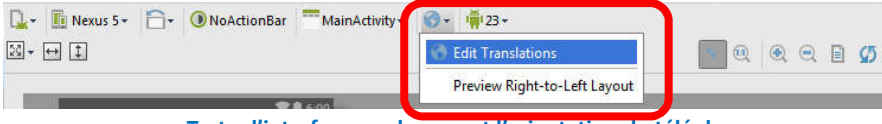
Dr. Sabri GHAZI



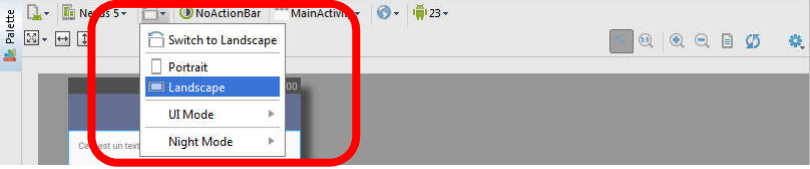
Modifier la version de l'API Android



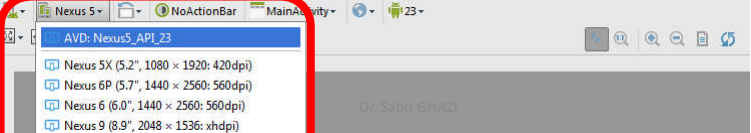
Ajouter une traduction en une autre langue



Tester l'interface en changeant l'orientation du téléphone

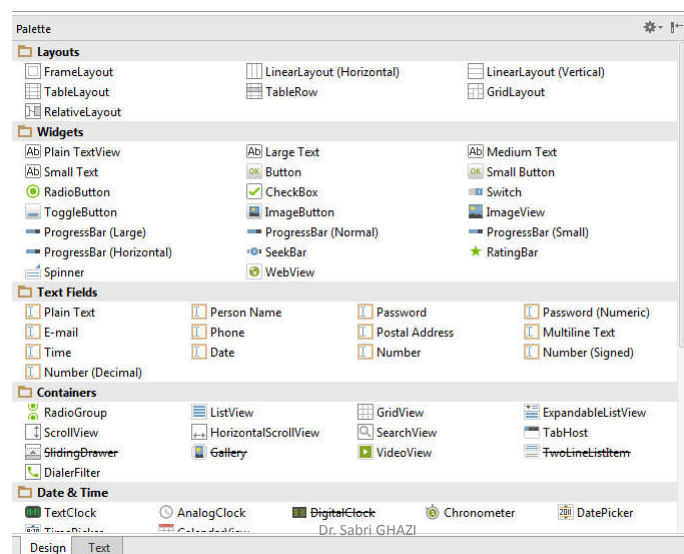


Tester l'interface sous un autre téléphone



Dr. Sabri GHAZI

La palette des composants



Le TextView

- Un élément **TextView** possède de nombreux attributs:
 - Text : le texte qu'on cherche à l'afficher
 - textStyle
 - textSize
 -

Widgets

Ab Plain TextView

Ab Large Text

Ab Medium Text

Ab Small Text

Dr. Sabri GHAZI

Les zones de saisie: EditText

- Permet de définir des zones de saisi de texte.
- Ce qui permet à l'utilisateur d'introduire des informations textuelles.

Text Fields

- Plain Text
- Person Name
- Password
- Password (Numeric)
- E-mail
- Phone
- Postal Address
- Multiline Text
- Time
- Date
- Number
- Number (Signed)
- Number (Decimal)

Dr. Sabri GHAZI

Les zones de saisie: EditText

- En java on récupère le contenu de la zone de saisi en utilisant la méthode ***getText()***.
- On peut aussi modifier le texte saisi en appelant la méthode ***setText(...)***

Dr. Sabri GHAZI

Button

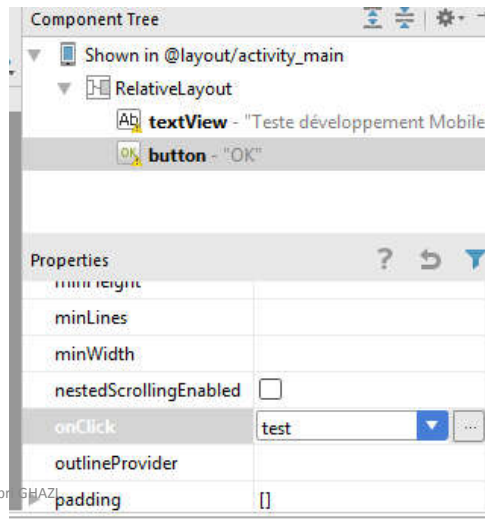
- C'est l'élément qui permet de gérer des événements , telle que l'action de clique.
 - Pour répondre à un événement de clique, on a deux méthodes:
 - implémenter l'interface **View.OnClickListener**.
 - Définir une méthode publique et définir la propriété **android:onClick**

- OK Button
- OK Small Button
- RadioButton
- CheckBox
- Switch
- ToggleButton
- ImageButton



Dr. Sabri GHAZI

Gestion des événements



Dr. Sabri GHAZI

Gestion des événements

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Valider"
    android:id="@+id/btn_valider"
    android:layout_below="@+id/editText"
    android:layout_alignParentStart="true"
    android:layout_marginTop="25dp"
    android:textColor="#dd3d3d"
    android:onClick="test" />
```

Dr. Sabri GHAZI

Gestion des événements

Utiliser la propriété **OnClick** et après ajouter une méthode qui se lancera lors du clique qui doit avoir la signature suivante :

```
public void test( View v) {
//ici les instructions qui permettent de traiter
l'événement
}
```

Dr. Sabri GHAZI

```
package dz.univannaba.sabrighazi.tpdevmob001;

import android.content.Intent;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends Activity{
    TextView t=null;
    int cpt=0;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);

        t=(TextView) findViewById(R.id.txt1)
    }

    public void test(View v){
        t.setText("Ce btn a été cliqué " +cpt+ " fois");
        cpt=cpt+1;
    }//fin
} // fin de la calsse
```



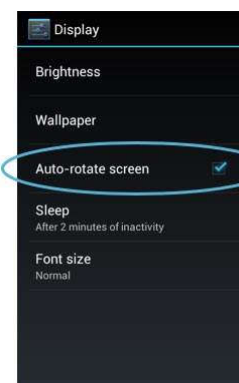
Dr. Sabri GHAZI

- La classe hérite de la classe Activity les méthodes :
 - **findViewById** : cette méthode permet de récupérer la référence d'un composant de l'interface utilisateur, elle reçoit l'identifiant de ce composant.
 - **setContentView** : cette méthode permet de charger l'interface utilisateur décrite dans le fichier XML de l'activity.
 - La fonction qui traite l'événement du bouton, possède un paramètre de type View, ce paramètre contient la référence du bouton cliqué.

Dr. Sabri GHAZI

CheckBox

- C'est le composant qui permet de représenter des informations du genre On/Off (**vrais/faux**).
- Les propriétés principales :
 - **android:text**
 - Le texte affiché
 - **android:checked**
 - L'état du composant (coché ou non)

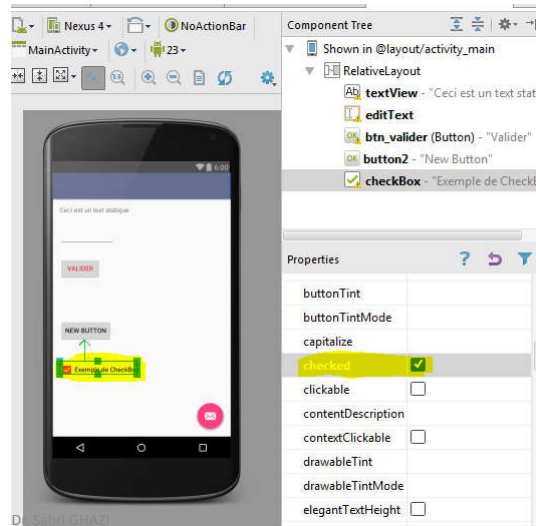


Dr. Sabri GHAZI

CheckBox

Méthodes java :

- On peut récupérer son état en utilisant la méthode `isChecked()`
 - True ou False
- On peut modifier son état en appelant `setChecked (bool b)`



RadioButton

- Permet à l'utilisateur de sélectionner une option parmi un ensemble de valeur possible.

ATTENDING?

☒ Yes ☐ Maybe ☐ No

Exemple de RadioButton

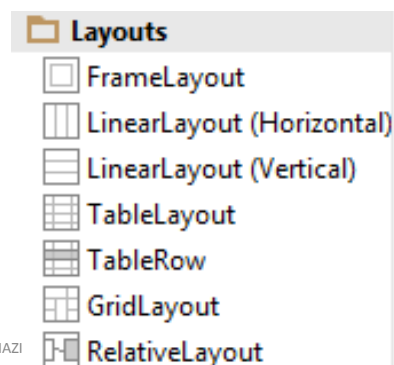
```
<RadioGroup android:layout_width="fill_parent" android:layout_height="fill_parent"
    android:layout_below="@+id/imageButton"
    android:layout_alignLeft="@+id/textView2"
    android:layout_alignStart="@+id/textView2">
    <RadioButton android:layout_width="142dp" android:layout_height="wrap_content"
        android:text="JAVA" android:id="@+id/radioButton" android:textSize="25dp"
        android:textColor="@android:color/holo_red_light" android:checked="false"
        android:layout_gravity="center_horizontal" />
    <RadioButton android:layout_width="wrap_content"
        android:layout_height="wrap_content" android:text="ANDROID"
        android:id="@+id/radioButton2" android:layout_gravity="center_horizontal"
        android:checked="false" android:textColor="@android:color/holo_red_dark"
        android:textSize="25dp" />
    <RadioButton android:layout_width="136dp" android:layout_height="wrap_content"
        android:text="HTML" android:id="@+id/radioButton3"
        android:layout_gravity="center_horizontal" android:checked="false"
        android:textSize="25dp" android:textColor="@android:color/holo_red_dark" />
</RadioGroup>
```

Dr. Sabri GHAZI

La boîte à outils

- **Android** offre une boîte à outils très riche, elle offre un choix très large en ce qui concerne les composants de l'interface utilisateur.

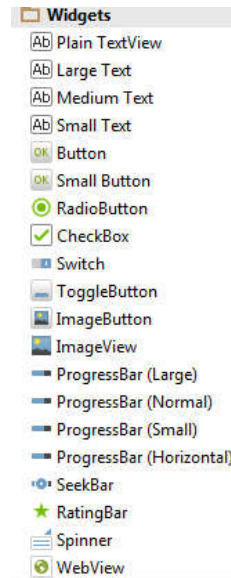
– Les conteneurs



Dr. Sabri GHAZI

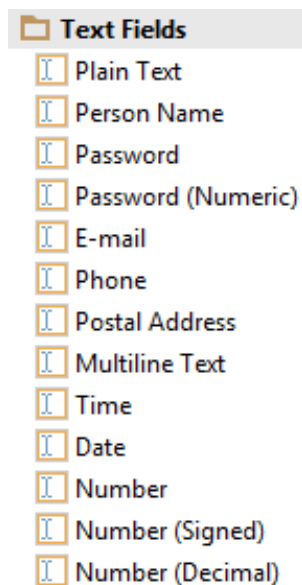
Les widgets

- Button,
- TextView,
- Radio



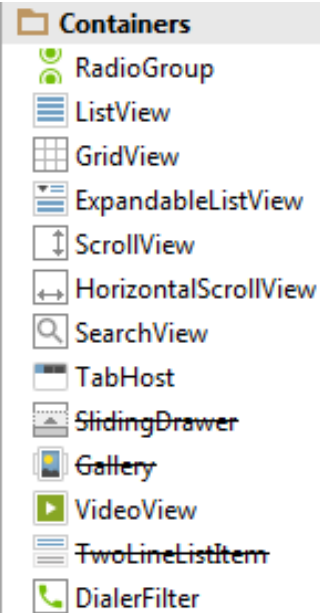
Dr. Sabri GHAZI

Les zones de texts



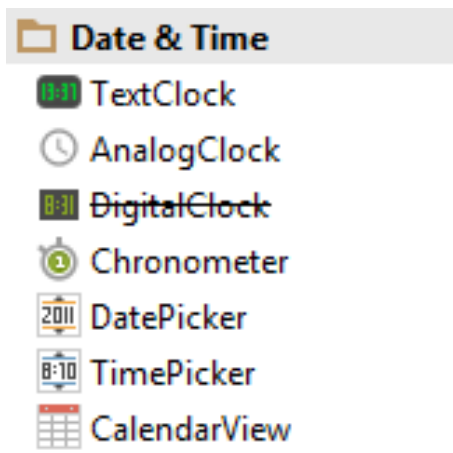
Dr. Sabri GHAZI

D'autres conteneurs



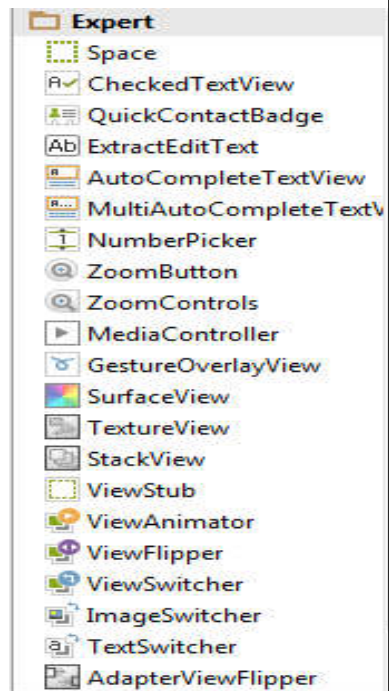
Dr. Sabri GHAZI

Les composants pour manipuler les dates



Dr. Sabri GHAZI

D'autres éléments



Dr. Sabri GHAZI

Les **Layout** (gabarit)

- C'est des conteneurs qui aide à positionner les composants de l'interface.

Dr. Sabri GHAZI

Les ViewGroup prédéfinis (Layout)

- **LinearLayout** : dispose les éléments de gauche à droite ou du haut vers le bas.
- **RelativeLayout** : les éléments sont placés les uns par rapport aux autres.
- **TableLayout** : les éléments sont placés dans un tableau, chacun dans une cellule.
- **FrameLayout** : disposition en haut à gauche en empilant les éléments.
- **GridLayout** : disposition matricielle avec N colonnes et un nombre infini de lignes.

Dr. Sabri GHAZI

LinearLayout

- Propriétés principales
 - **orientation**, elle peut être **vertical** ou **horizontal**
 - **Layout_width** : spécifie la largeur en nombre, on peut mettre une constante prédéfinie comme:
 - **fill_parent** : le composant remplit tous l'espace disponible (selon la taille de son père)
 - **Wrap_content** : le contenu sera ajusté selon la taille du composant.
 - **Layout_height** : l'hauteur du composant, peut avoir es mêmes valeurs que le **width**.

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

```
</LinearLayout>
```

Ici on doit placer les composants de notre écran

Dr. Sabri GHAZI

LinearLayout

Exemple



Dr. Sabri GHAZI

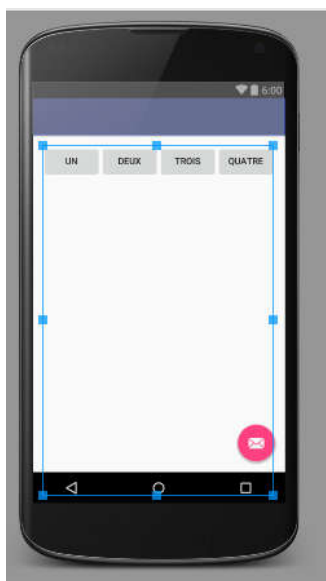
Component Tree

- Shown in @layout/activity_main
 - RelativeLayout
 - LinearLayout (vertical)
 - btn1 (Button) - "Un"
 - btn2 (Button) - "Deux"
 - btn3 (Button) - "trois"
 - btn4 (Button) - "quatre"

Properties

layout:toEndOf	
layout:toStartOf	
layout:alignComponent	[]
layout:alignParent	[]
layout:centerInParent	
style	
orientation	vertical
gravity	[]
accessibilityLiveRegion	
accessibilityTraversalAfter	
accessibilityTraversalBefore	

LinearLayout



Dr. Sabri GHAZI

Component Tree

- Shown in @layout/activity_main
 - RelativeLayout
 - LinearLayout (horizontal)
 - btn1 (Button) - "Un"
 - btn2 (Button) - "Deux"
 - btn3 (Button) - "trois"
 - btn4 (Button) - "quatre"

Properties

layout:toEndOf	
layout:toStartOf	
layout:alignComponent	[]
layout:alignParent	[]
layout:centerInParent	
style	
orientation	horizontal
gravity	[]
accessibilityLiveRegion	
accessibilityTraversalAfter	
accessibilityTraversalBefore	

Le RelativeLayout

- Comme son nom l'indique il permet de positionner les éléments de l'interface relativement les uns par rapport aux autres.
- On peut, par exemple, spécifier le fait qu'un composant **X** soit positionner en dessous du gauche d'un autre composant **Y**.

Dr. Sabri GHAZI

RelativeLayout

- Les propriétés principales:
 - `android:layout_toRightOf`
 - `android:layout_toLeftOf`
 - `android:layout_below`
 - `android:layout_above`
 - `android:layout_alignTop`, `lyout_alignButtom`,
`layout_alignLeft`, `layout_alignRight`,
`layout_alignBaseline`
- Il est à noter, que ces propriétés prennent les Id des éléments qu'on positionne notre composant.

Dr. Sabri GHAZI

Exemple

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">

    <TextView android:id="@+id/usernameLbl"
        android:layout_width="fill_parent" android:layout_height="wrap_content"
        android:text="Username:"
        android:layout_alignParentTop="true" />

    <EditText android:id="@+id/usernameText"
        android:layout_width="fill_parent" android:layout_height="wrap_content"
        android:layout_toRightOf="@id/usernameLbl" />

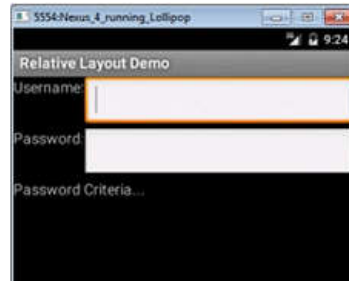
    <TextView android:id="@+id/pwdLbl"
        android:layout_width="wrap_content" android:layout_height="wrap_content"
        android:layout_below="@id/usernameText"
        android:layout_toRightOf="@id/usernameText" />

    <EditText android:id="@+id/pwdText"
        android:layout_width="fill_parent" android:layout_height="wrap_content"
        android:layout_toRightOf="@id/pwdLbl"
        android:layout_below="@id/usernameText" />

    <TextView android:id="@+id/pwdCriteria"
        android:layout_width="fill_parent" android:layout_height="wrap_content"
        android:layout_below="@id/pwdText"
        android:layout_toRightOf="@id/pwdText" />

    <TextView android:id="@+id/disclaimerLbl"
        android:layout_width="fill_parent" android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:text="Use at your own risk..." />

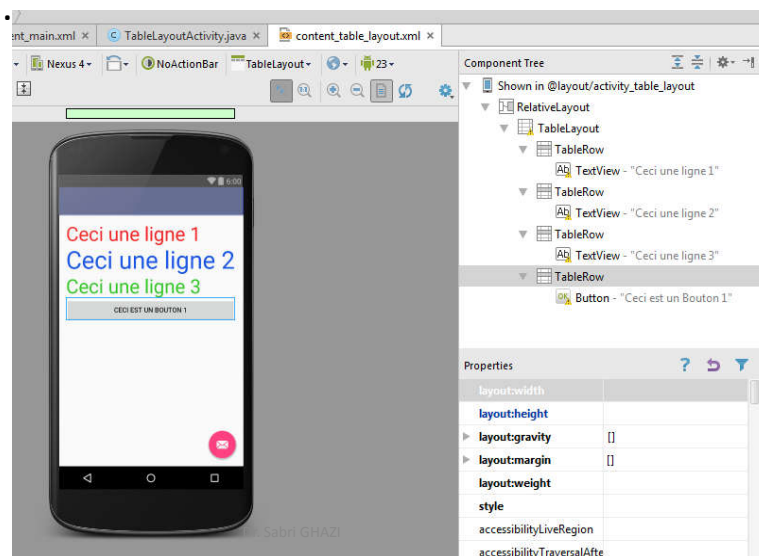
</RelativeLayout>
```



Dr. Sabri GHAZI

TableLayout

- Ce layout vous permet de positionner les éléments de l'interface sous forme d'une grille (tableau).

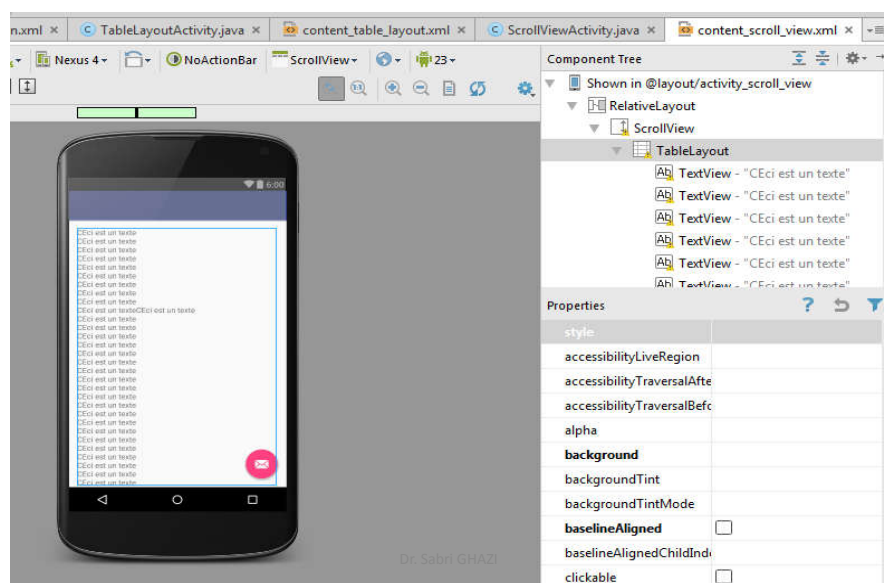


ScrollView

- Les écrans de téléphone sont généralement assez petits, ce qui rend la tâche difficile aux développeurs.
- De ce fait on peut employer quelques techniques qui permettent d'afficher les informations d'une façon interactive.
- C'est-à-dire des parties ne s'affichent que lorsque l'utilisateur fait une interaction.
- Le **ScrollView** est un **Layout** qui offre un défilement à son contenu

Dr. Sabri GHAZI

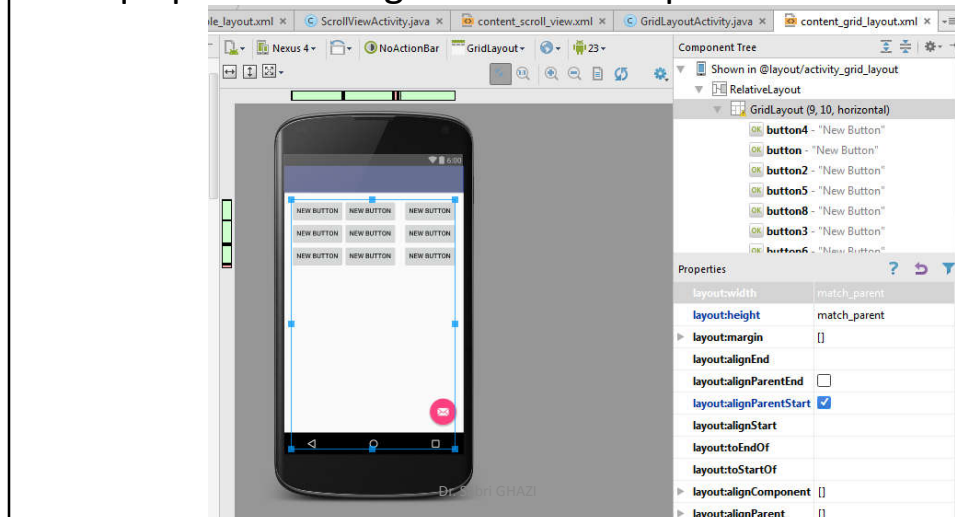
ScrollView



Dr. Sabri GHAZI

GridLayout

- Permet de placer les éléments sous forme d'un tableau avec des lignes et des colonnes.
- Ce qui permet un agencement très précis.



En code XML

```
<GridLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_alignParentStart="true"
    android:columnCount="10"
    android:rowCount="9">

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Button"
        android:id="@+id/button4"
        android:layout_row="1"
        android:layout_column="4" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Button"
        android:id="@+id/button"
        android:layout_row="2"
        android:layout_column="1" />
```

Dr. Sabri GHAZI

Questions ?

Dr. Sabri GHAZI