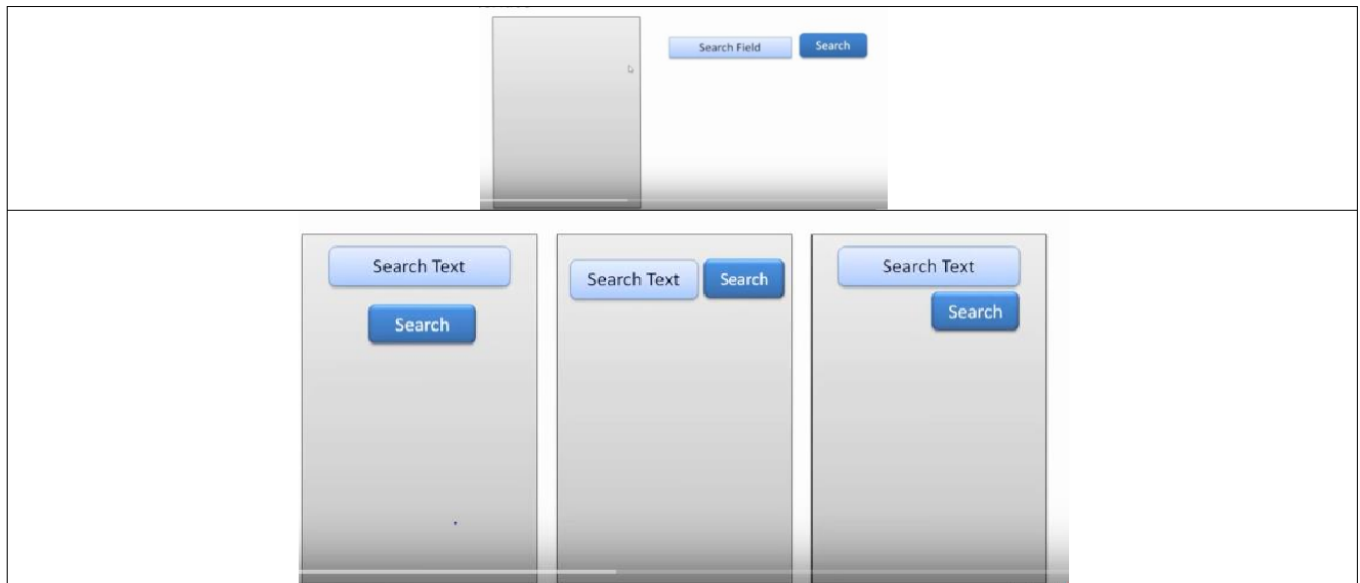
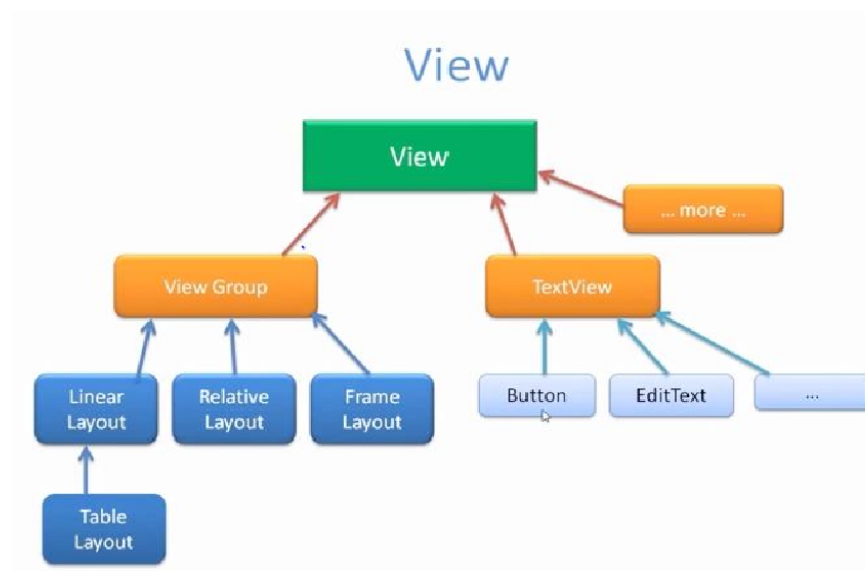


ACTIVITE N°1 : CREATION DES INTERFACES GRAPHIQUES INTERACTIVES INTEGRANT LES DIFFERENTS COMPOSANTS ANDROID ET LA GESTION EVENEMENTIELLE

A-Rappel



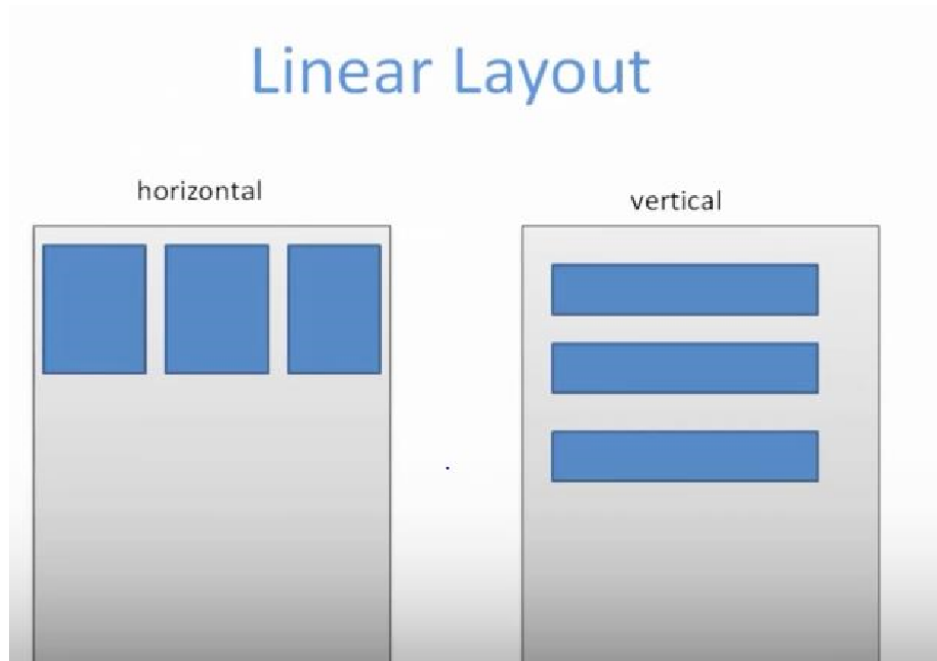
1- La classe VIEW



2- Layout

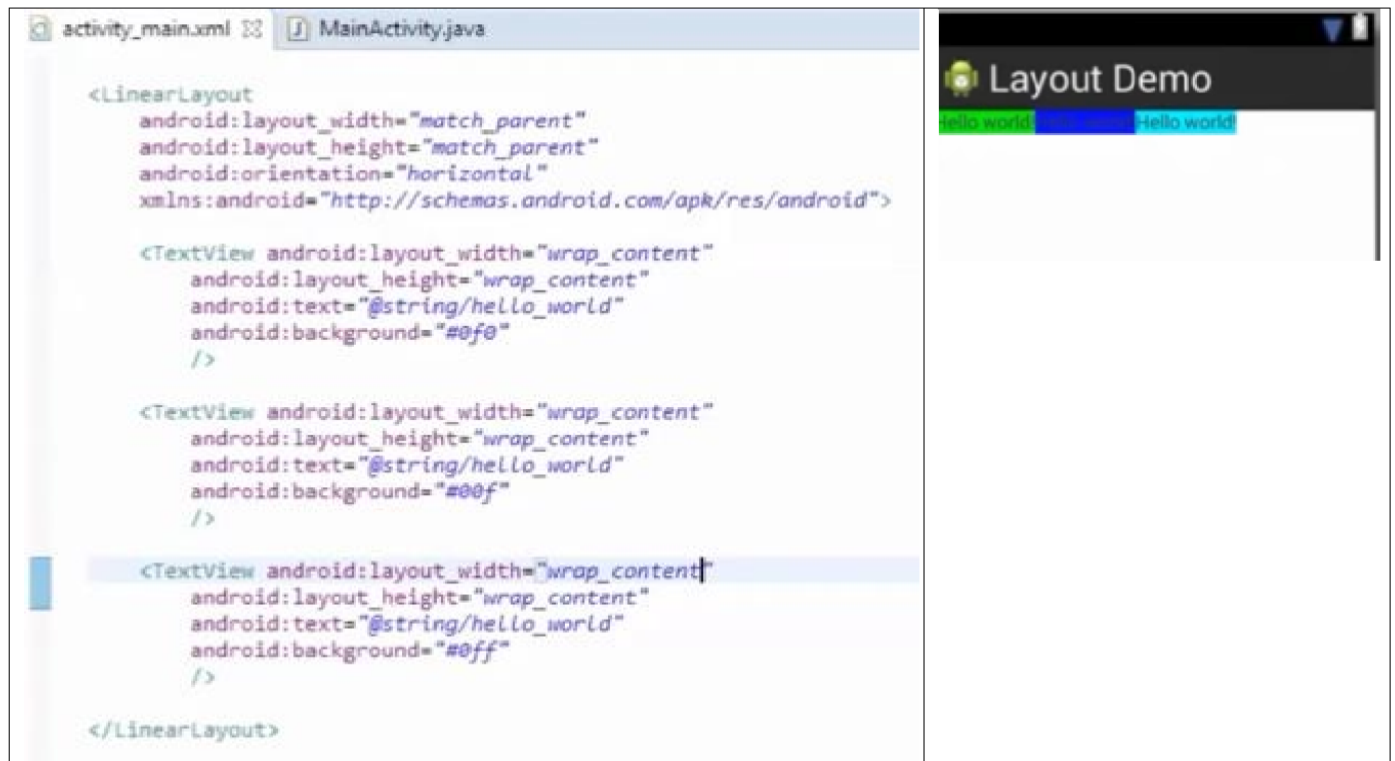
a. Les différents types de Layout

Cas 1 : Linear Layout



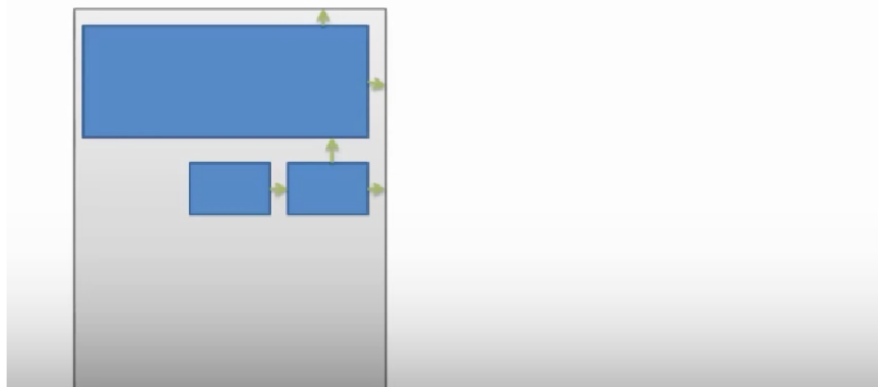
Exemples :

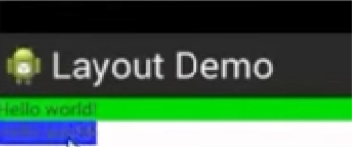

Code	Résultat
<pre> activity_main.xml MainActivity.java <LinearLayout android:layout_width="match_parent" android:layout_height="match_parent" android:orientation="vertical" xmlns:android="http://schemas.android.com/apk/res/android"> <TextView android:layout_width="match_parent" android:layout_height="wrap_content" android:text="@string/hello_world" android:background="#0f0" /> <TextView android:layout_width="match_parent" android:layout_height="wrap_content" android:text="@string/hello_world" android:background="#00f" /> <TextView android:layout_width="match_parent" android:layout_height="wrap_content" android:text="@string/hello_world" android:background="#0ff" /> </LinearLayout> </pre>	<p>The screenshot shows the application's output. It has a title bar 'Layout Demo'. Below the title bar, there are three horizontal bars. The top bar is green and contains the text 'Hello world!'. The middle bar is blue and contains the text 'Hello world!'. The bottom bar is cyan and contains the text 'Hello world!'. A mouse cursor is pointing at the middle bar.</p>



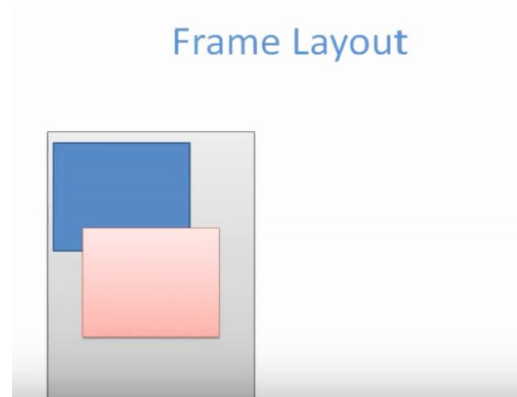
Cas 2 : Relative Layout

Relative Layout



Code	Résultat
<div> <div>activity_main.xml</div> <div>MainActivity.java</div> <div> <pre> <RelativeLayout android:layout_width="match_parent" android:layout_height="match_parent" xmlns:android="http://schemas.android.com/apk/res/android" <TextView android:text="@string/hello_world" android:layout_width="match_parent" android:layout_height="wrap_content" android:background="#00f0" android:id="@+id/view1" /> <TextView android:text="@string/hello_world" android:layout_width="wrap_content" android:layout_height="wrap_content" android:background="#00f" android:layout_below="@id/view1" /> </RelativeLayout> </pre> </div> </div>	
<div> <div>*activity_main.xml</div> <div>MainActivity.java</div> <div> <pre> <RelativeLayout android:layout_width="match_parent" android:layout_height="match_parent" xmlns:android="http://schemas.android.com/apk/res/android" <TextView android:text="@string/hello_world" android:layout_width="match_parent" android:layout_height="wrap_content" android:background="#00f0" android:id="@+id/view1" android:layout_margin="10dp" /> <TextView android:text="@string/hello_world" android:layout_width="wrap_content" android:layout_height="wrap_content" android:background="#00f" android:layout_below="@id/view1" android:layout_alignRight="@id/view1" /> </RelativeLayout> </pre> </div> </div>	

Cas 3 : Frame Layout



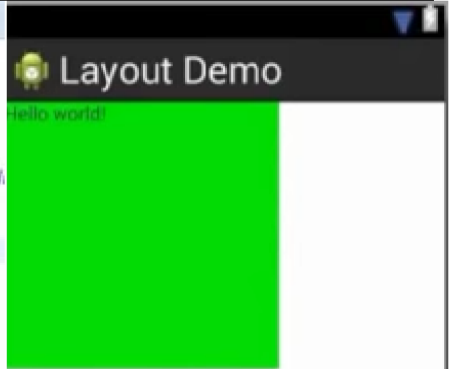
Code	Résultat
<pre>activity_main.xml MainActivity.java <FrameLayout android:layout_width="match_parent" android:layout_height="match_parent" <TextView android:background="#0f0" android:layout_width="300dp" android:layout_height="200dp" /> <TextView android:background="#00f" android:layout_width="200dp" android:layout_height="200dp" /> </FrameLayout></pre>	

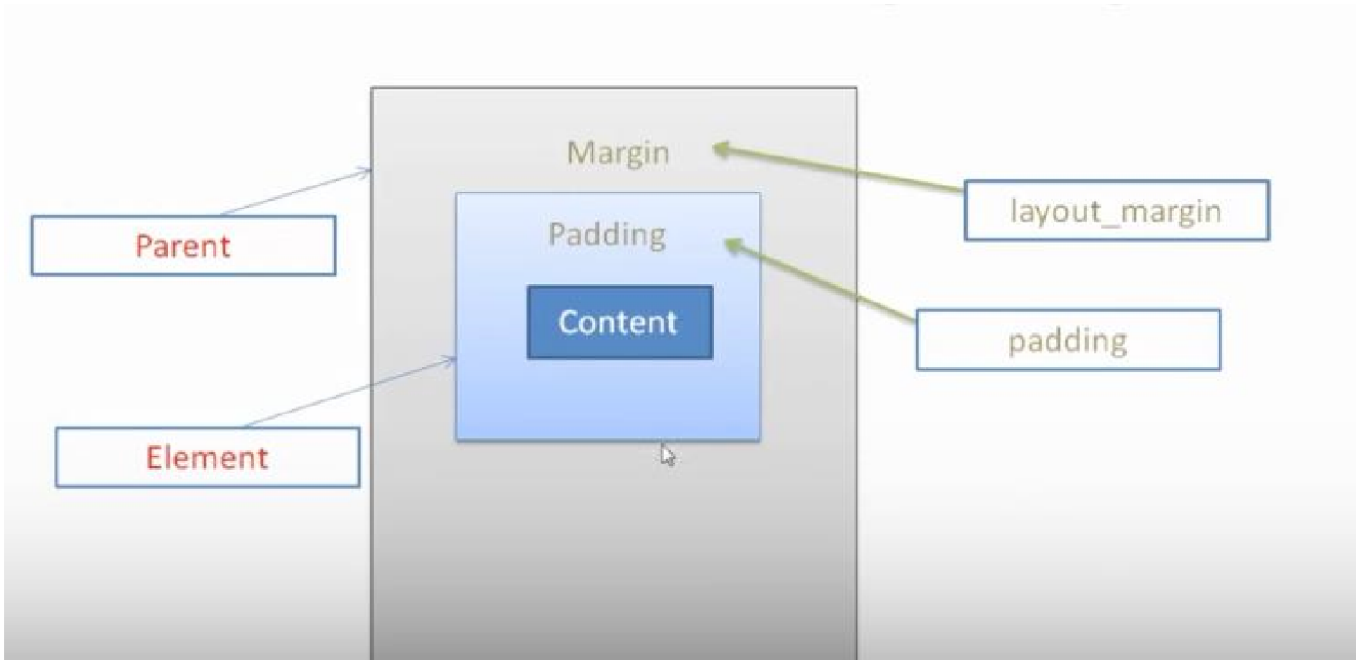
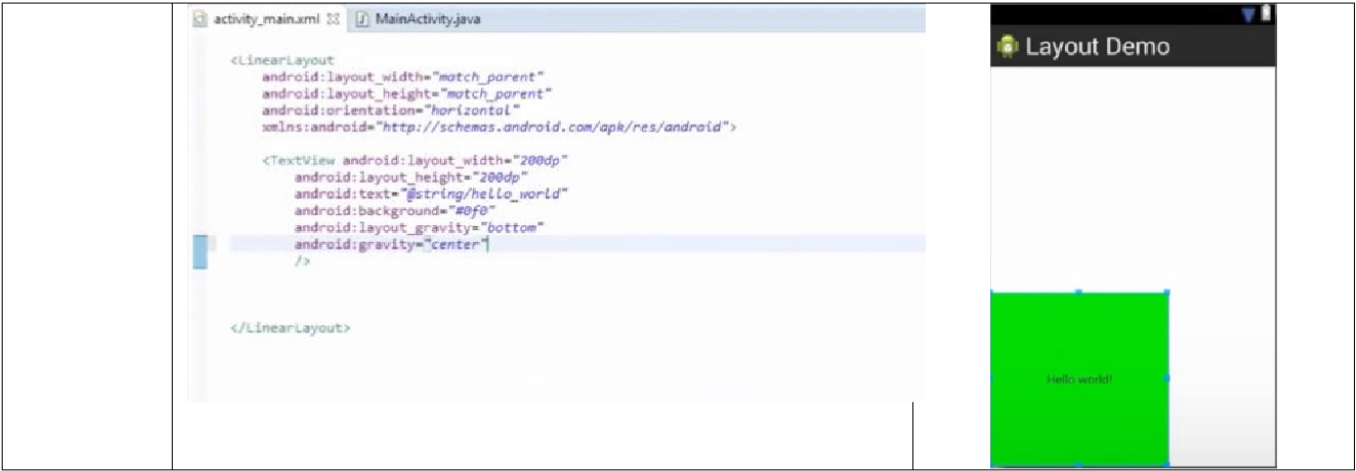
b. Les attributs de Layout

- Size
- Gravity
- Margin and Padding

Attribut	Explication/Code	Résultat
Size	<ol style="list-style-type: none"> 1. match_parent 2. wrap_content 3. in dp <p>dp: Density-Independent Pixel Calculating Pixels: $px = dp * \text{density ratio of the device.}$</p> <p>Device Density Ratio: ldpi = 0.5 mdpi = 1 hdpi = 1.5 xhdpi = 2</p>	<p>px vs dp</p> <p>Input in Pixels Width=100px Height= 100px</p> <p>mdpi Device 100px by 100px</p> <p>xhdpi Device 100px by 100px</p> <p>px vs dp</p> <p>Input in dp Width=100dp Height= 100dp</p> <p>mdpi Device Density ratio : 1 $px = 100 * 1$ 100px by 100px</p> <p>xhdpi Device Density ratio : 2 $px = 100 * 2$ 200px by 200px</p>

Gravity	<p>1. layout_gravity</p> <ul style="list-style-type: none"> Position of the Element with respect to its Layout (parent) <p>2. gravity</p> <ul style="list-style-type: none"> Position of the Content within the view 	<p>px vs dp</p> <p>Input in Pixels Width=100px Height= 100px</p> <div> <div>mdpi Device</div> <div>100px by 100px</div> </div> <div> <div>xhdpi Device</div> <div>100px by 100px</div> </div> <p>px vs dp</p> <p>Input in dp Width=100dp Height= 100dp</p> <div> <div>mdpi Device Density ratio : 1</div> <div>px = 100 * 1</div> <div>100px by 100px</div> </div> <div> <div>xhdpi Device Density ratio : 2</div> <div>px = 100 * 2</div> <div>200px by 200px</div> </div>

Gravity	<p>activity_main.xml MainActivity.java</p> <pre> <LinearLayout android:layout_width="match_parent" android:layout_height="match_parent" android:orientation="horizontal" xmlns:android="http://schemas.android.com/apk/res/and <TextView android:layout_width="200dp" android:layout_height="200dp" android:text="@string/hello_world" android:background="#0f0" /> </LinearLayout> </pre>	



B-TRAVAIL A FAIRE

Exercice n°1

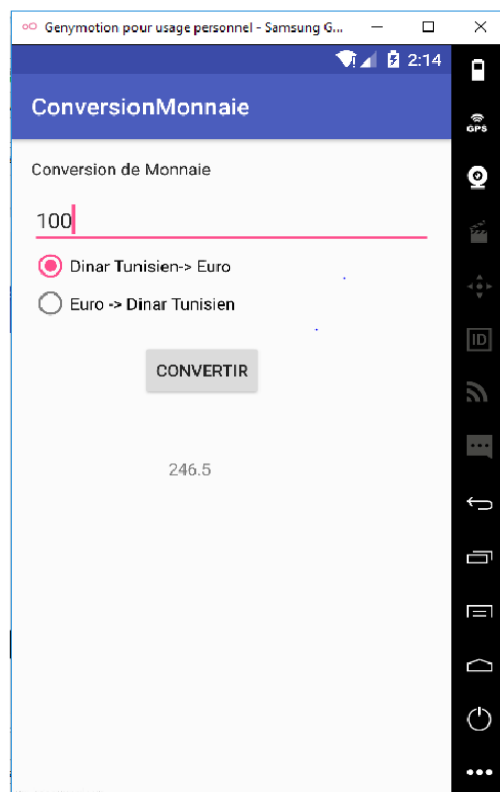
Le but de cet exercice est de construire une interface Homme Machine (IHM) présentant plusieurs composants graphiques Android (View). Cette IHM permet de faire une authentification en demandant un couple (nom de login, un mot de passe). Après avoir appuyé sur le bouton connecter, un Toast indique si le couple donné convient ou pas.



- 1- Ecrire le code XML permettant de créer l'IHM ci-dessus.
- 2- Ecrire le code Java qui, lorsque le bouton Connecter est actionné par l'utilisateur, une vérification d'authentification est faite et le Toast affiche le résultat (bon couple login, mot de passe ou pas)
- 3- Ecrire le code Java qui, lorsque le bouton Connecter est actionné par l'utilisateur, si le login et le mot de passe sont corrects, une nouvelle activité est affichée indiquant une bonne connexion.

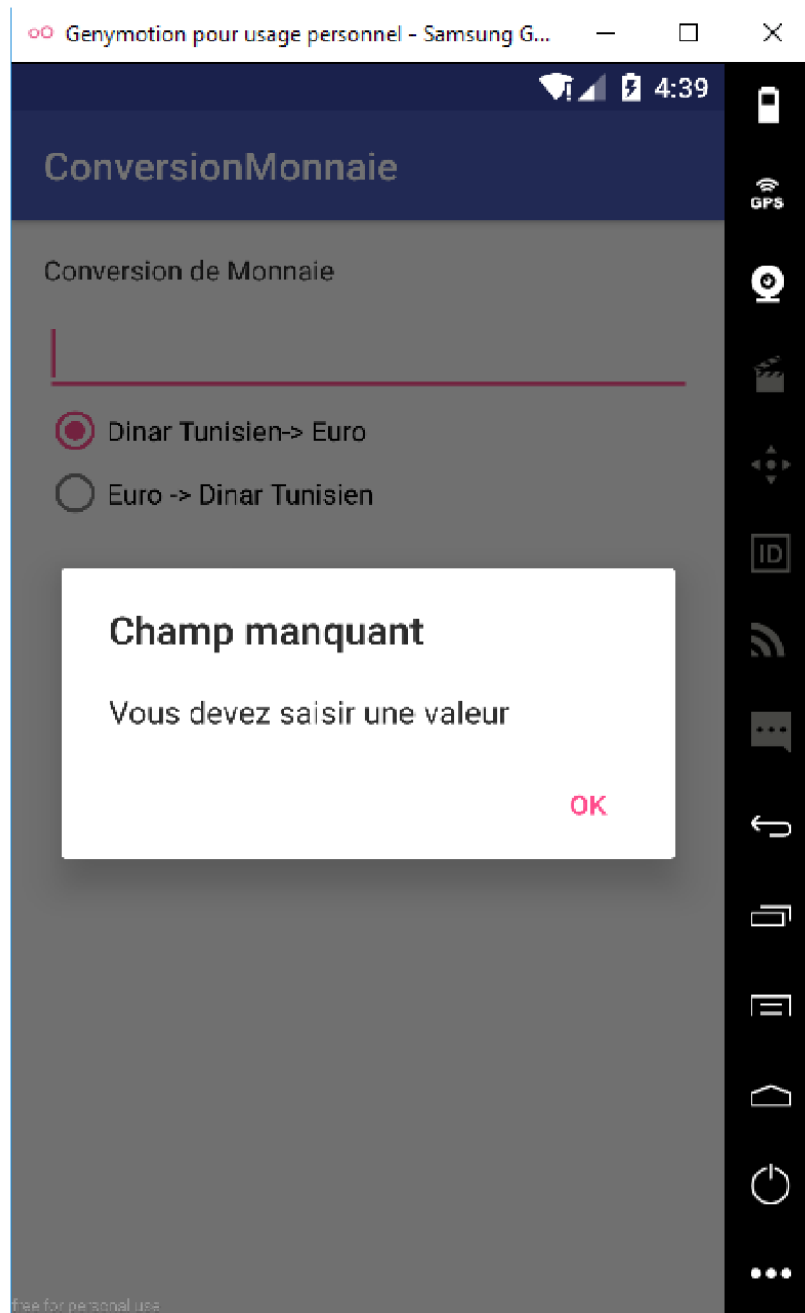
Exercice n°2

On vous demande de réaliser progressivement une application de conversion de monnaie du Dinars en Euro et de l'Euro vers le dinar.



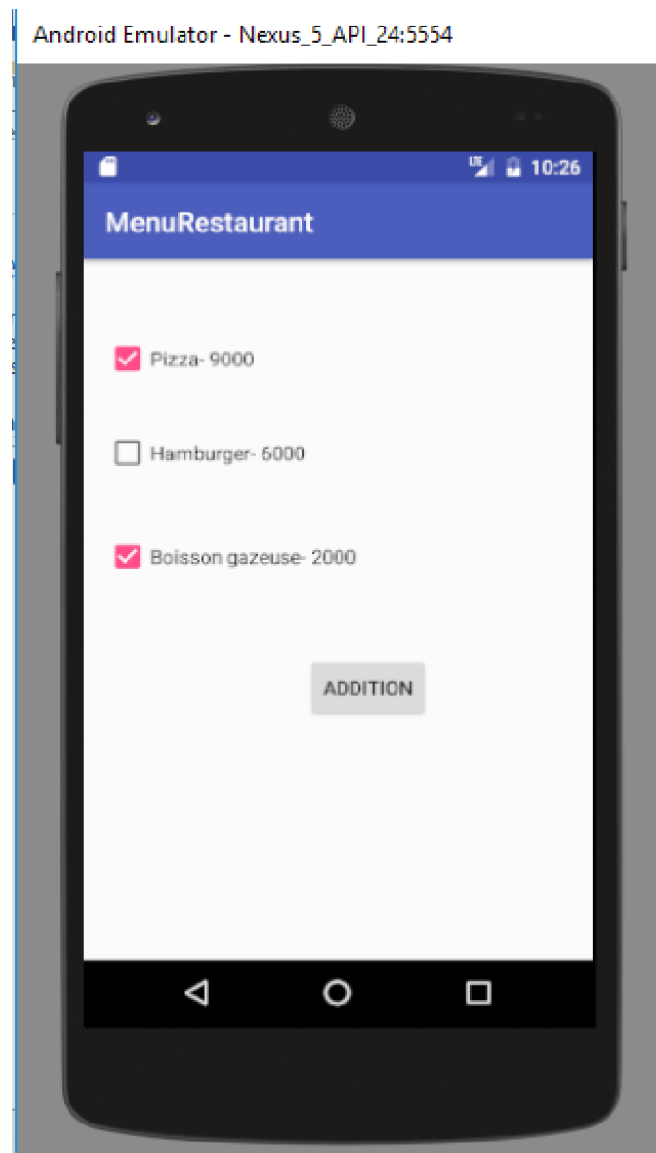
Pour ce faire, on vous demande :

- 1- Ecrire le code XML permettant de créer l'IHM ci-dessus.
- 2- Ecrire le code Java du bouton « Convertir » qui permet de calculer et d'afficher le nouveau montant qui sera calculé selon le choix de l'utilisateur. Une vérification de la saisie de donnée est faite (un message d'alerte sera affichée à l'utilisateur si le montant de la conversion est vide en utilisant la classe `AlertDialog.Builder`)



Exercice n°3

On vous demande de réaliser progressivement une application de calcul de recette d'un client dans un restaurant. Pour ce faire, on vous demande de concevoir l'interface graphique suivante :



Pour ce faire, on vous demande :

- 1- Ecrire le code XML permettant de créer l'IHM ci-dessus.
- 2- Ecrire le code Java du bouton « ADDITION » qui permet de calculer et d'afficher le montant de l'addition.