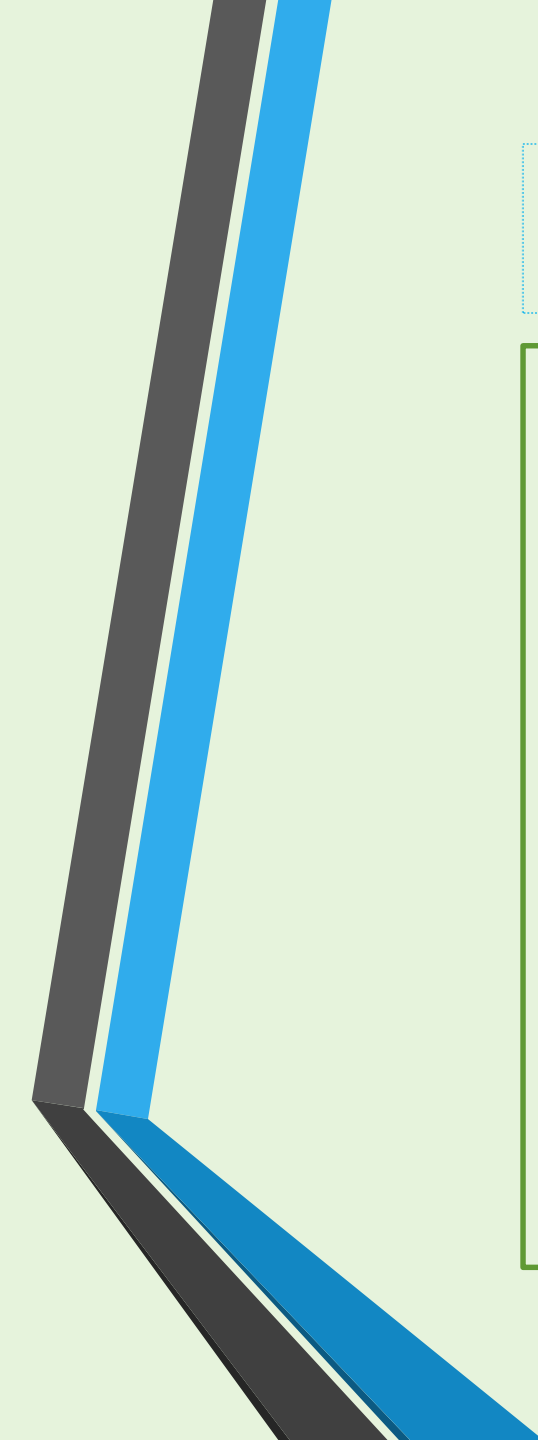


# Les vues de base d'Android

Formateur : Mohammed LAMNAOUR





# Sommaire

- Les messages TOASTS
- Button
- TextView
- EditText

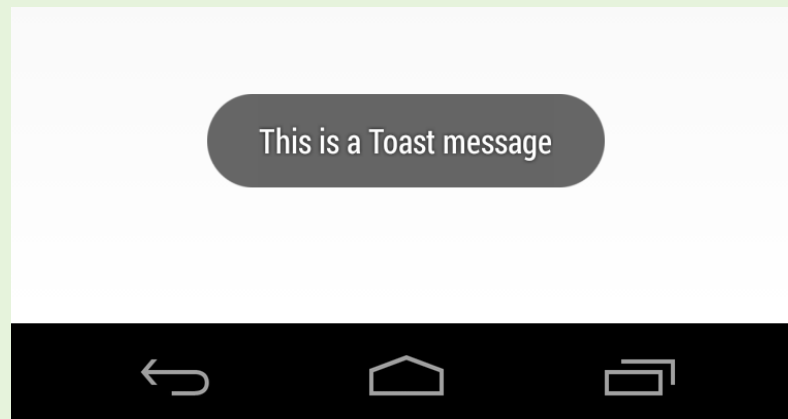


# Messages Toast

Développement Mobile

# Les messages TOAST

- Un «**toast**» est un message apparaissant en bas d'écran pendant un instant, par exemple pour confirmer la réalisation d'une action.



# Les messages TOAST

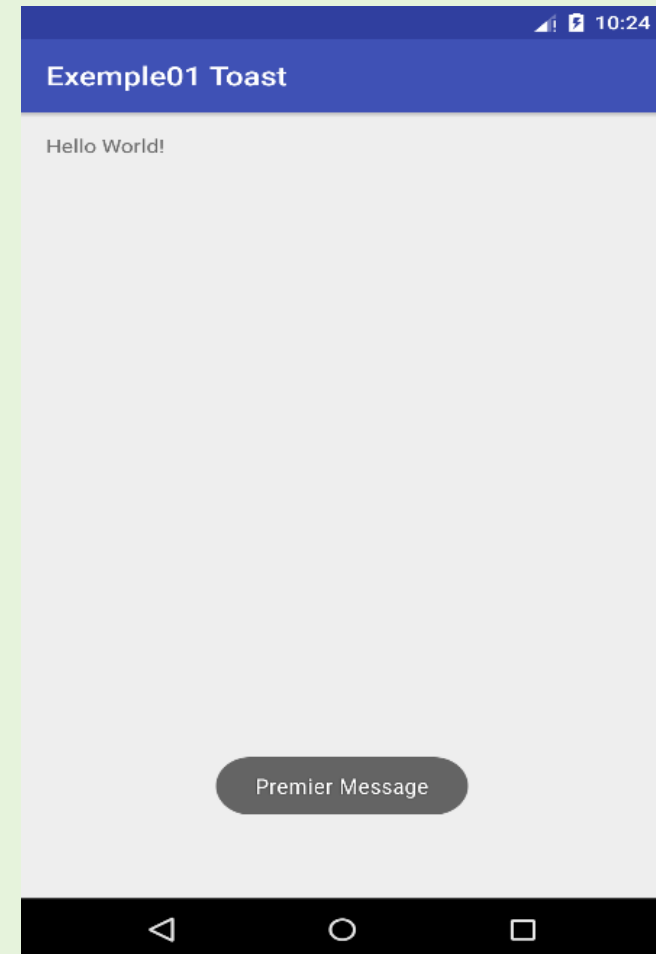
- Pour afficher un Toast nous avons besoin de trois paramètres :
  - Le contexte de notre application : Les deux valeurs possible pour ce champ sont : **this** ou **getApplicationContext()**
  - Le texte à afficher
  - La durée d'affichage : les valeurs possibles sont :
    - Toast.LENGTH\_SHORT (2 secondes)
    - Toast.LENGTH\_LONG (5 secondes).

```
Toast.makeText(this, "Premier Message", Toast.LENGTH_SHORT).show();
```

# Les messages TOAST : Exemple

Exemple qui affiche un message au chargement de l'activité :

```
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        Toast.makeText(this, "Premier  
Message", Toast.LENGTH_SHORT).show();  
    }  
}
```





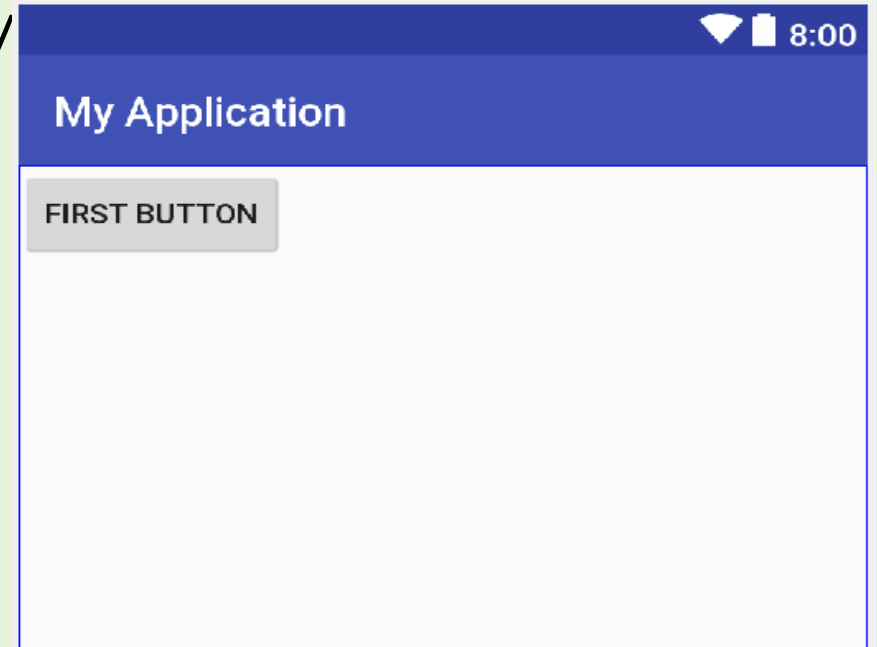
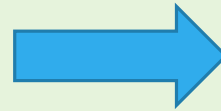
# Button

Développement Mobile

# Button

- Un button est une vue qui peut être pressé ou cliqué par l'utilisateur d'effectuer une action ou un traitement.
- La syntaxe de déclaration d'un button en XM

```
<Button  
    android:id="@+id/simpleButton"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="First Button"/>
```





# Button – Les principales propriétés

- **Id** : Identifiant du contrôle. Il doit être unique dans le l'activité.
- **Gravity** : Permet de spécifier l'alignement du texte dans le contrôle.
- **Text** : Le texte qui sera affiche dans le contrôle.
- **TextColor** : La couleur du texte du contrôle

```
<Button  
    android:id="@+id/simpleButton"  
    android:layout_width="fill_parent"  
    android:layout_height="wrap_content"  
    android:text="First Button"  
    android:layout_centerInParent="true"  
    android:gravity="right|center_vertical"  
>
```

# Button – Les principales propriétés

- **TextStyle** : Cet attribut peut contenir les valeurs suivantes :
  - 0 : normal
  - 1 : Bold
  - 2 : Italic
- **Background** : Modifier la couleur de l'arrière plan du contrôle.
- **drawableBottom** : Permet de spécifier l'image en bas du button.
- **drawableTop, drawableRight And drawableLeft.**

# Button – Les principales propriétés

- Vous pouvez modifier les propriétés du contrôle a l'aide du code java, comme le montre l'exemple suivant :

```
Button btn;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    btn = (Button) findViewById(R.id.simpleButton);

    btn.setText("Hello");
    btn.setTextColor(Color.BLUE);
    btn.setTextAlignment(0);
}
```

# Button – Les évènements

- Il existe 2 méthodes pour gérer l'évènement clique d'un bouton :
  - OnClick dans le fichier xml Layout
  - Utiliser l'interface OnClickListener

# OnClick dans le fichier XML

## Dans le fichier XML Layout

```
<Button  
    android:id="@+id/simpleButton"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="First Button"  
    android:onClick="test"  
>
```

## Dans le fichier Java

```
public void test(View view) {  
  
}
```

- La fonction doit être publique
- Ne retourne aucune valeur
- Accepte comme paramètre un objet de Type View

# OnClickListner – Création d'une classe Anonyme

```
public class MainActivity extends AppCompatActivity {  
  
    Button t;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        t = (Button) findViewById(R.id.b1);  
  
        t.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
                // Code du Button.....  
            }  
        });  
    }  
}
```

# OnClickListner – Implémenter l'interface

```
public class MainActivity extends AppCompatActivity implements
Button.OnClickListener{

    Button t;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        t = (Button) findViewById(R.id.b1);
        t.setOnClickListener(this);

    }

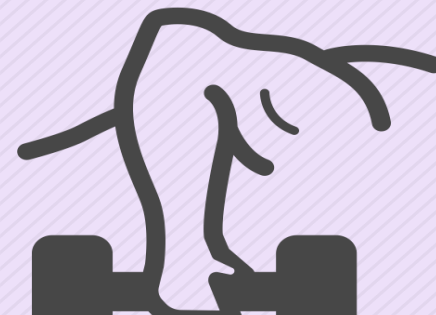
    @Override
    public void onClick(View v) {

    }

}
```

# Exercices d'application

- **Exercice 1** : Créer une application qui contient seulement un button qui va afficher un Toast avec le texte « Premier Exemple Button/Toast»
- **Exercice 2** : Créer une application qui affiche un message Toast le nombre de clique sur un button. (L'activité contient un seul button).







# TextView

Développement Mobile

# TextView

- Un TextView peut être défini comme un titre ou bien un paragraphe se situant à l'intérieur d'une page d'une application mobile Android.
- Un TextView est utilisé pour afficher sur une page une information à un utilisateur. Cela vous permet de mettre de plus un titre à des images ou des descriptions.

# TextView

- Voici un exemple d'accès à un TextView en utilisant le code JAVA. Le TextView possède le code t1.

```
public class MainActivity extends AppCompatActivity {  
  
    TextView t;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        t = (TextView) findViewById(R.id.t1);  
        t.setText("TDI - Module de développement");  
        t.setTextColor(Color.GREEN);  
        t.setBackgroundColor(Color.GRAY);  
    }  
}
```

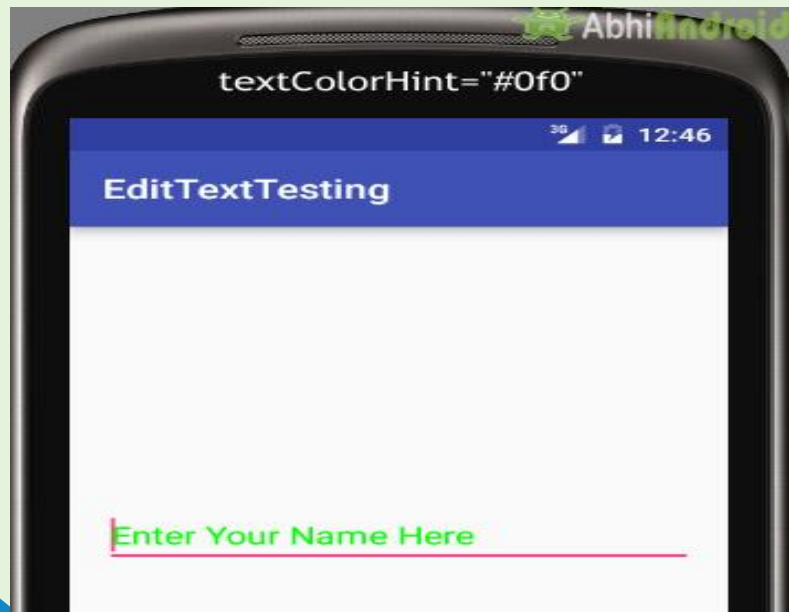


# EditText

Développement Mobile

# EditText

- Permet de spécifier des zones de saisis dans vos activités.
- Les principales propriétés de ce vue est :
  - Hint : Permet de saisir dans une zone de texte une explication. Exemple :



# EditText – Les principales Propriétés

<b>Android :hintColor</b>	Permet de spécifier un couleur au texte <b>hint</b>
<b>Android :InputType</b>	Permet de spécifier le type de texte : <ul style="list-style-type: none"><li>- Number</li><li>- Text</li><li>- Phone</li><li>- ...</li></ul>

# EditText

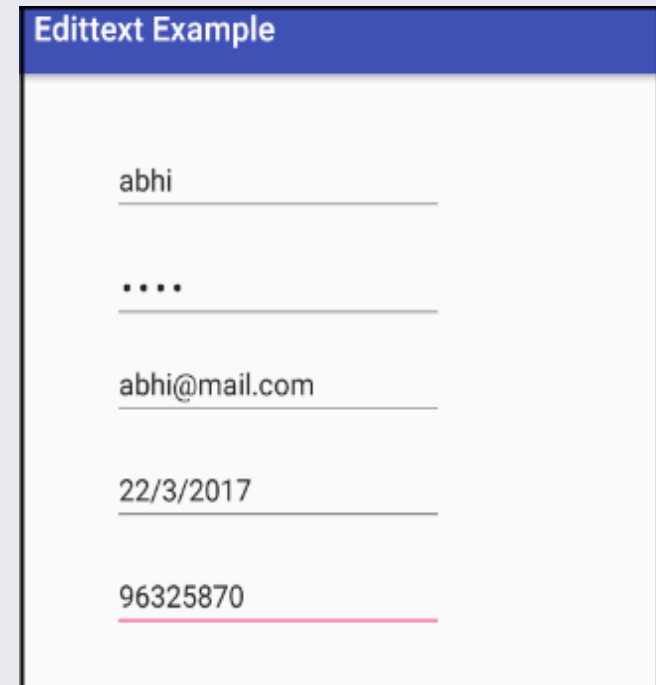
- Voici un exemple d'accès à un TextView en utilisant le code JAVA. Le TextView possède le code t1.

```
public class MainActivity extends AppCompatActivity {  
  
    EditText t;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        t = (EditText) findViewById(R.id.t1);  
  
    }  
}
```

# Exercices d'application

## Exercice 1 :

- Créer une application sous Android studio.
- Créer l'IHM suivant :



Edittext Example

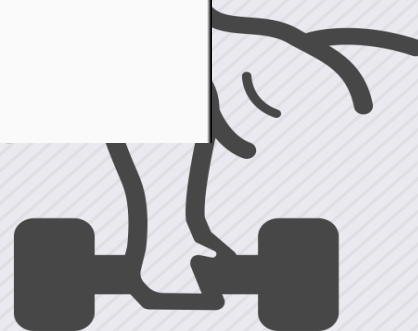
abhi

\*\*\*\*

abhi@mail.com

22/3/2017

96325870

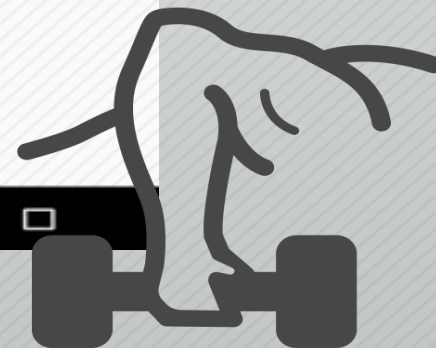
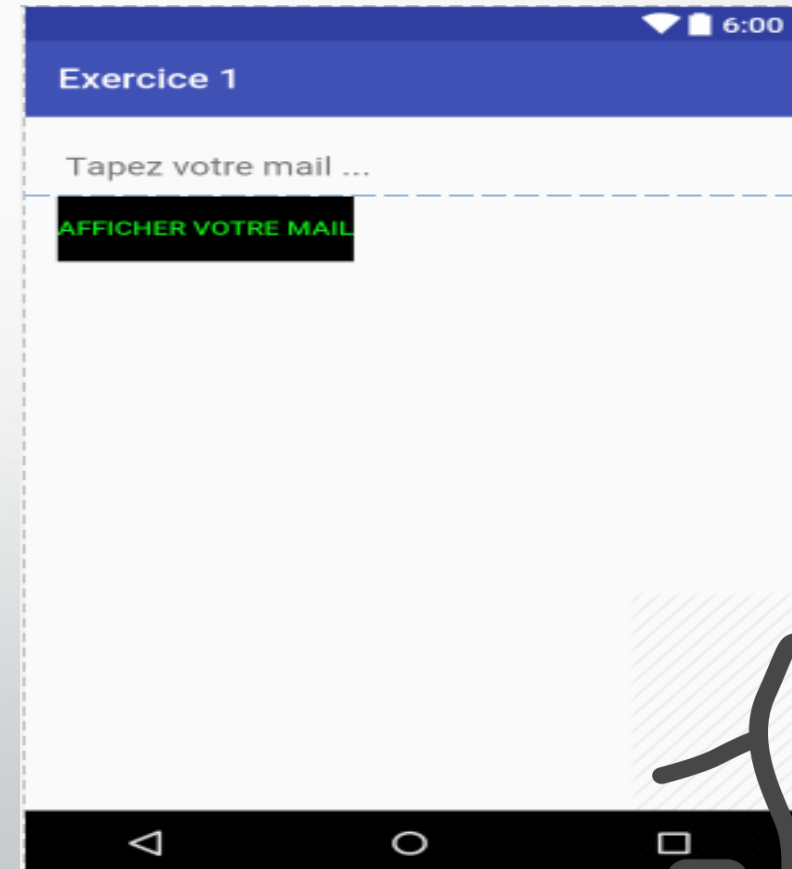




# Exercices d'application

## Exercice 2 :

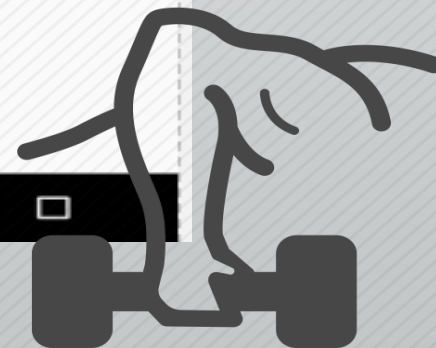
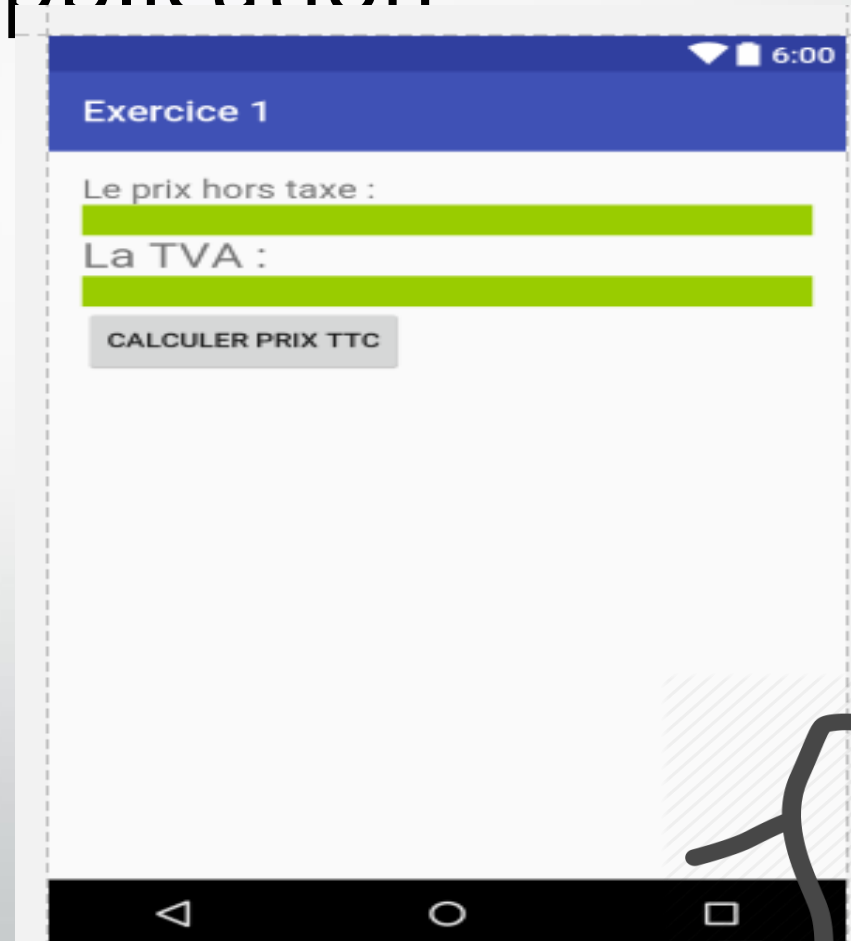
1. Créer une application sous android studio.
2. Créer l'interface suivante :
3. Ajouter un événement click au button qui permet d'afficher un Toast qui contient le mail saisi.



# Exercices d'application

## Exercice 3 :

1. Créer une application sous android studio.
2. Créer l'interface suivante :
3. Ajouter un événement au button, qui permet d'afficher le message TVA le prix TTC dans un message Toast, sous la forme **PRIX TTC = 12.45 DH.**
4. Modifier l'activite precedente pour afficher le prixTTC dans un TextView



# Exercices d'application

## Exercice 4 :

- Créer une application sous android studio
- Créer l'interface suivant :
- Ajouter le code du button qui permet d'afficher Connection OK si le login saisi est « toto » et le mot de passe « toto ». sinon le message « Connection KO » sera affiché



The screenshot shows an Android application interface with a black background. At the top, there is a title bar labeled "AuthentificationProjet". Below the title bar, the text "Enter votre nom et votre mot de passe" is displayed. There are two input fields: the first contains the text "toto", and the second is a password field with six dots and a cursor. Below the input fields is a button labeled "Connecter". At the bottom of the screen, there is a button labeled "Connection OK".

# Exercices d'application

## Exercice 5 :

- En se basant sur l'ancien projet de l'exercice 3.
- Ajouter à votre projet la classe Utilisateur qui contient les éléments suivant :
  - Les attributs : login et password
  - Les accesseurs
  - Constructeur d'initialisation et par défaut
- Dans la onCreate de la classe java de l'activité remplir une liste d'utilisateur de votre choix.
- Modifier l'événement du bouton pour afficher Connection OK si le login et le mot de passe existe dans la liste sinon le message « Connection KO » sera affiché.

# Exercices d'application

## Exercice 6 :

1. Créer une application sous Android studio.
2. Créer l'interface suivant :
3. Ecrire le code des buttons

The screenshot shows an Android application titled "Exercice01". The interface includes two input fields labeled "Nombre 1 :" and "Nombre 2 :". Below these are four buttons labeled "SOMME", "SOUSTRACTION", "PRODUIT", and "DIVISION". At the bottom, there is a label "Resultat :" followed by a large empty text area for the result. The status bar at the top shows the time as 11:36. The bottom navigation bar shows the standard Android navigation icons.