



Utilisation des objets
xml de type
EditText



EditText

The screenshot displays the Android Studio interface. The top panel is the 'Palette', which is divided into categories on the left and a list of widgets on the right. The 'Text' category is selected, and 'Ab Plain Text' is highlighted. A mouse cursor is positioned over the 'Ab Plain Text' widget. Below the Palette is the 'Component Tree' panel, which shows a single component: 'ConstraintLayout'. The interface includes search, settings, and collapse icons in the top right of each panel.

Category	Widget
Common	Ab TextView
Text	Ab Plain Text
	Ab Password
Buttons	Ab Password (Numeric)
Widgets	Ab E-mail
Layouts	Ab Phone
Containers	Ab Postal Address
Google	Ab Multiline Text
	Ab Time
Legacy	Ab Date
	Ab Number
	Ab Number (Signed)

Component Tree

- ConstraintLayout

Android Studio interface showing a text input field with constraints.

Palette:

- Common
 - Ab TextView
- Text
 - Ab Plain Text
- Buttons
 - Ab Password
 - Ab Password (Numeric)
- Widgets
 - Ab E-mail
- Layouts
 - Ab Phone
- Containers
 - Ab Postal Address
- Google
 - Ab Multiline Text
- Legacy
 - Ab Time
 - Ab Date
 - Ab Number
 - Ab Number (Signed)

Component Tree:

- ConstraintLayout
 - Ab editText2- "Name"

Design View:

The design view shows a text input field with the text "Name". The field is centered horizontally and vertically. It is connected to the parent container (ConstraintLayout) by four blue wavy constraint lines: one to the top center, one to the bottom center, one to the left edge (labeled "8"), and one to the right edge (labeled "8").

```
String strName = monEditTEXT.getText().toString();
```

on récupère un String strName grâce au getText(), mais getText est d'un genre particulier, il faut lui signaler le type par toString().

On peut faire un peu de contrôle de saisie avec un : `if (strName.equals(" ")) return;`

Pour que return soit "seul", il faut que la méthode qui l'englobe ne renvoie donc rien (void). Idéal dans un Public void onClick...

```
BTN1.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        .....  
        if (STRCFOND.equals("")) return;  
    }  
})
```

si on clique sur le bouton, et que notre chaîne de caractère est vide on fait une sortie via return. Du coup on peut contrôler s'il y a bien eu une saisie ou pas. Le choix de votre EditTexte déterminera le type de données que l'utilisateur peut saisir.

```
int maValeurEnInt = Integer.parseInt(monString);
```

Dans le cas ou vous devez transformer votre "saisie" (qui est de type String) en entier.

```
double monDouble = Double.parseDouble(monString);
```

Pour aller plus loin... <https://developer.android.com/reference/java/lang/Integer>

PARSE = Analyser, cette fonction sera souvent utile...



```
String pseudo = et_pseudo.getText().toString().trim();  
String email = et_email.getText().toString().trim();  
String password = et_password.getText().toString().trim();  
String password2 = et_password2.getText().toString().trim();
```

La méthode trim() permet d'enlever les espaces avant et après la saisie