

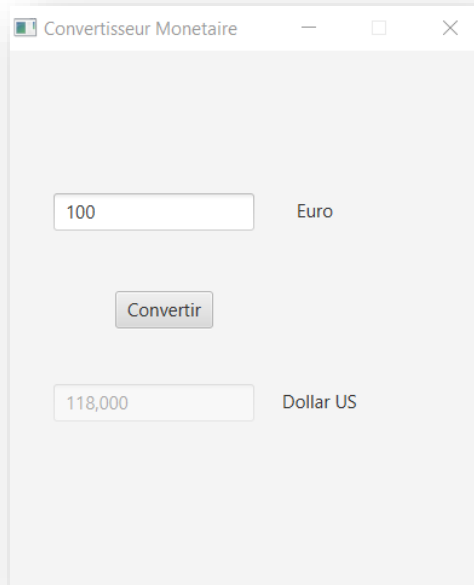


# CONVERTISSEUR MONETAIRE

## Java fx

### 1 - CONVERSION EURO - DOLLARD

Créer une application pour convertir une valeur monétaire en euros en dollars US. Le taux de conversion au 27/02/2022 est de 1,11.



Le calcul est déclenché par un click souris sur le Bouton

### 2 – STYLAGE DU BOUTON

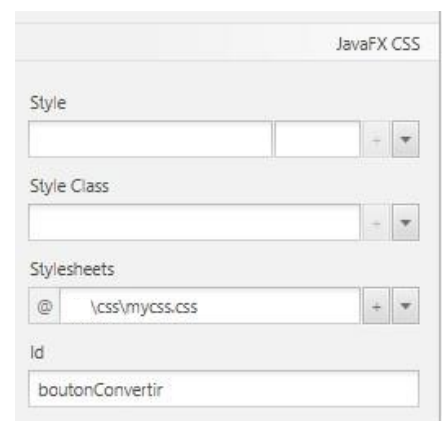
L'image associée au composant *Button* est une image *png* 40 x 40 (*convertir.png*) à stocker dans un dossier *src/images*

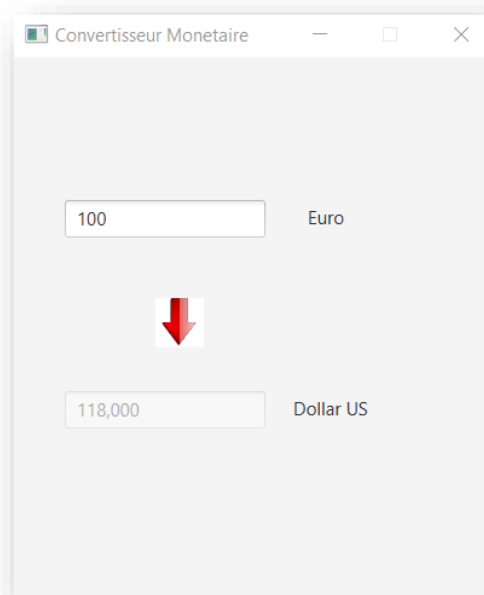
Le bouton est stylé par un fichier *css* externe.

- Créer un dossier *css* dans le dossier *src*
- Créer dans ce dossier *css* un fichier de type *css* (*cascading style sheet*).
- Le contenu de ce fichier définit les attributs de style du bouton.
- Contenu du fichier :

```
#bouttonConvertir{  
-fx-background-image:  
url(..\images\convertir.png);  
-fx-background-size: stretch;  
}
```

- Associer au bouton un nom de *Id* de style (*bouttonConvertir*) et le chemin au fichier *css*.



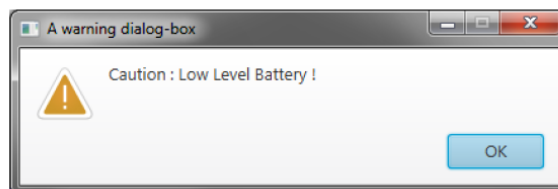


### 3 – AMELIORATION DE LA SECURITE

En cas de saisie d'une valeur non numérique, la conversion *String/Double* ne peut se faire et déclenche une exception de type *NumberFormatException*.

Modifier l'application en interceptant cette exception dans un bloc *try/catch* qui provoque l'affichage d'une fenêtre d'alerte.

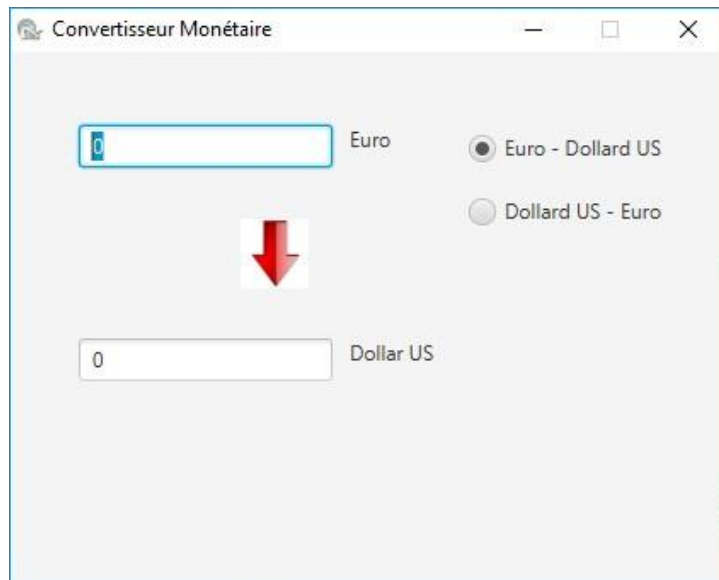
```
Alert dialogW = new Alert(AlertType.WARNING);  
  
dialogW.setTitle("A warning dialog-box");  
dialogW.setHeaderText(null); // No header  
dialogW.setContentText("Caution : Low Level Battery !");  
dialogW.showAndWait();
```



### 4 – CONVERSION BIDIRECTIONNELLE

Rajouter sur l'IHM deux *RadioButton* pour choisir le sens de conversion.

L'exclusivité entre les deux *RadioButton* est assurée en saisissant le même nom de *Toggle Group* dans leurs propriétés.



- Mettre en place un listener sur l'évènement *Action*
- Dans la méthode associée :
- Modifier le taux de conversion et l'affichage des labels

#### Remarque :

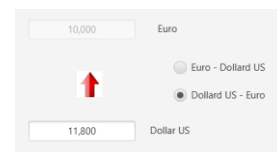
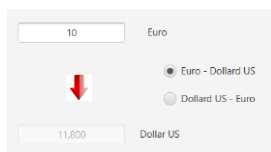
L'état d'un *RadioButton* est lié à la propriété *checked*

Exemple :

```
If(radioButtonEuroDollard.isSelected() == true){  
...  
}
```

Vous pouvez aussi réaliser une transition de type ***RotateTransition*** qui permettra de changer l'orientation de la flèche pour indiquer le sens de la convention.

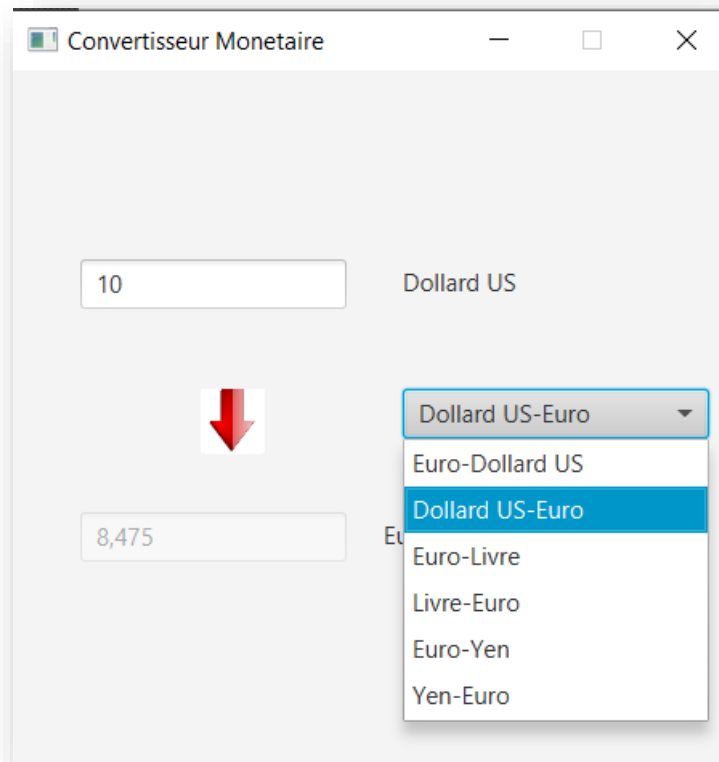
```
public RotateTransition rotation;  
...  
Dans initialize() :  
    rotation = new RotateTransition(Duration.seconds(0.5), buttonConversion);  
...  
Au moment d'appliquer la transition  
    rotation.setByAngle(180);  
    rotation.play();
```



## 5 - GENERALISATION

Généraliser en rajoutant d'autres devises.

Dans cette généralisation, les *radioButton* peuvent être remplacés par un *ComboBox*. L'évènement à écouter par le *listener* est *Action*



### Quelques règles d'utilisation du **ComboBox**.

Lorsque le *ComboBox* est créé avec le concepteur Graphique de *SceneBuilder*, On ne peut pas définir les données qui lui sont liées dans le concepteur graphique, mais dans le code du *controleur*.

Ces données sont regroupées sous la forme d'une collection *ObservableList* de *String*.

#### Remplissage des données:

```
comboSelection.getItems().add("une donnée");
```

- *comboSelection.getItems()* récupère la référence de la liste associée au *comboBox* *comboSelection*.
- La méthode *add* ajoute une *String* à la liste.

#### Initialisation du combobox:

```
comboSelection.setValue(comboSelection.getItems().get(0));
```

Initialisation avec la méthode *setValue* à la première valeur de la liste.

#### Récupération d'un valeur sélectionnée

```
String ValeurSelectionnee = comboSelection.getValue()
```

## 6 – APPROCHE OBJET

Le taux de conversion peut être englobé dans un objet, l'ensemble des données dans un tableau d'objets.

#### Proposition de classe modèle **public**

```
class ConversionDevis {

    private String prompt;
    private String source;
    private String cible;
    private double Taux;

    ....
}
```

