

PROO / SEW III - Assignment: *JavaFx* Currency Converter

Objective

Create a currency converter including a selfmade chart using *JavaFx* and *FXML*.

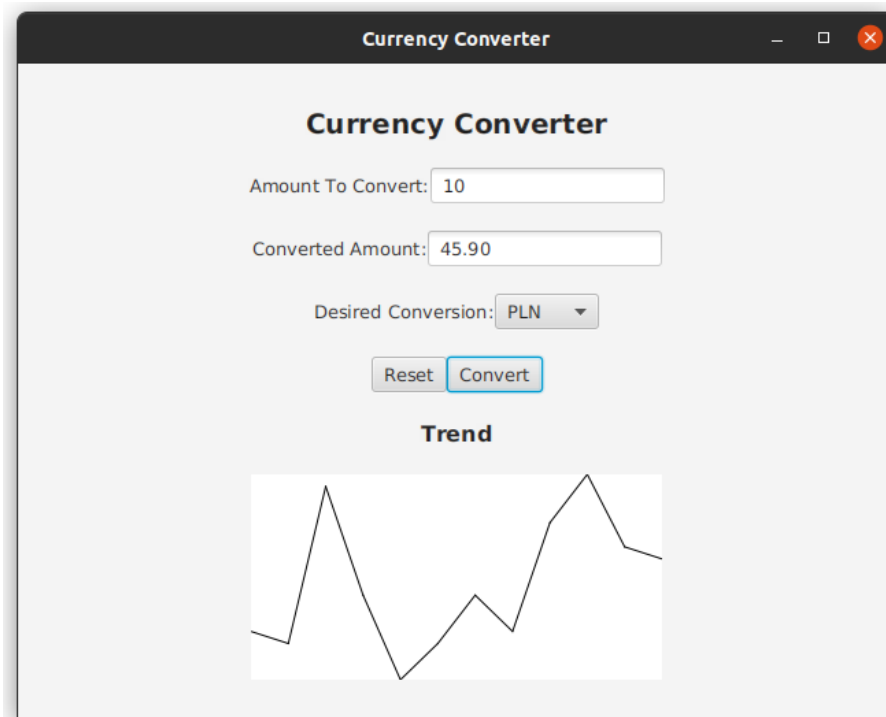
Things To Learn

- Create a user interface using *FXML* and a *controller*.
- Get to know more useful *JavaFX* *controls*.
- Display errors as *alerts*.
- Draw on a *canvas*.

Submission Guidelines

- Your implemented solution as **zipped** *IntelliJ*-project.

Task



1. **OBSOLETE:** Create a new project using the `javafx-maven-archetypes-archetype`. Select `javafx-archetype-fxml` as `archetypeArtifactId`.

2. Recreate the interface shown in the figure above using *FXML*.
 - Apart from multiple `HBoxes` you'll need `Buttons`, `Labels`, `TextFields`, a `ComboBox` and a `Canvas`.
 - Make sure the `TextField` for the output is not *editable* (Yes, the last word is a hint).
 - Make sure to set an `fx:id` for every control you'll need for the functionality of your program.
 - Use the `onAction` attribute on relevant controls to define methods for handling events. Don't forget the `#` at the start.
3. Create fields for all relevant controls and methods for all handlers inside your *controller* class.
 - Make sure to use the `@FXML` annotations so you can set the fields and methods to `private`!
4. Create the `initialize`-method in the controller to populate the `ComboBox` with the currencies and load the data for the charts.
 - Your program should be able to convert *Euros* to *Japanese Yen*, *Polish Zloty* and *US Dollars*. The exchange rates of all three currencies over the year 2021 are provided as `.txt`-files.
 - The last line of each file constitutes the most recent exchange rate for the respective currency and should be used for conversions.
 - Nevertheless you should store all values for each currency, since you'll be needing them for the charts. Make sure to choose a suitable *collection*!
 - **Instructions on how to load files from the *resources*-folder are provided in the *Hints* section!**
5. You should now be able to implement the functionality when the user presses one of the buttons.
 - If the user clicks the *Convert*-button with an invalid amount or no selected target currency, an appropriate error message should be displayed as `Alert` (with the `AlertType.ERROR`).
 - The *Reset*-button should clear the text fields and the canvas, as well as reset the dropdown menu.
6. In the last step, use the canvas at the bottom to display the trend of the exchange rate of the selected currency.
 - Changing the selected currency should automatically clear and redraw the chart (and clear the output text field).

Hints

The exchange rate data should go into the *resources* folder. You can then use the following exemplary code to load all lines from a file:

```
Files.readAllLines(Paths.get(this.getClass().getResource(fileName).toURI()))
```

Your project structure should look something like this:

- ▼ src
 - ▼ main
 - ▼ java
 - ▼ at.htlleonding.currency
 - App
 - ConverterController
 - module-info.java
 - ▼ resources
 - ▼ at.htlleonding.currency
 - converter.fxml
 - jpy.txt
 - pln.txt
 - usd.txt
