Retail Price Calculator

The course project of (2025W) Object-Oriented Programming-2477-WA

Professor: Dr. Jinan Fiaidhi

Student: Aydin Ghojogh

Student ID: 1166949

Question of the project:

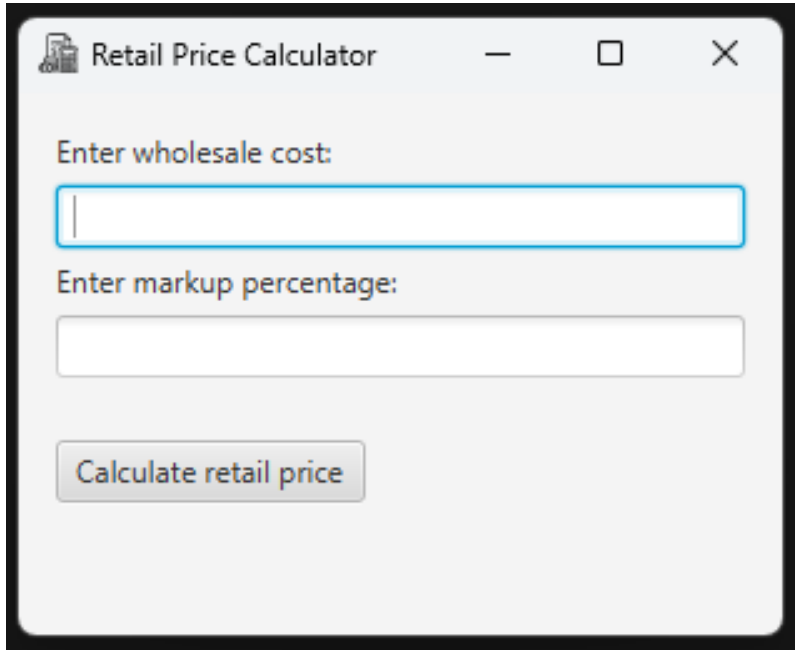
Create a GUI application where the user enters the wholesale cost of an item and its markup percentage into text fields. (For example, if an item's wholesale cost is $5 and its markup percentage is 100 percent, then its retail price is $10.) The application should have a button that displays the item's retail price when clicked. Create the GUI programmatically. Do not use FXML.

Outline

- The final GUI appearance
- Creating the class
- Overriding the method
- Label and text field for wholesale
- Label and text field for markup
- Button for calculate
- Label for result of calculation (retail price)
- Layout of GUI
- Size and title of window
- Icon for the window
- Event handling for the button
- Main for running the code
- A few example runs of the program

The final GUI appearance

- The final GUI appearance that I made is shown here.
- In the next slides, I explain the code which resulted in this GUI.



Creating the class

- Importing the class Application from javafx.application

```
import javafx.application.Application;
```

- Creating the class RetailPriceCalculator which inherits from the class Application

```
public class RetailPriceCalculator extends Application{
```

Overriding the method

- Overriding the method start() in the class Application

```
import javafx.stage.Stage;
```

```
@Override
public void start(Stage stage){
```

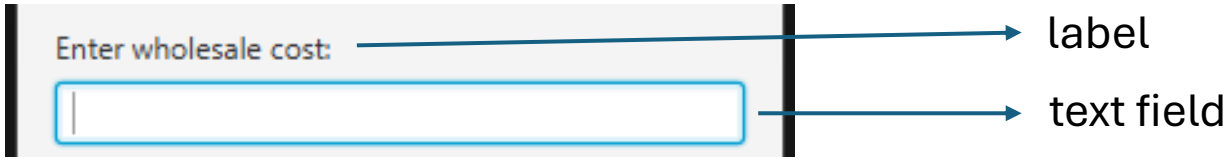
Label and text field for wholesale

- Creating the label and text field for wholesale:

```
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
```

```
// create label and text field for wholesale:
Label wholesaleLabel = new Label("Enter wholesale cost:");
TextField wholesaleTextField = new TextField();
```

- Result:

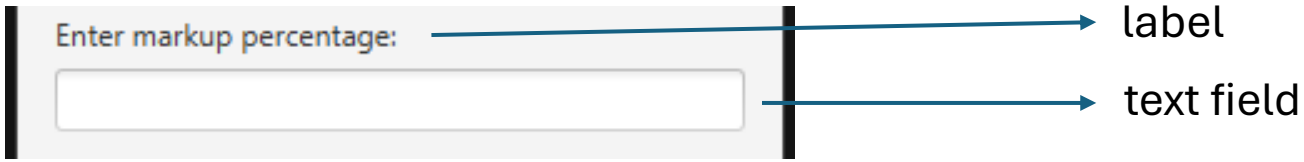


Label and text field for markup

- Creating the label and text field for markup:

```
// create label and text field for markup:
Label markupLabel = new Label("Enter markup percentage:");
TextField markupTextField = new TextField();
```

- Result:



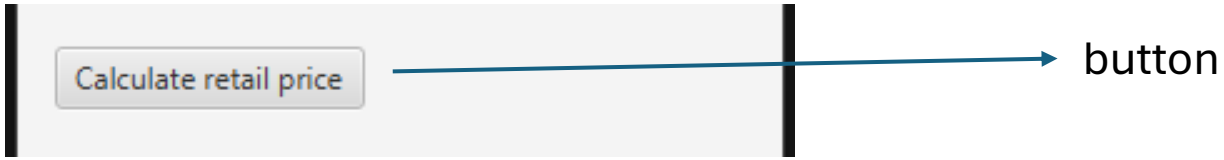
Button for calculate

- Creating the button for calculate:

```
import javafx.scene.control.Button;
```

```
// button:
Button calculateButton = new Button("Calculate retail price");
```

- Result:

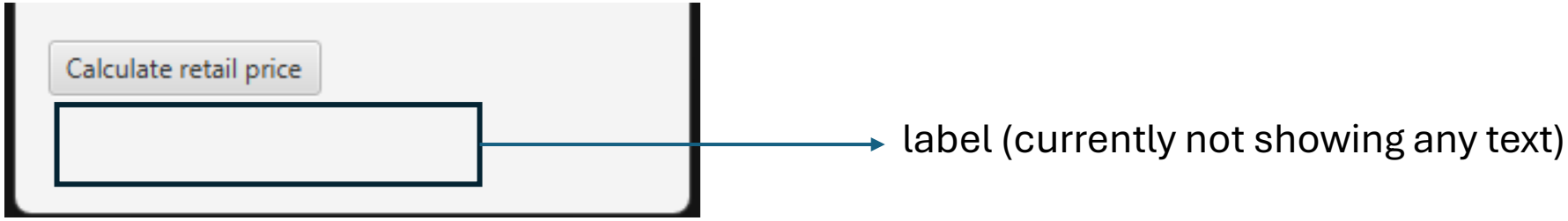


Label for result of calculation (retail price)

- Creating the label for retail price (result of calculation):

```
// label for result of calculation (retail price):
Label resultLabel = new Label();
```

- Result:



Layout of GUI

- Vertical box (VBox) for layout container:
  - I put the labels and textfields and button below each other.
  - I put 5 pixels space between them.

```
import javafx.scene.layout.VBox;
```

```
// layout of the wholesale:
VBox box = new VBox(5, wholesaleLabel, wholesaleTextField, markupLabel, markupTextField, calculateButton, resultLabel);
```

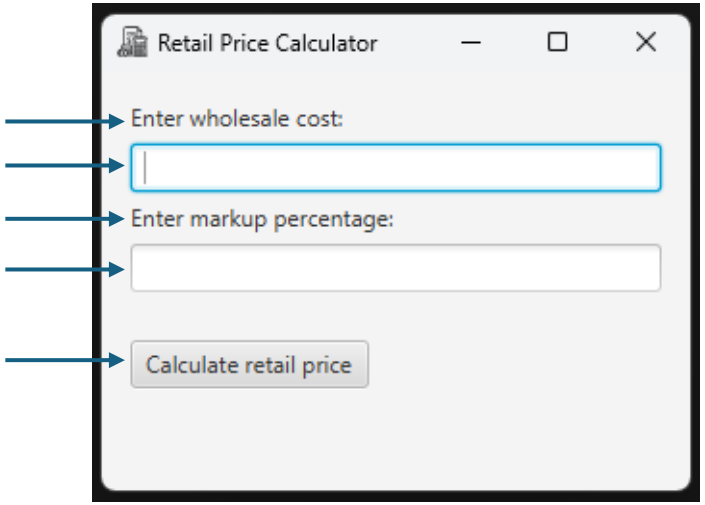
- Padding (gap) around the box: 15 pixels

```
// padding (gap) around the box:
box.setStyle("-fx-padding: 15px;");
```

- Add extra space (20 pixels) above the button:

```
import javafx.geometry.Insets;
```

```
// add extra space above the button:
VBox.setMargin(calculateButton, new Insets(20, 0, 0, 0));
```



Size and title of window

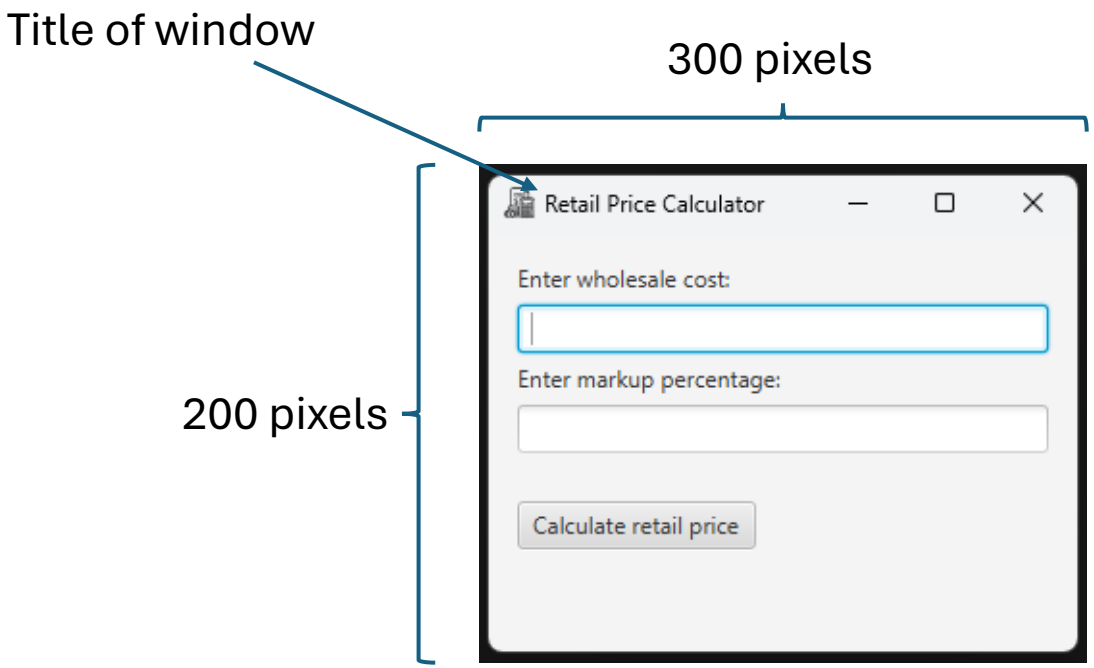
- Make an object of class Scene with size 300 pixels (horizontal) and 200 pixels (vertical)

```
import javafx.scene.Scene;
```

```
// scene (and size of window):
Scene scene = new Scene(box, 300, 200);
```

- Title of window:

```
// title of window:
stage.setTitle("Retail Price Calculator");
```



Icon for the window

- Icon of the window

```
import javafx.scene.image.Image;
```

```
// icon of window:
stage.getIcons().add(new Image(getClass().getResourceAsStream(name:"/icon.png")));
```

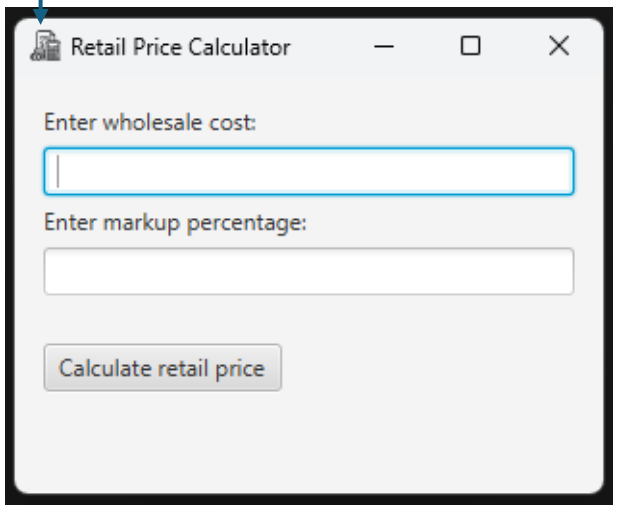
- I downloaded the icon from Internet: https://www.flaticon.com/free-icons/finance

- Set the scene for the stage and show the stage:

```
// set scene for the stage:
stage.setScene(scene);

// show the stage:
stage.show();
```

Icon of window



Event handling for the button

- Event handling for the button:
  - Read the entered numbers in the wholesale textfield and markup text field
  - Calculate the retail price
  - Set the text of retail price label
- Exception handling:
  - If the user does not enter valid numbers in wholesale textfield and markup text field
  - I catch it and set the text of retail price label to: “Please enter valid numbers.”

```
// event handling for the button:
calculateButton.setOnAction(e -> {
    try{
        // read the entered numbers:
        double wholesaleCost = Double.parseDouble(wholesaleTextField.getText());
        double markupPercentage = Double.parseDouble(markupTextField.getText());

        // calculate the retail price:
        double retailPrice = wholesaleCost * (1 + (markupPercentage / 100));
        resultLabel.setText(String.format(format:"Retail Price: $%.2f", retailPrice));
    }
    catch(NumberFormatException ex){
        resultLabel.setText("Please enter valid numbers.");
    }
});
```

Main for running the code

- Main for running the code and launch the GUI
  - The method launch() is in the class Application

```
public static void main(String[] args) {
    launch(args);
}
```

A few example runs of the program