

Challenge: Grade Calculator

Description: Create a Java 8 SE application in NetBeans using JavaFX that contains four buttons, three labels, and three text fields. JavaFX code is to be used to create the buttons, labels, text fields, and the user interface (UI). The UI objects are to be laid out using a GridPane, based on the design specified in the requirements.

Purpose: This challenge introduces the use of JavaFX to create Java user interfaces.

Requirements:

Project Name: <Pawprint>GradeCalculator

For the Project Name, follow the same naming scheme used in the previous challenges. The Project Name is to be comprised of your pawprint with the first letter capitalized followed by GradeCalculator. For example, if the pawprint is **abcxyz9** the project is to be named **Abcxyz9GradeCalculator**.

In NetBeans create a **JavaFX Application** project for this challenge. The project that is created will contain sample code that displays a button that when clicked prints "HelloWorld!" to standard output. You will need to replace the sample code in the start() method with the code you write to meet the requirements of this challenge.

Write the code necessary to do the following inside the start() method of the application's Main Class which will be a subclass of Application.

The user interface (UI) is to contain four buttons, three labels, and three text fields. The three labels are set to text "Category 1 (30%):", "Category 2 (70%)", and "My Final Score". The three text fields are "Score1", "Score2", and "Score3". The text on the first button is to be "Maximize", the second button "Calculate", the third button "Clear", and the fourth button "Alert".

- When the Maximize button is clicked, the text in Score1 and Score2 will be set to 100
- When the Calculate button is clicked, the Score3 text field will be set to "My final score should be $\text{SCORE1} * 0.3 + \text{SCORE2} * 0.7 = \text{<Final Value>}$ " in which the SCORE1 and SCORE2 are actual real values in the text fields of Score1 and Score2, and <Final Value> will be of type double of the actual final value after the calculation is performed. Make sure you format the final value to only display two decimal places. In order to read in

Challenge: Grade Calculator

the values from the text fields, you will need to research about conversions from String to double.

- When the Clear button is clicked then all three text fields are cleared.
- When the Alert button is clicked the data from the text field Score3 should be displayed in a JavaFX alert box.

A GridPane is to be used to position the UI elements. A value of 15 is to be used for the horizontal and vertical gaps for the GridPane. The alignment is to be set centered (Pos.CENTER) on the GridPane for the positioning of UI elements.

The labels and text fields should be put on the GridPane in the following order: label1, textfield1, label2, textfield2, label3, textfield3. There are also screenshots below. The following table shows the row and column index:

Element	Row	Column
Label1	0	0
Textfield1	1	0
Label2	2	0
Textfield2	3	0
Label3	4	0
Textfield3	5	0

All text fields should have a preferred width of 400.

The Maximize, Calculate, Clear, and Alert Buttons are to be added to a VBox and the VBox should be added to the GridPane at row 6 and column 0.

The VBox should have spacing set to 15, padding set to 0, 0, 15, 0 for each of the four corresponding corners, respectively.

To make all the buttons the same width, set the max width of the button to "Double.MAX_VALUE" and let the VBox handle everything else.

Challenge: Grade Calculator

- **Note:** DO NOT specify the width of each button individually! You should not go through the trouble of determining the width of each button and then setting the preferred size of each. An easier option is to let the layout panes do the work.

The Scene for the JavaFX application is to be 500 pixels wide by 400 pixels high. The title for the Stage is to be “Grade Calculator”.

Things to submit on Canvas:

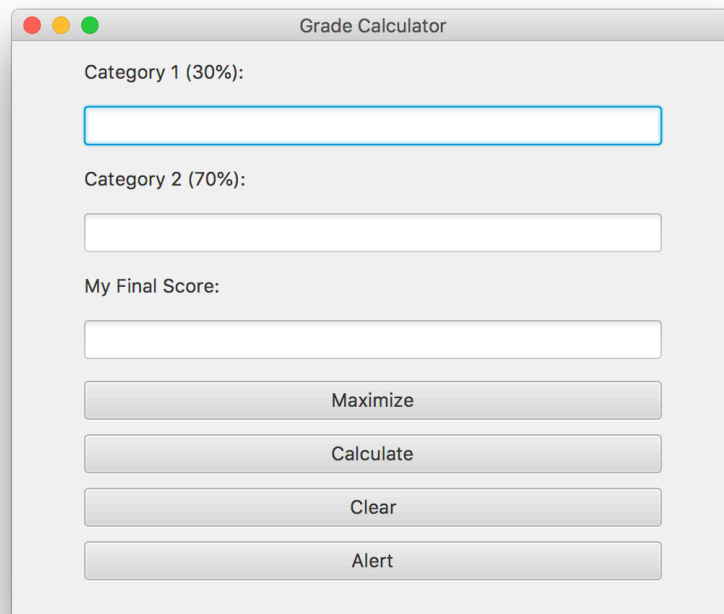
- The zip file created after you export your project from NetBeans.
- You may also submit screenshots of your application running for proof (optional). Put all your screenshots in a folder, name the folder “<Pawprint>Screenshots” where you replace <pawprint> with your pawprint, and zip them, even if you only take one. Then submit the zip of screenshots on canvas.

Note: You are only allowed to submit one thing at a time on canvas. You cannot submit a zip file and your screenshots. Therefore, first submit the screenshots, then click “re-submit”, and submit your zip file. On your end, it will look like you only submitted the zip file, however, on our end, we will see both.

Challenge: Grade Calculator

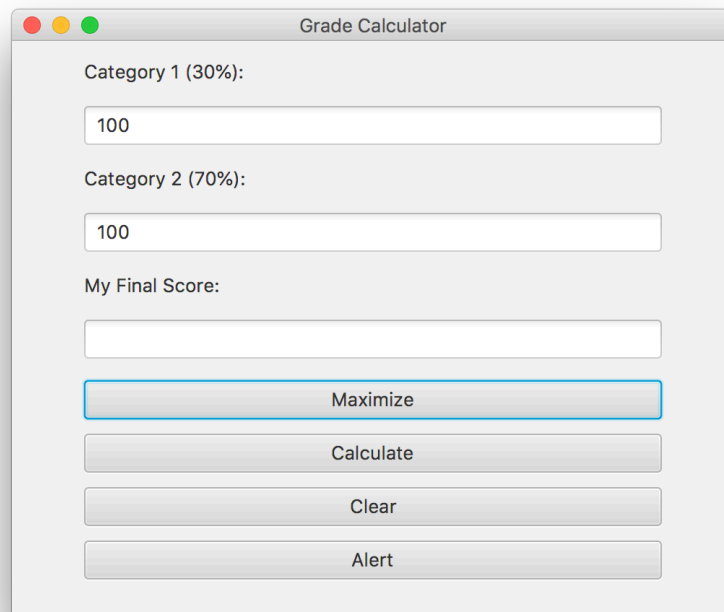
Example Screenshots:

The following is a screen capture of how the user interface is to look:



A screenshot of a macOS-style window titled "Grade Calculator". The window has a light gray background and rounded corners. It contains the following elements from top to bottom: a label "Category 1 (30%):" followed by an empty text input field; a label "Category 2 (70%):" followed by an empty text input field; a label "My Final Score:" followed by an empty text input field; and four buttons stacked vertically: "Maximize", "Calculate", "Clear", and "Alert". All buttons have a light gray gradient and rounded corners.

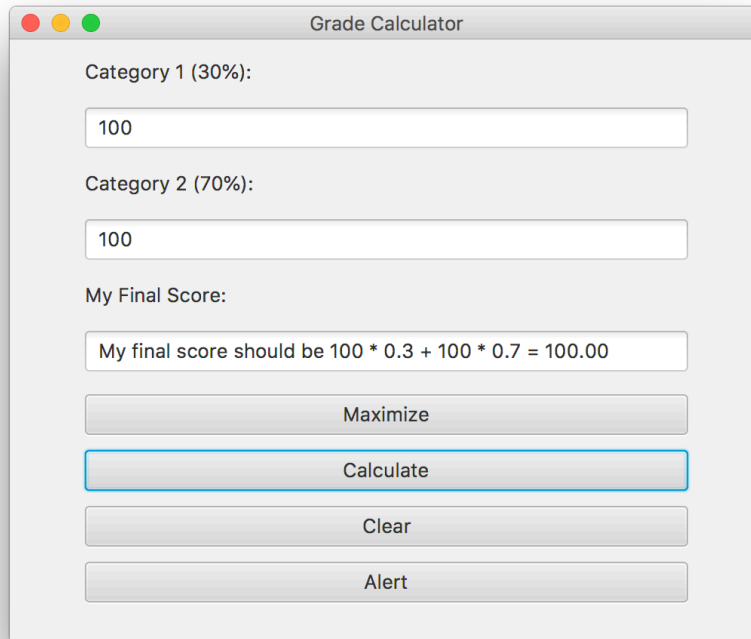
When the "Maximize" button is clicked:



A screenshot of the "Grade Calculator" window after the "Maximize" button has been clicked. The window's appearance is identical to the first screenshot, but the "Maximize" button is now highlighted with a blue border, indicating it is the active element. The text input fields remain empty.

Challenge: Grade Calculator

When the “Calculate” button is clicked:



Grade Calculator

Category 1 (30%):

Category 2 (70%):

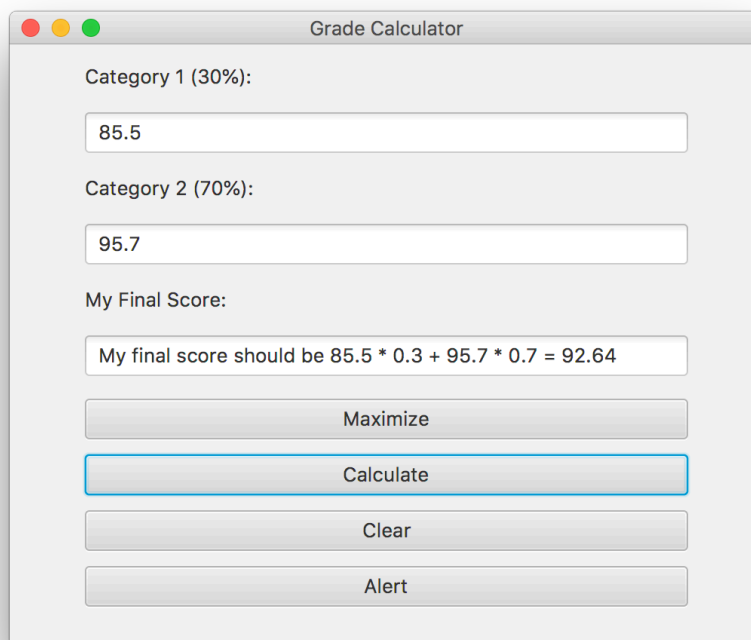
My Final Score:

Maximize

Calculate

Clear

Alert



Grade Calculator

Category 1 (30%):

Category 2 (70%):

My Final Score:

Maximize

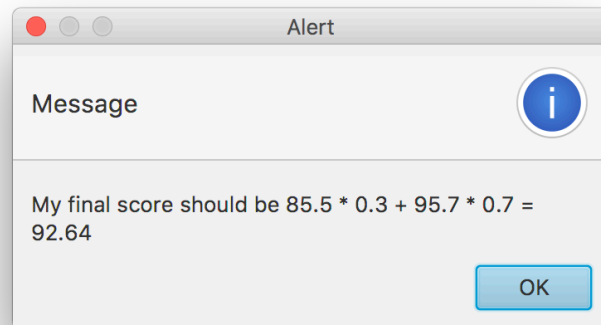
Calculate

Clear

Alert

Challenge: Grade Calculator

When the “Alert” button is clicked:



When the “Clear” button is clicked:

