# Project Calculator

Step-by-Step Calculator Guideline

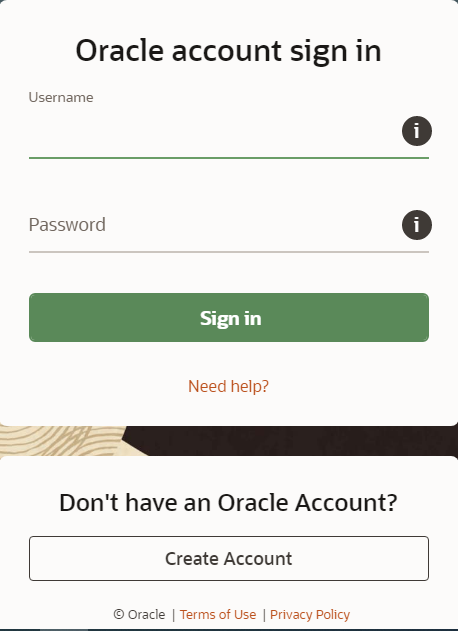
To make our calculator we are going to use JavaFx and its Scene Builder as it is part of Java SE Development Kit 8 for this we will have to download and install it.

1. **Download and install Java SE Development Kit 8**

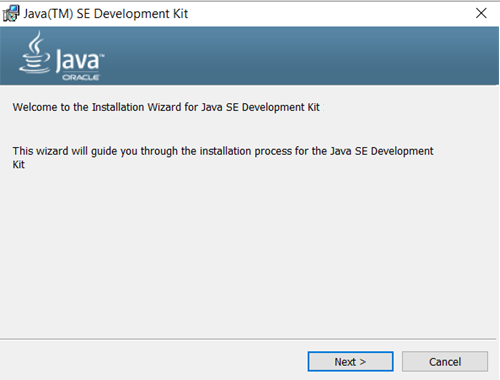
You can download Java SE Development Kit 8 from here:

<https://www.oracle.com/java/technologies/javase-jdk8-downloads.html>

Installing an earlier version of Java, such as Java 8, requires Oracle registration. After clicking on the download link of JDK 8, you will be redirected to a page to log in to your Oracle account. If you don't have an account, click the "Create Account" button and sign up.



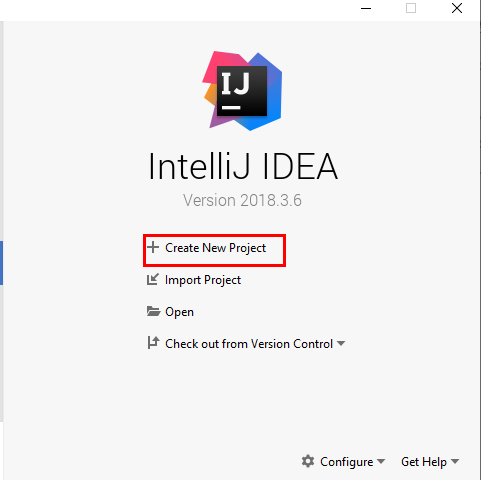
After we download the installation file and open it, the JDK installation panel is displayed. Continue pressing the "Next" button



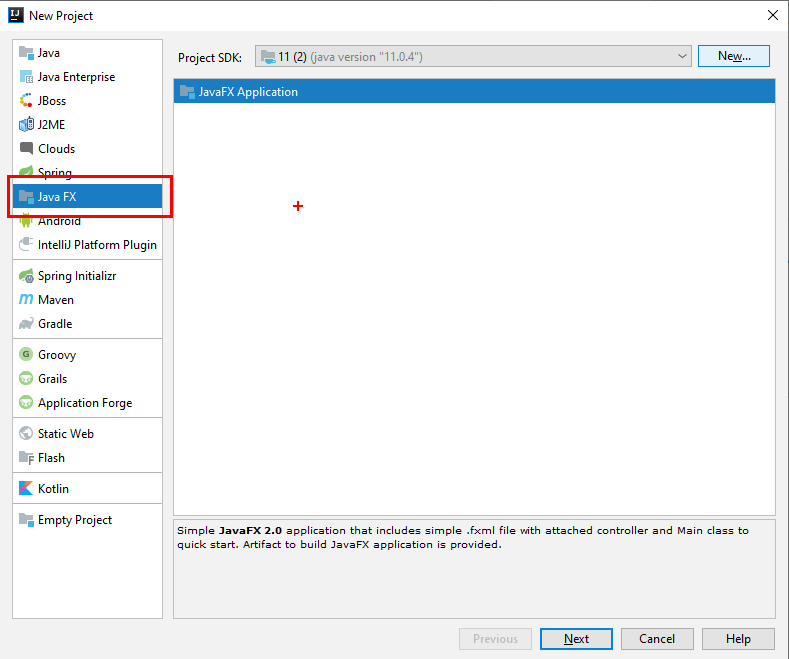
In the following steps, simply follow the instructions, it is preferable to select the following installation directory: C: \ Program Files \ Java

1. **Create JavaFX Application**

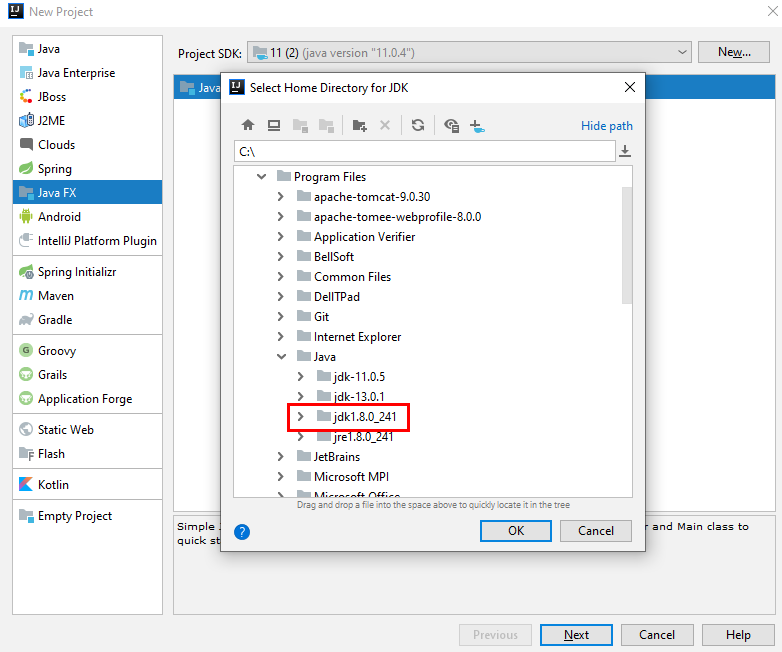
From the IntelliJ IDEA initial window, select "Create New Project".



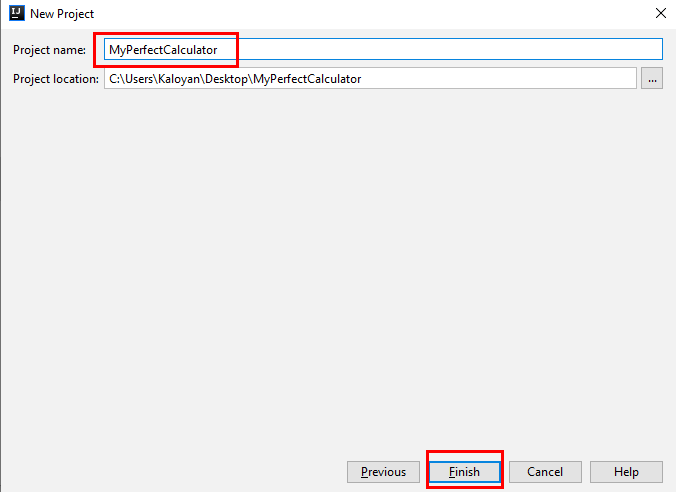
In the next window you have to select "JavaFX" and also set up the Project SDK after pressing the "New" button and selecting the first option "JDK".



Then select the folder where is the already installed JDK 8 package, if you didn't change the default folder during the installation, it will be by located in: C: \ Program Files \ Java

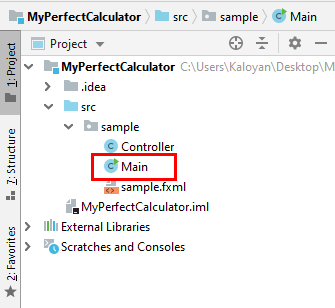
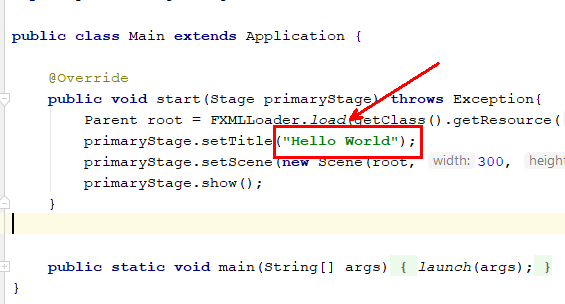


Choose adequate name and click Finish.



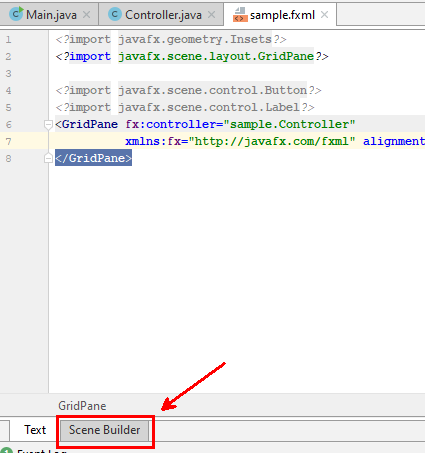
Now our project is already created, you can run it right away, although it is empty.

The next think to do is to change application name by default is "Hello World".

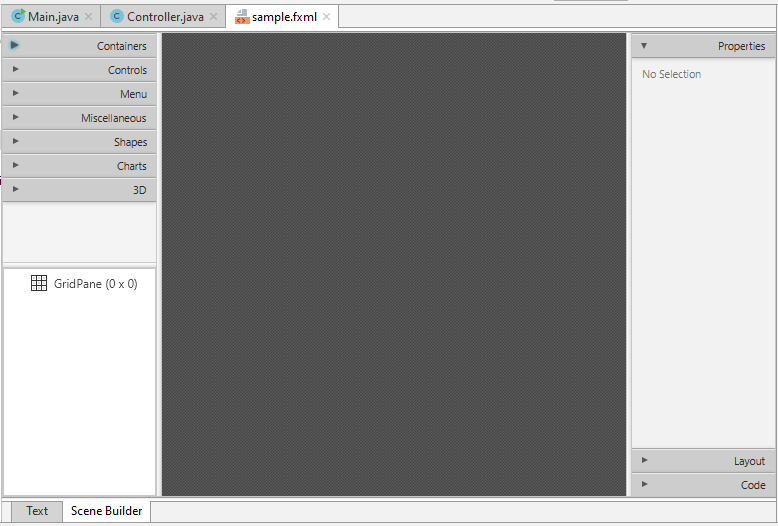
 

1. **Open sample.fxml file**

Select sample.fxml -> double click the file. You are going to see following window click on the Scene Builder

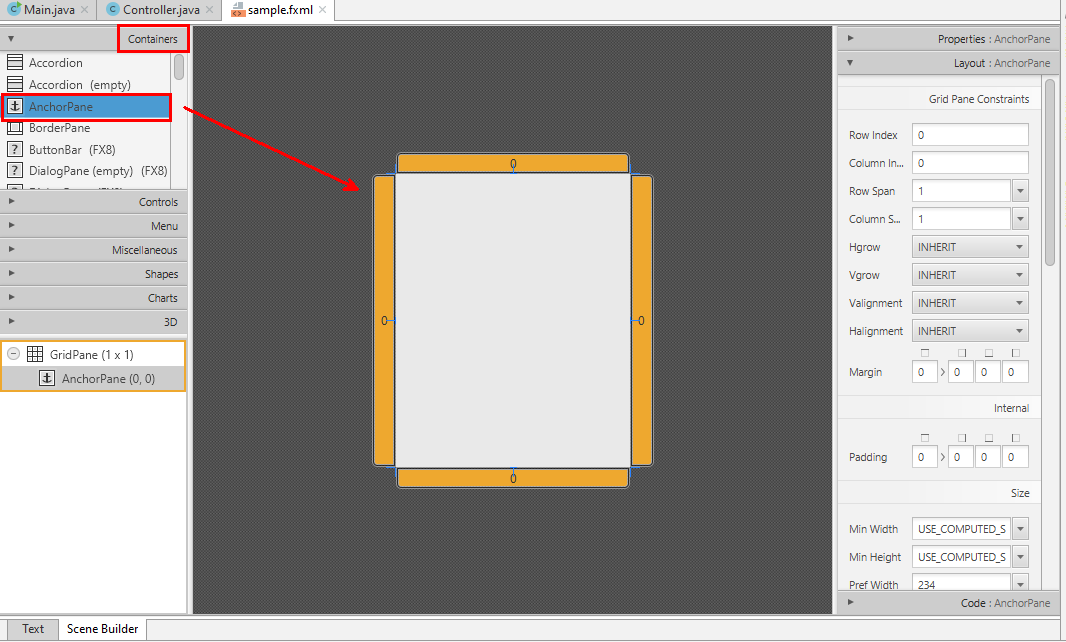


This is the blank main window of our desktop application.

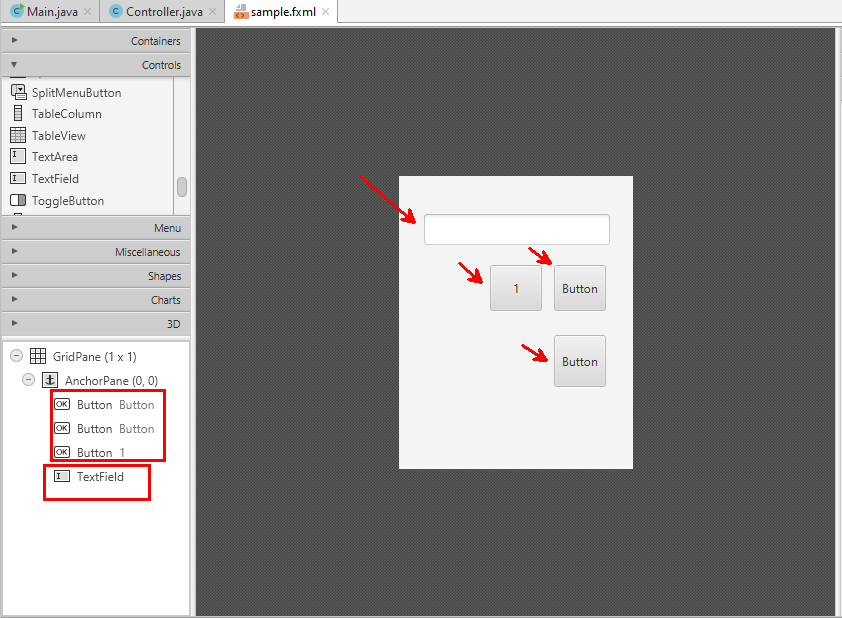


1. **Add elements**

First we need to add a container in which to place the buttons.

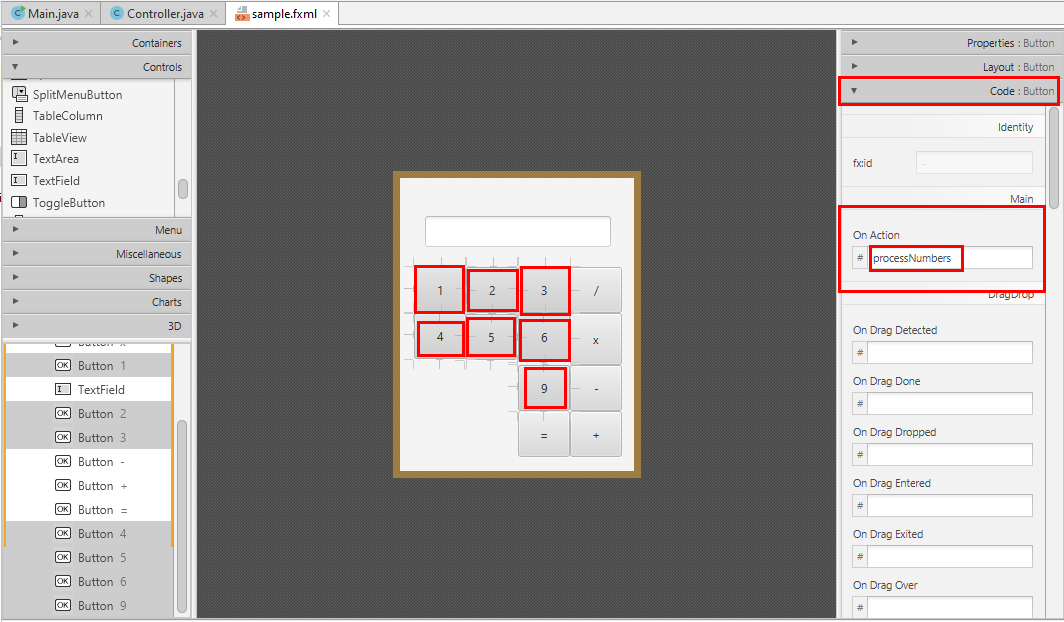


Now we can add buttons for operands (numbers from 0-9) and operators (+, -, \*, / etc.) and a textField for the display you can find it on the left toolbar in controls menu. You can change the alignment of the text, the font size and etc. You can easily change the label of a button by double-clicking it.

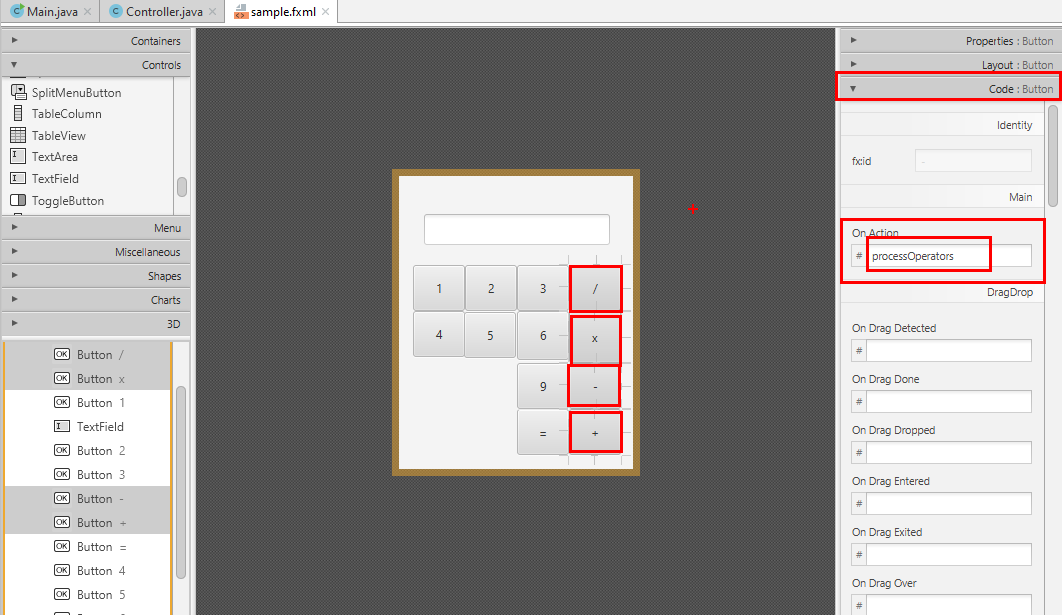


Next step is to set the up the buttons, on the right toolbar in code menu fill the **On Action field** for each operands (numbers) you have to write: **processNumbers.**

**Hint:** You can make multiple selection by holding **ctrl** button.

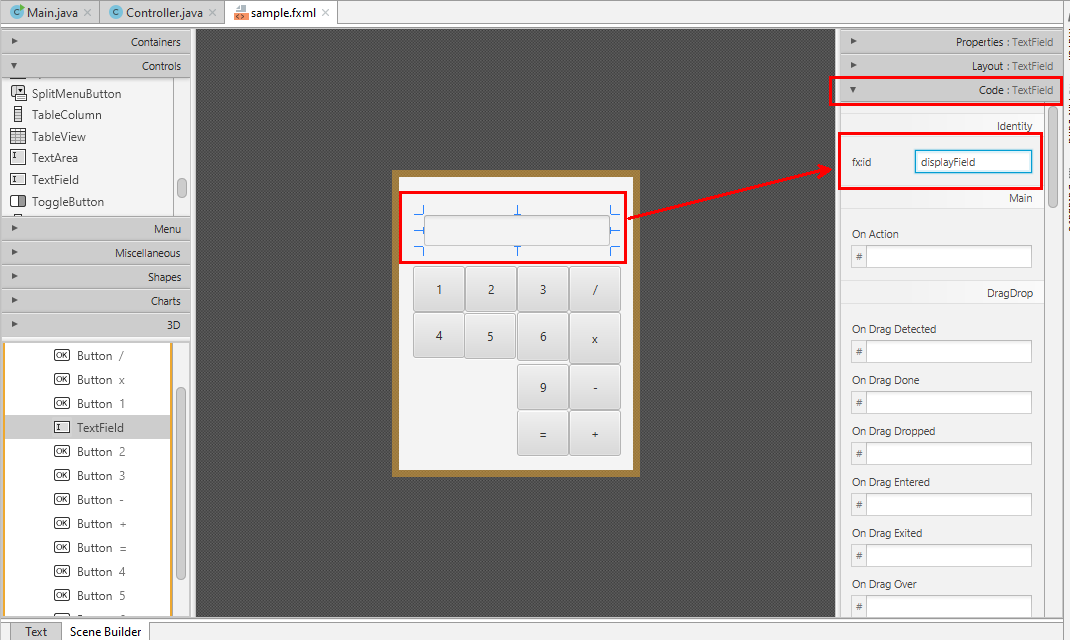


for operators(+, -, \*,/ ): **processOperators**



For equals operator (=): On Action -> **equal**

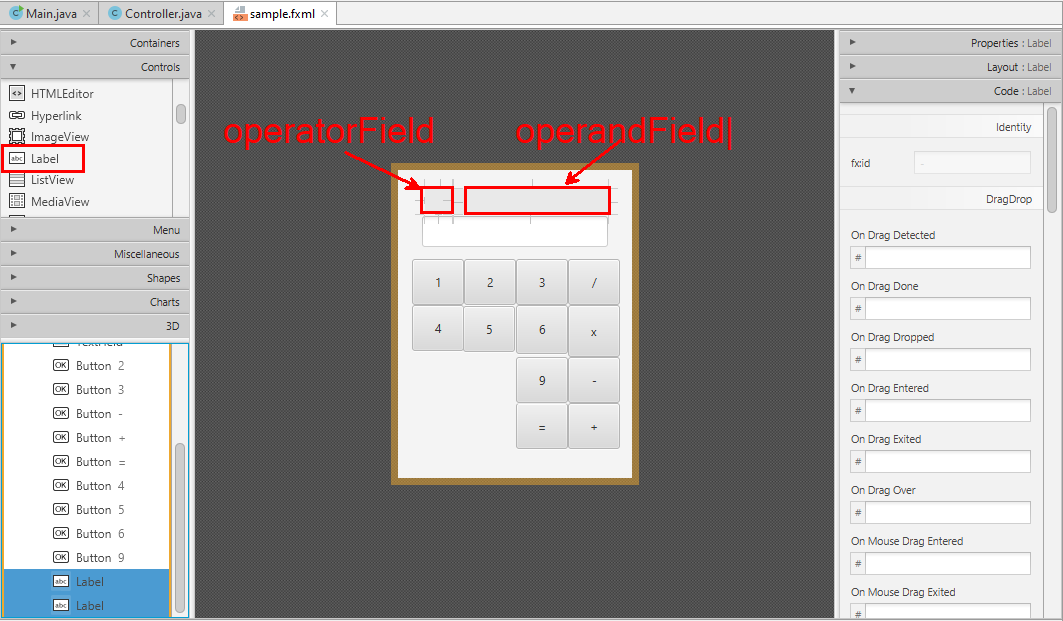
For the display field: fx:id -> **displayField**



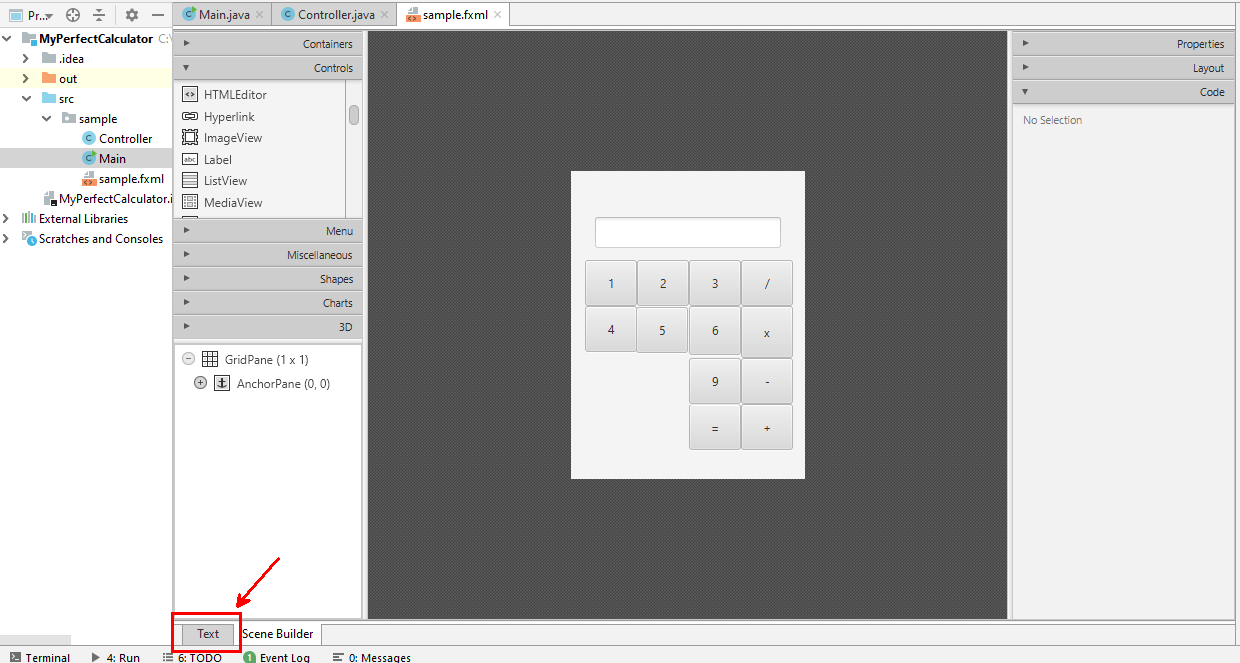
We need two more fields one is to store the number which the user entered in the displayField and the other is for the operand.

Operand field -> fx:id -> **operandField**

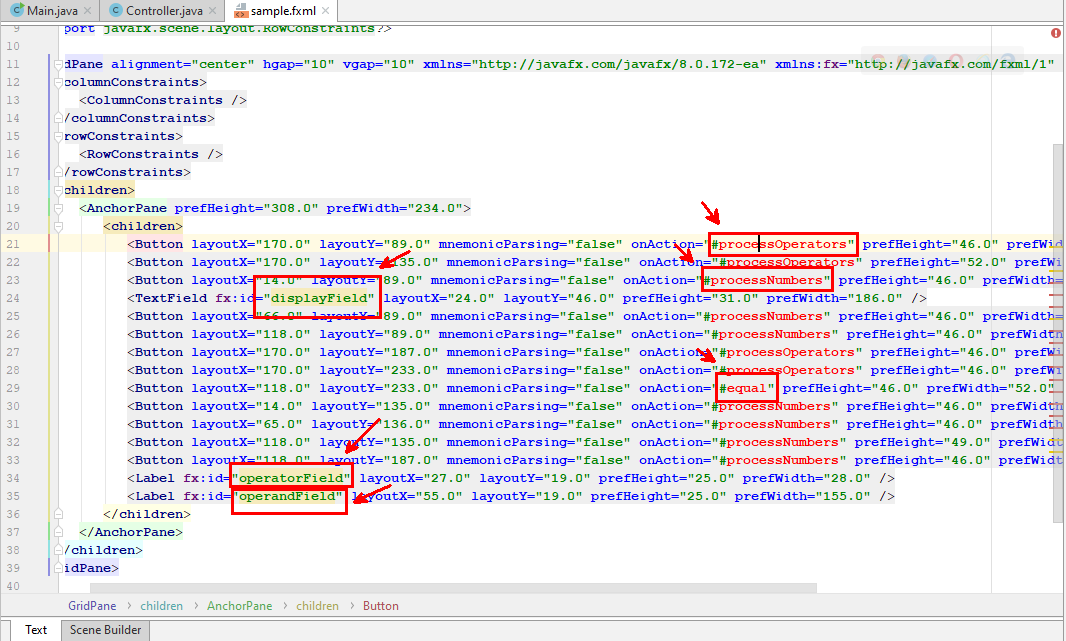
Operator field -> fx:id -> **operatorField**



Before to proceed to implementation we should do one more thing by clicking on Text

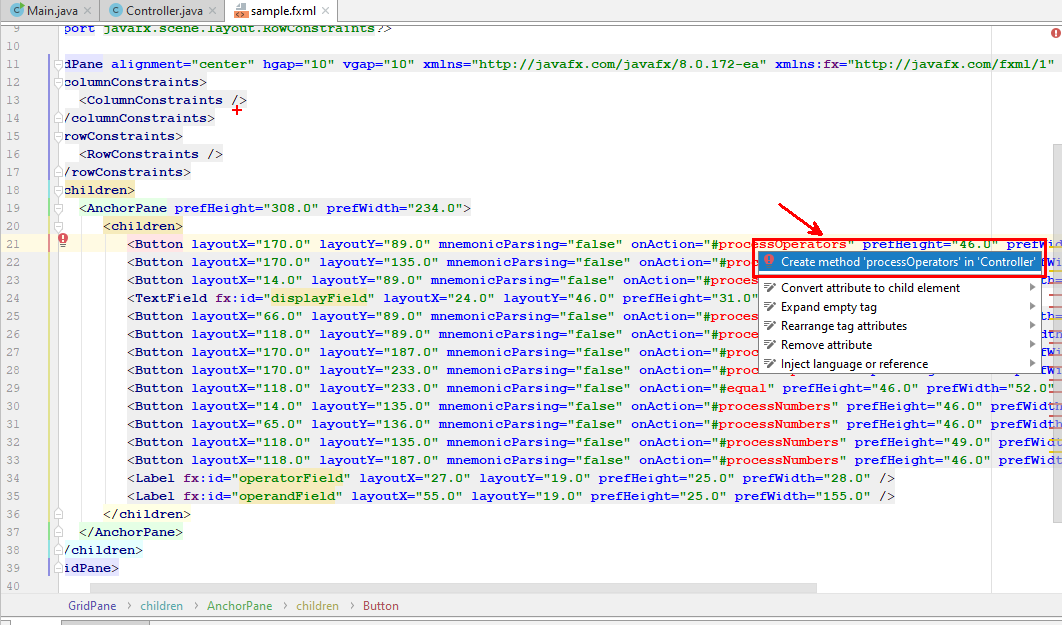


It will appear following window:

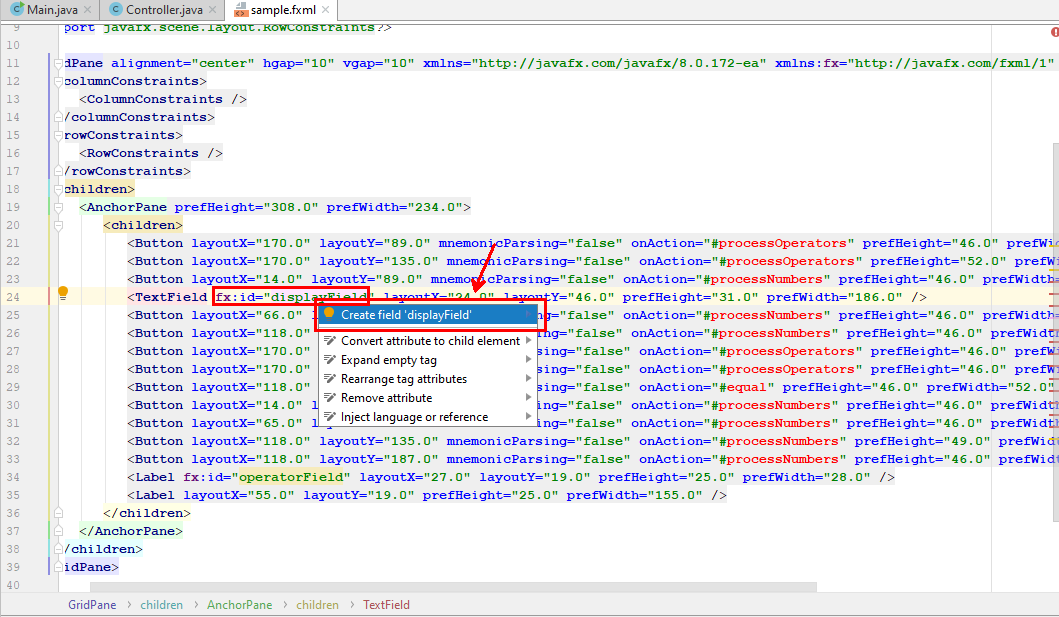


On each of the marked fields put the mouse cursor and hit the key combination **alt + enter.**

It will appear a menu and you have to choose the first option for the onAction fields is:



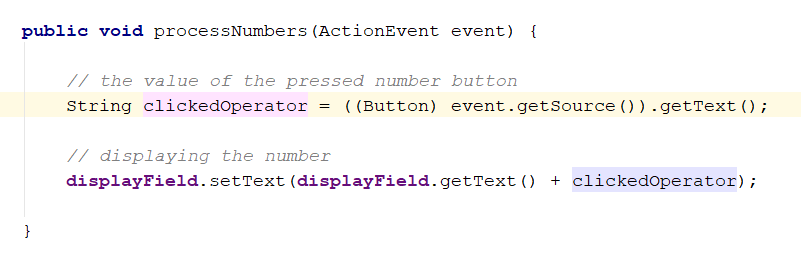
And for the fx:id ->



In **Controller** class were created following methods let's implement them.

1. **Implement Number Buttons**

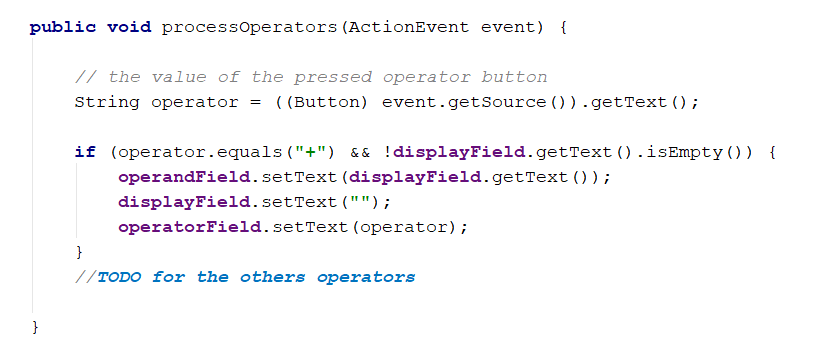
When the user presses any **number** it creates an event, because we have already attached for each number button "On Action" -> processNumbers.



1. **Implement Operator Buttons**

First check if there is any input from the user in the **displayField**.

Then transfer the displayField value to the operandField Value and add the "+" sign in the operatorField.



1. **Implement Equals Operator**

When we are done implementing all the buttons. We can add the equal operator.

First, we should check if there are operands and operator.

Then we should use the operator to understand which operation we should use.  
After we calculate the result, we should display the result in the **DisplayBox** and reset our **OperatorDisplay** and **OperandDisplay**.

Good Luck!

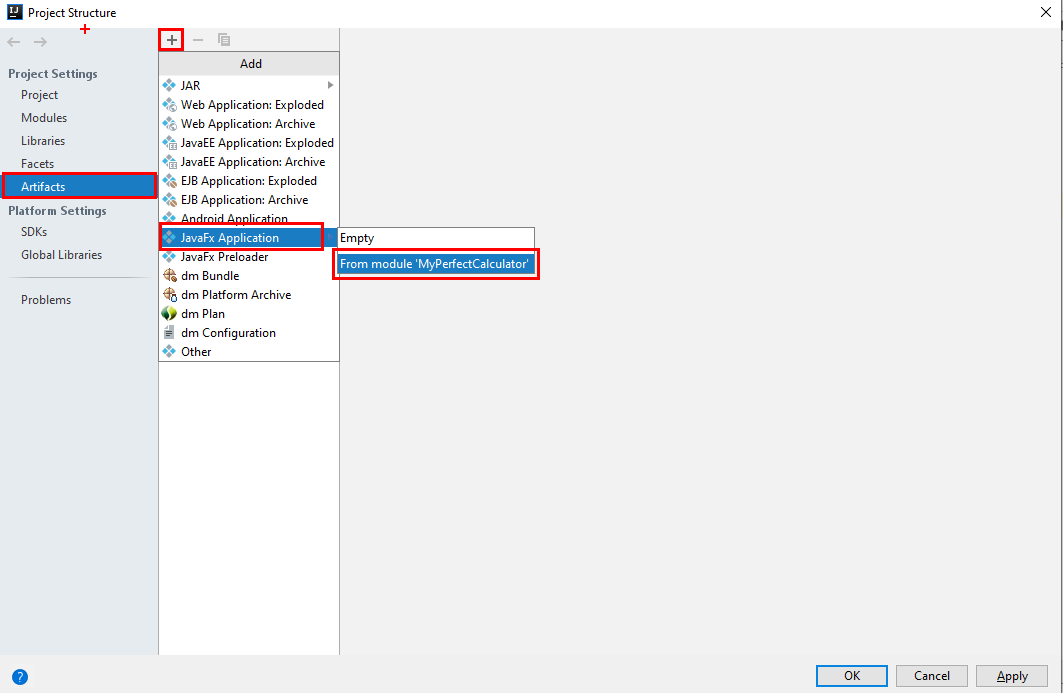
1. **Implement other functionalities**

Try implementing dot (.) so the user can enter real numbers, not only integers.

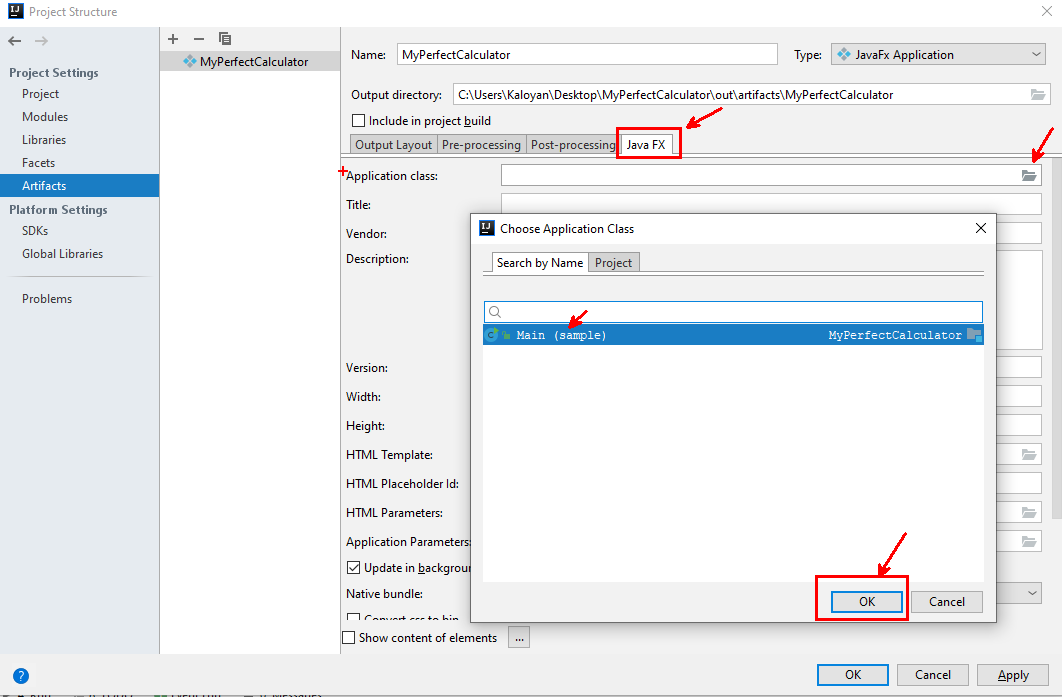
Then continue developing your program logic and functionalities.

1. **Publish**

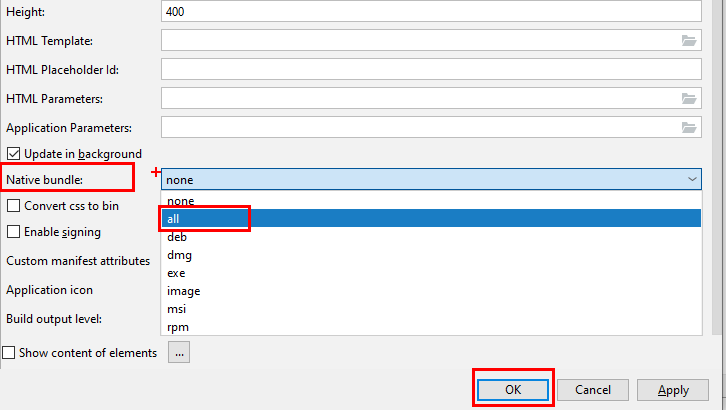
Open Project Structure: File -> Project Structure



After that ->



And the last thing to do is ->



In the end click on the Build -> Build Artifacts -> Build Artifact ->Action -> Build

In "\MyPerfectCalculator\out\artifacts\MyPerfectCalculator\bundles\MyPerfectCalculator" folder you can find the MyPerfectCalculator.exe file.

Congratulations you've managed to create your own first desktop application!