

# ASSIGNMENT II

**Team Members:**

**Pengfei Guo**

**Vikas Gautam**

**Thomas Sherk**

# Problem 1: Converter

## Analysis:

- The converter should be able to evaluate the values for the given input for Temperature, Height and Mass.
- The values for Temperature can be given in degree **Fahrenheit** (°F) or degree **Celsius** (°C). So we should be able to evaluate if any of its value is given.
- For Height conversion, we should be able to convert input form **Feet and Inches** to **meters** or vice versa.
- Also for the Mass conversion we should be able to perform the operation of converting **Pounds** to **Kilograms** or vice versa.

## Issues:

- When you take the value for one unit say Height, the user might give the input just as numbers like 32 for Feet or the user can give it as 32 feet 12 inches. So, we should be able to convert the value for given input for both the instances into meters.
- This goes same for all the units that is to be converted.
- After the execution of conversion of one unit, we should be able to clear out the text Field area for another calculation.
- We should be able to toggle from one conversion unit to another when the particular button for the unit is selected like when the user selects to convert the values for Mass then the prompts for the mass conversion should appear and ask the user to input the values for Pounds or Kilograms.
- The prompts for other units should be invisible when a particular unit is selected.

## Working:

We used the formula for the conversion of :

### 1. Temperature

- a. Fahrenheit to Celsius

$$F = (C * (9/5)) + 32$$

- b. Celsius to Fahrenheit

$$C = (F - 32) * (5/9)$$

### 2. Height

- a. As we know that **1 Foot = 0.3048 Meter** we can use this equation for our conversion for foot to meter
- b. And **1 Meter = 3.28084 Foot**, so we can use this equation for our calculations

### 3. Mass

- a. We know that **1 Pound = 0.453592 Kilograms**
- b. And **1 Kilogram = 2.20462 pounds**

We have three main operations to perform in this problem and those are to convert and those are converting the units of temperature measurement i.e. Fahrenheit to Celsius or vice versa. And the same operations are performed for height and Mass i.e. Feet with inches into meters and Pounds with ounces into kilograms respectively or vice versa for both of them.

Firstly we defined and named all the buttons, labels and text fields that are displayed on the Unit Converter in the scene builder i.e. javaFX.

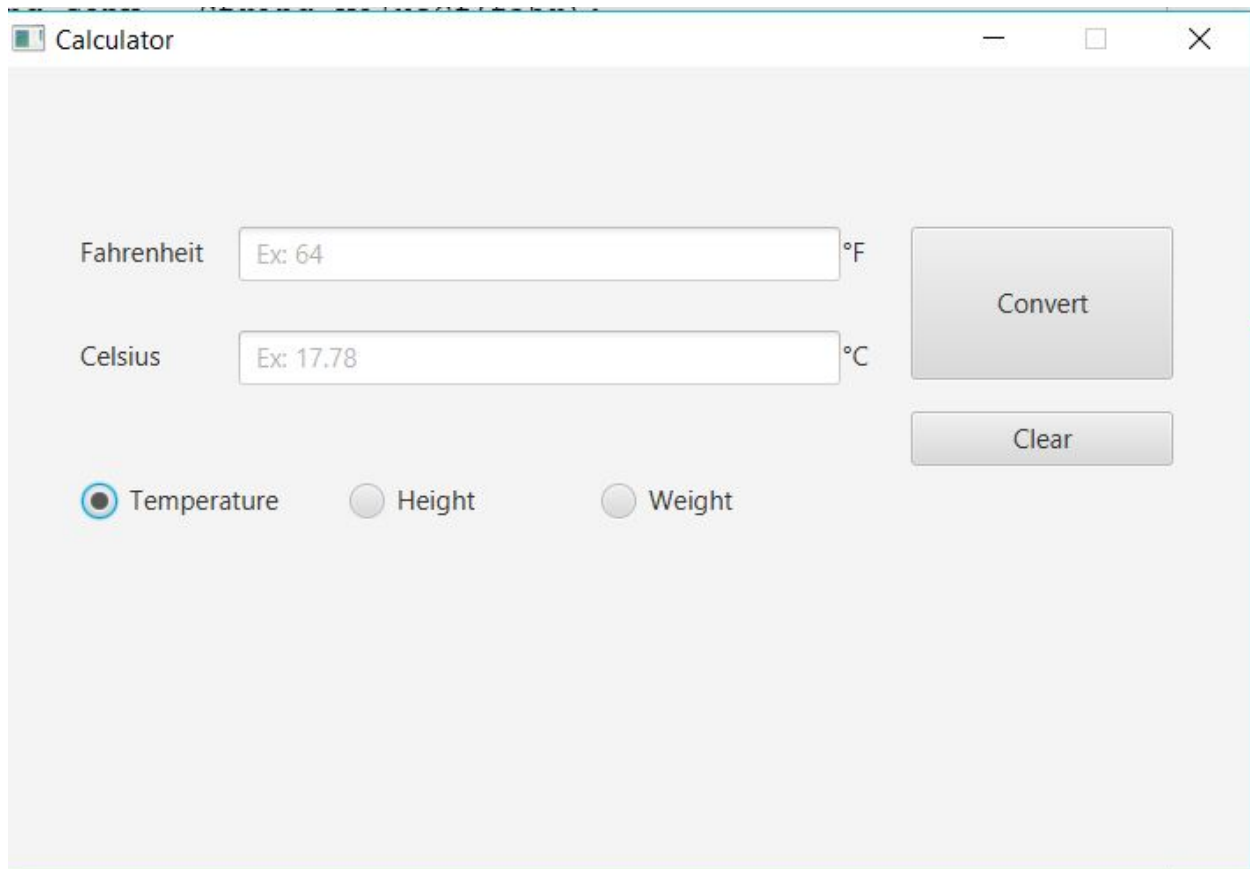
Then we defined the ***HandleButtonConvertAction()*** function which performs the operation for converting all the values that are given in the text field for conversion.

We also defined function *tempSelected()*, *heightSelected()* and *massSelected()* which is responsible to change the units for conversion when three different operations (Temperature, Height and Mass) are selected.

### Operation For Temperature Button:

When temperature button is selected. Then the other radio buttons for height and weight are disabled and their functions too because the *tempSelected()* function forces it to do so.

The function checks for the text fields if both of them are empty. If they are empty it asks the user to enter one of them or the user enters the value in both of the text fields then it throws an exception stating that select only one.

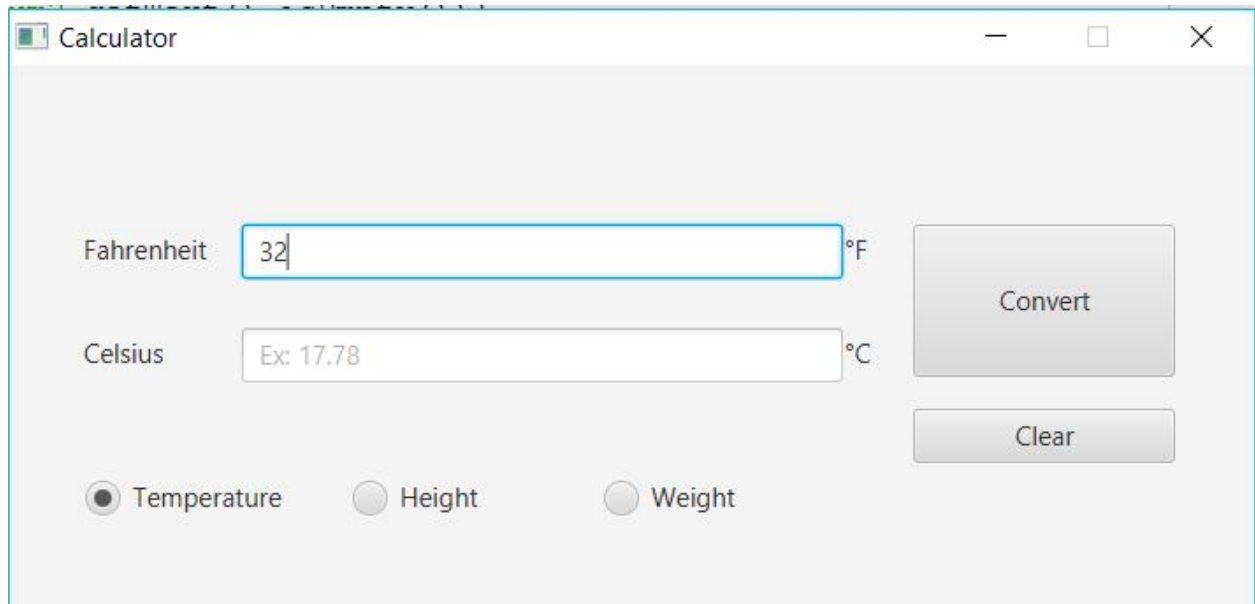


The screenshot shows a Java Swing window titled "Calculator" with a standard Mac OS X title bar (red, yellow, and green buttons). The window contains a temperature conversion interface. It features two text input fields: the top one is labeled "Fahrenheit" and contains the text "Ex: 64" followed by a "°F" unit indicator; the bottom one is labeled "Celsius" and contains the text "Ex: 17.78" followed by a "°C" unit indicator. To the right of these fields are two buttons: "Convert" and "Clear". At the bottom of the window, there are three radio buttons: "Temperature" (which is selected, indicated by a blue dot), "Height", and "Weight".

When the user gives the input in any one of the field then the converted value of the input is shown in the other text field.

**Example:**

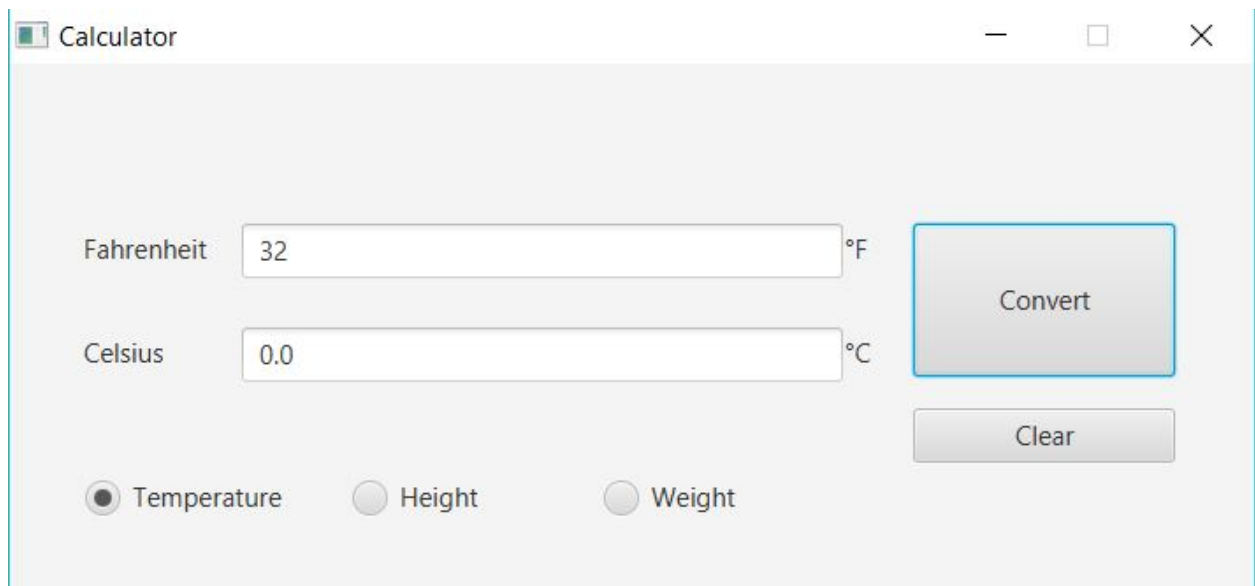
When  $F = 32^{\circ}\text{F}$



The image shows a window titled "Calculator". Inside, there are two text input fields. The first field is labeled "Fahrenheit" and contains the value "32". To its right is a "°F" symbol. The second field is labeled "Celsius" and contains the text "Ex: 17.78". To its right is a "°C" symbol. Below these fields are three radio buttons: "Temperature" (which is selected), "Height", and "Weight". To the right of the input fields are two buttons: "Convert" and "Clear".

*Fig 1 : This figure illustrates the value which is given into text field of fahrenheit*

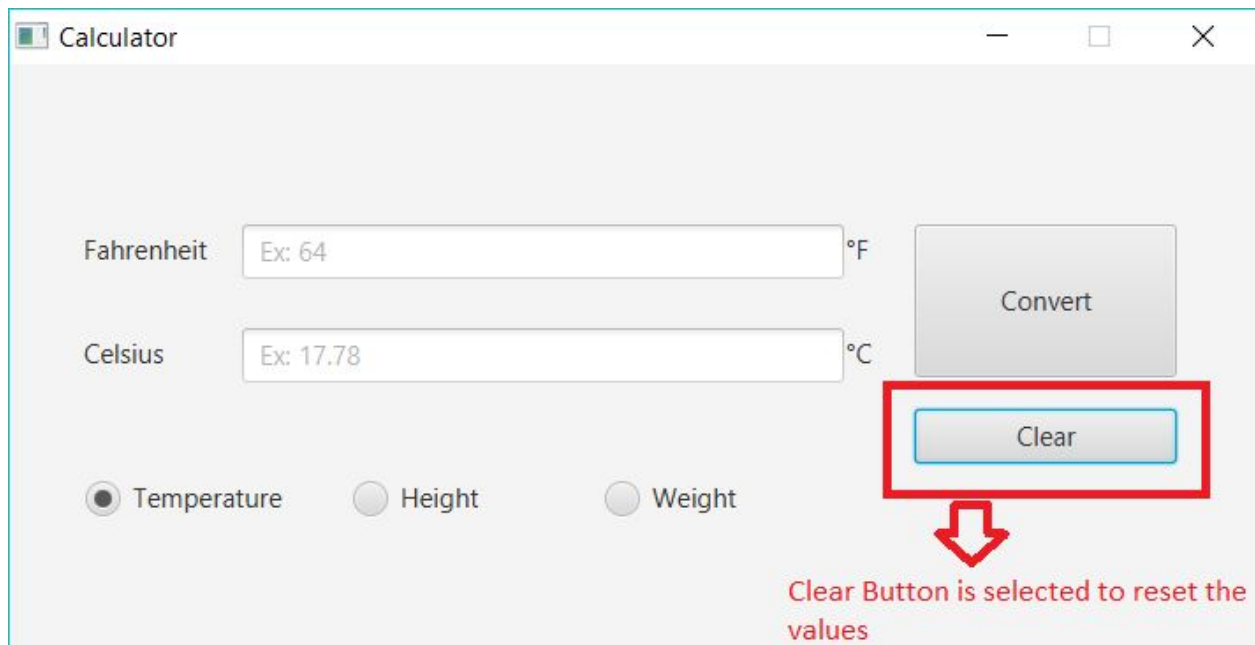
Output:



The image shows the same "Calculator" window as in Fig 1. The "Fahrenheit" field still contains "32". However, the "Celsius" field now contains the value "0.0". The "Convert" button is highlighted with a blue border, indicating it was just clicked. All other elements, including the radio buttons and the "Clear" button, remain the same.

*Fig 2: The output is printed into the other text field*

Then when you press the clear the text fields are cleared and the values can be re-entered.



### **Operation For Height Button:**

When Height radio button is selected then temperature and mass functions are deactivated and the units for conversion for height is popped onto the converter.

The ***heightSelected()*** helps for the function to work accordingly. If the user wants to give the input in Feet and inches then he can directly put the values into the respective text fields. The user can give the values in text field like “5 ft 9 in” or “5 feet 9 inches” or “ 5’9 ” or any other way. So to avoid the confusion we defined the different text field.

### **Example:**

When Height : 5 ft 9 in

The image shows a window titled "Calculator" with a light gray background. It contains a height conversion interface. At the top, there are two input fields: "Feet" with the value "5" and "ft", and "Inches" with the value "9" and "in". Below these is a "Meters" output field showing "1.7526000000000002" and "m". To the right of the input fields are two buttons: "Convert" (highlighted with a blue border) and "Clear". At the bottom, there are three radio buttons: "Temperature", "Height" (which is selected, indicated by a black dot), and "Weight".

The output is printed as 1.753 meters.

### **Operation For Mass Button:**

This button also works as two other buttons as it also disables functions of other two buttons and the units for the mass conversion is popped.

The *massSelected()* helps the user to input the data in pounds format or the kilograms format. When the user enters the value for kilograms the value is converted into pounds and ounces.

As,

1 Pound = 16 ounces, and

1 Kg = 2.20462 pounds

### **Example:**

Hence, if the value for 7.5 kg is given by the user then the output is displayed as 16lb and 8.554oz.

A screenshot of a unit conversion application window. The window has a title bar with standard minimize, maximize, and close buttons. The main area contains input fields for different units: 'Lbs' (with 'Ex: 20'), 'lb' (with 'Ex: 3'), 'oz', and 'Kg' (with '7.5'). Below these fields are three radio buttons labeled 'Temperature', 'Height', and 'Weight', with 'Weight' being the selected option. To the right of the input fields are two buttons: 'Convert' and 'Clear'.

*Fig: In this figure the input is given as 7.5kgs*

A screenshot of the same unit conversion application window after the conversion. The 'Kg' field now displays '7.5'. The 'oz' field displays the converted value '8.554714621853066'. The 'Lbs' field now displays '16.0' and the 'lb' field displays '8.554714621853066'. The 'Convert' button is highlighted with a blue border, indicating it was the button clicked to perform the conversion. The 'Clear' button remains below it. The 'Weight' radio button is still selected.

*Fig: In this figure the output of 7.5 kgs is obtained*



## **Problem 2: Calculator**

### **Analysis**

The calculator should be able to do basic calculations as well as handle large number and numbers with decimals and store each calculation in a .txt file. Other buttons include the +/- sign, clear button, % button and decimal button.

### **Issues**

- Issue (1): Modifying the Textfield, Storing numbers, and Performing the operation.
- Issue (2): Storing, Writing, and Rewriting Calculations in .txt file.
- Issue (3): Decimals, Negatives, Percentages, and Clear

### **Solutions**

- Solution (1):
  - The basic premise of our calculator revolves around displaying a String in the textfield which is modified based upon which button is pressed.
  - Numbers are added to the end of the string with the push of the numerical buttons. Operators store the string as a double in one of two variables. The operator itself is stored as a string.
  - Each button initializes, adds on, or performs operations depending on what other variables have been initialized through if statements.
  - Example 1: first number has not been initialized. Pressing a number will add a number to the end of the string in the textfield. Pressing 'plus' will store whatever is in textfield as a double in the first variable. 'Plus' is stored as the operand. When a second number is pressed it will show up as a new number in the textfield. Pressing

minus will store the textfield as a second double variable, perform the operation, store the sum in the first variable, store 'minus' as the operator, and clear the second variable. Whatever number is then stored in the textfield will be set as the second number, a second operation will be performed when either an operator or equal sign are pressed. This continues unless cleared.

- **Solution (2):**

- In the main function, we initialize a file, calculations, with the document "calculations.txt".
- The main function checks to see if the file exists and, if not, creates a new file.
- Through the use of a PrintWriter and BufferedWriter, the function clears the file.
- If statements in the number button actions and equal button action call a method called writeCalculation that appends the file with each calculation storing the first number + operand + second number + = + output on each line.

- **Solution (3):**

- The decimal button action is simple. It checks to see if there is a decimal point in the textfield string. If there is not one, it adds it to the end of the string.
- The +/- button uses a . contains method to check if there is a negative sign. If there is one, the button returns the textfield string as a substring minus the first character. If there is not one, the button adds a negative sign to the beginning of the string.
- The percentage sign takes the first number initialized and multiplies it by the second number over 100 and stores that as the second number. If the second number has not been initialized, the % button stores the first number as the second number and then performs the same operation.
- The clear button sets the first and second number as Double.NaN and

sets the operand as null.