Get Valid Double work sheet

This code is fine:

```java
try {
    Weight = Double.parseDouble(weightField.getText());
    if ((Weight < 10) || (Weight > 200))
        throw new NumberFormatException();
}
catch (NumberFormatException e)
{
    outputLabel1.setText("You must enter Weight in KG " +
                        "between 10 and 200.");
}
```

However, with the above style of coding you end up with `try ... catch ...` blocks for each input item. So, we are reinventing the wheel and repeating code all the time.

What we need is a generic version of this code – a version that works for *any* floating point input.

Another problem is that the threshold values (10 and 200) are repeated in numerous locations ➔ Maintenance Nightmare !! A much better solution would be to use Global Constants.

We will now address these issues by creating a user defined method called `Get_Valid_Double()` which we can invoke whenever we need to get floating point user input from the user.

We will provide the method with the data it needs to do its job: the user input (as a string), the name of the data being input (for error messages, etc), and the minimum and maximum values allowed.
private double Get_Valid_Double (String Data_Name,  
String Input_String, double Min_Value, double Max_Value) 
{  
    // Declare any data you need here.  
    ____________________________________________  
    ____________________________________________  

    // Get the data (e.g. from the user)  
    // Check it isn’t blank  
    // Type check it.  e.g. convert input string to a double.  
    // Range check the data against Min and Max values.  
    // Hint: A try ... catch ... block is ideal for this !!  
    // If input data is invalid, display an informative  
    // error message and then return a value which is less  
    // than the Minimum allowed to signal to the calling  
    // program that an error occurred.  

    try 
    {  
        _______________ = Double.parseDouble (_______________________);  
        if ((_____________ < _____________)  ||  
            (_____________ > _____________))  
            throw new NumberFormatException();  
    }  

    catch (NumberFormatException e)  
    {  
        outputLabel1.setText(_____________________________________________  
                                __________________________ ________________________);  
            // An error has occurred, what is a safe value to return ?  
            // The value must indicate to the calling method that  
            // the input was invalid.  
            _______________  
    }  

    // Return the input value (or a value to indicate error)  
    // to the calling program.  
    ____________________________________________  
}
**Calling this method:**

```java
Weight = Get_Valid_Double("Weight (Kg)", WeightTextField.getText(), 10.0, 200.0);
Height = Get_Valid_Double("Height (m)", HeightTextField.getText(), 1.0, 3.0);
```

**Additional:**

- The above code displays the error in a label. Please extend the code so that an error message also appears in a Swing Error dialog.
- Place this code in a Java Applet and call the `Get_Valid_Double()` for multiple user inputs.
- How does the calling method know if an error has occurred? (Hint: see above).
- Ensure that the calling method stops – does not ask for any more user inputs or do any calculations or display results – if an error is encountered in one of the inputs.
- Implement all of this code in your Applet and make sure that it is fully working.