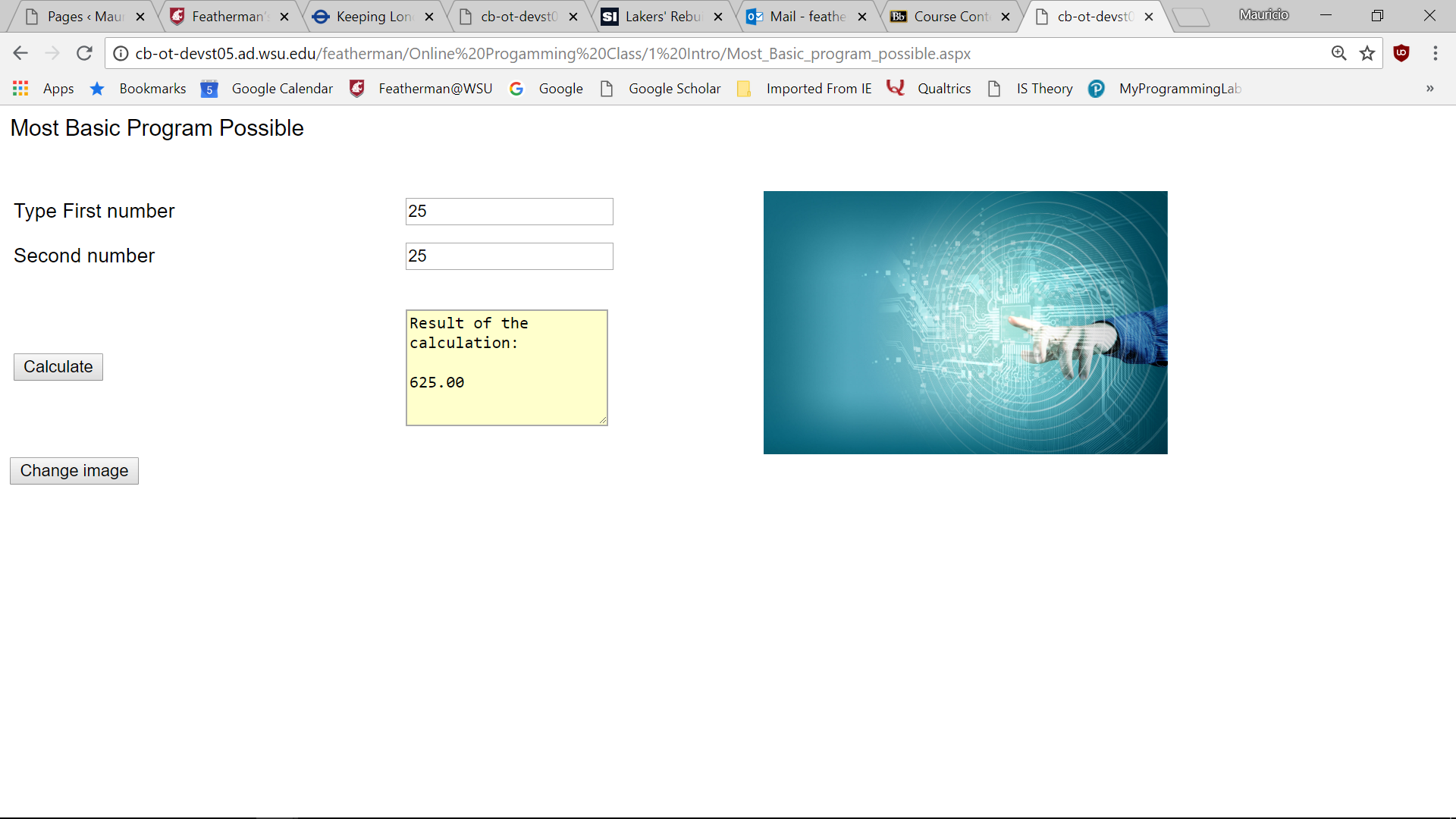
<http://cb-ot-devst05.ad.wsu.edu/featherman/Online%20Progamming%20Class/1%20Intro/Most_Basic_program_possible.aspx>



'The program creates a functioning webpage that accepts numbers that people type into some textboxes shown on a webpage. Yes even this simple program creates a cool webpage. The program uses a formula to multiply the two numbers together and displays that formatted output. Even this simple program has a lot of parcels to diagnose. Take your time to fully understand this program prior to creating a program of your own. You can perhaps create a program with different buttons that do different calculations (subtraction, division, etc.)

'Feel free to change the formula so that your program generates different results

Partial Class Most\_Basic\_program\_possible

Inherits System.Web.UI.Page

'This \*.aspx.vb page is designed to hold code and implement the interactivity to facilitate a business transaction. the aspx.vb page holds the code that runs in response to button clicks or interactions with controls on the .aspx page. So each webpage is really two files (GUI & functionality) that fit together.

' Take a look at some properties of the web controls. That is programming too! For example look at the imageURL properties of the image(box) and the imagebutton.

#Region "Calculation"

Protected Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

' 'When you are creating the web page and you drag a button onto the page, if you double click the button you are brought to this coding interface. Further a procedure is created for you. the code you type after the term protected sub and before the end sub will be run whenever the button is pressed.

'The code enclosed in this Protected sub/EndSub will perform the processing, taking input fron the human program user, checking that user entry, assigning the user entered values to variables so it can be manipulated, usually performing calculations, and finally providing some output to display. We start by creating a using a few variables - we assign values from the human to the variables.

'the next lines create variables used to store the values typed by the program user. Usually you create one variable for each piece of data the program user types in. This is a guideline not a hard rule.

Dim decFirstNumber As Decimal

Dim decSecondNumber As Decimal

'This variable used to hold the results of the calculation and then display the calculated answer.

Dim decAnswer As Decimal

'the following is IF THEN ELSE END IF code which is used to make sure the program user entered a number in the first textbox. Half of all coding web pages is validating data entered into the processor. Garbage In (bad data entry) creates Garbage Out (useless results). so we have to validate user entry so the programs don't crash. It is tedious but necessary

'the test condition to validate the first user entry.

'Does the textbox have a number in it? If yes then convert the entry to a decimal number and assign it to a decimal variable. Assignment (the = sign) reads right to left. Once you have a number assigned to a variable you could modify it (plenty on that later). The IF statement is a test condition, if the test equates to true then the code immediately after the IF statement is run. if the test equates to false then the code after the

If IsNumeric(txtFirstNumber.Text) Then

decFirstNumber = Convert.ToDecimal(txtFirstNumber.Text)

'this next code segment only runs if the prior test condition was false - in this case textbox1 didn't have a number, perhaps the human typed a word or left it blank

Else

'this code runs when the test condition equates to false. Write an error message at the top of the form and stop the program

Response.Write("Please type a number in the first input control")

Exit Sub ' stop the show. Exit the procedure.

'This Exit Sub syntax must ALWAYS be contained inside an IF THEN statement or else your code will just stop and you will wonder what went wrong.

End If

'now make sure the program user entered a number in the 2nd textbox. if they did type a number then convert it to a decimal and assign it to the

If IsNumeric(txtSecondNumber.Text) Then

decSecondNumber = Convert.ToDecimal(txtSecondNumber.Text)

Else

Response.Write("Please type a number in the second input control")

Exit Sub 'stop the show exit the procedure

End If

'now do a simple calculation, when you see an = sign the code goes right to left so the result is stored into the decimal variable we created named DecAnswer

decAnswer = decFirstNumber \* decSecondNumber

'next is the code that provides the output that gets displayed in a textbox. Notice the name of the texbox is txtOutput. Reading and fixing your own code becomes easier if you use naming conventions to self-document the program. No need to guess what the use of txtOutput is (it displays the program's output). In these set of programming modules, the course designeer establishes that text that is output (displayed) to the human always uses a textbox called txtOutput.text. Check the textmode property of the txtOutput it is set to multiline, which allows the vbNewLine syntax in the code below, that syntax adds a new line.

'BTW the program flow is as follows:

'1. At the top of the program provide a little documentation of what the procedure accomplishes.

'2. create the variables followed by error checking of the user entry and assigning the user entry to the variables.

'3. Next perform the processing

'4. Finally display the output. Guess what numeric formatting in applied? Currency with two decimal places.

'BTW textboxes can only show text so you have to turn it back into a string, here with numeric formating with 2 decimal places. Also important here is concatenation. the & symbol is used to connect two objects into a string. The textbox has a text property that displays content, the content has to be connected into one string. Here we connect some text, some blank rows, and a variable that is formatted into a number. You have to get good at using concatenation.

txtOutPut.Text = "Result of the calculation: " & vbNewLine & vbNewLine & decAnswer.ToString("N2")

End Sub

#End Region

Protected Sub Button3\_Click(sender As Object, e As EventArgs) Handles Button3.Click

'here we just change the picture that is displayed.

Image1.ImageUrl = "tablet.jpg"

'if a procedure is not connected to a button click event or other event, we can call the procedure any time we need to. here run the clear procedure to clear up the form.

Call Clear()

End Sub

Protected Sub Clear()

'This is a general procedure not connected to any button procedure. This procedure can be called from different procedures on the page as shown above.

'the code clears out the contents of teh textboxes (setting the text property of a text box to nothing, clears out any text).

txtFirstNumber.Text = Nothing

txtSecondNumber.Text = Nothing

txtOutPut.Text = Nothing

End Sub

End Class