Lab Document and Questions
Name: Key

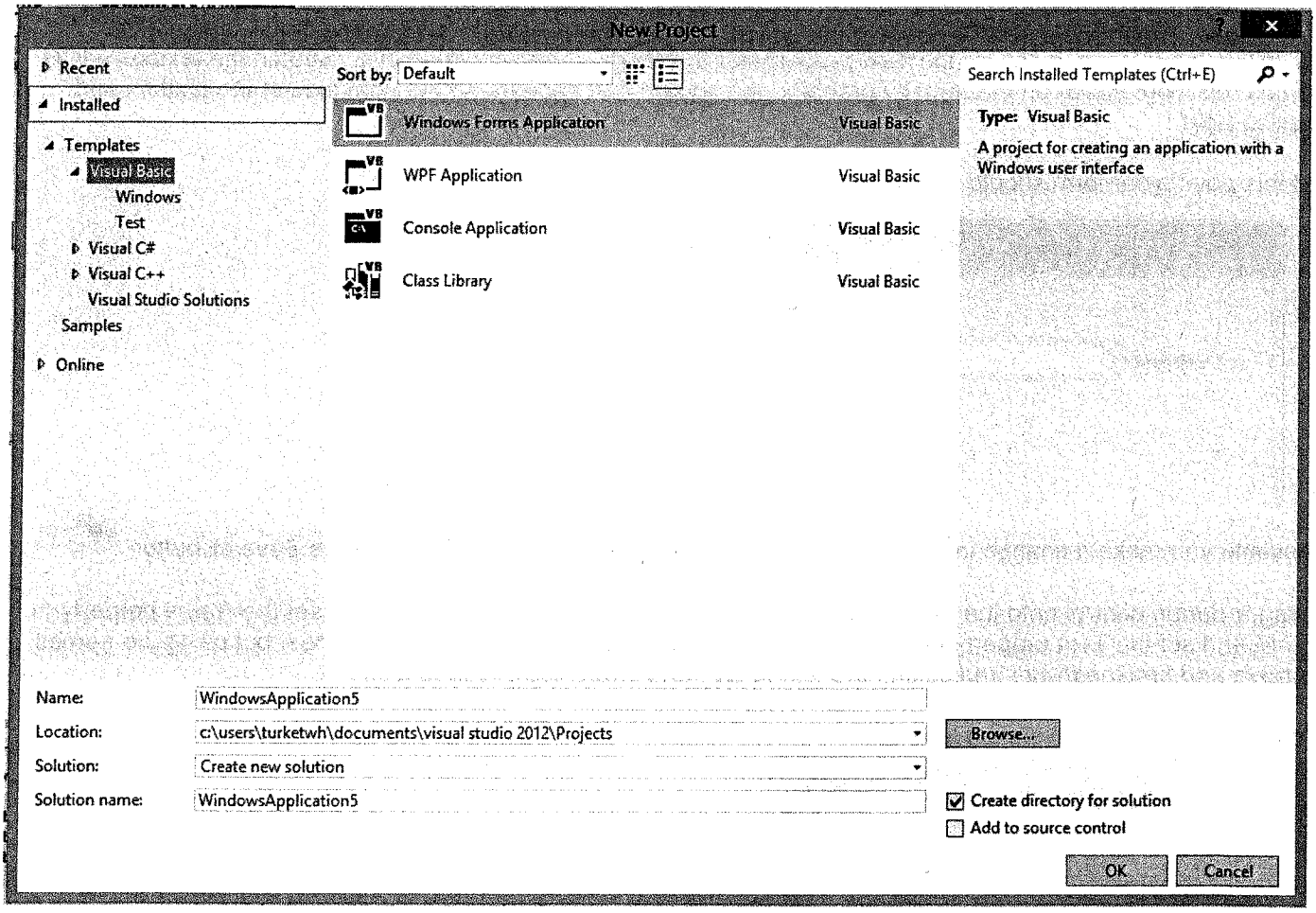
Lab #2, 09/05/2013
WFU Username: Key

CSC 111E: Lab #2 - Developing Simple GUIs and Event Handlers
Lab Date: Thursday, 09/05/2013 Due Date: Friday, 09/06/2013 @ 5:00pm

Purpose: The purpose of this lab is to have you work with designing graphical user interfaces and writing event handlers to support responding to button presses.

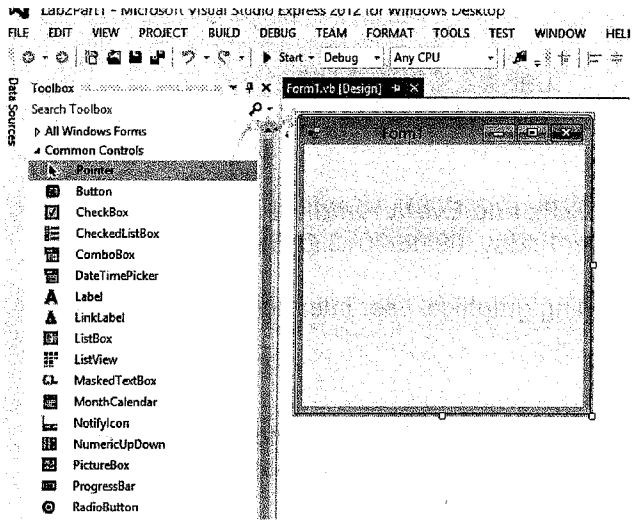
Greeting Generator
To become more familiar with developing GUIs and event handers, we will start with a simple program that writes a greeting on the screen depending on the button that a user presses. Keep that goal in mind as we progress through developing the program.

To start, open Visual Basic and under the *File* menu select the *New Project* menu option. A window should appear similar to below.



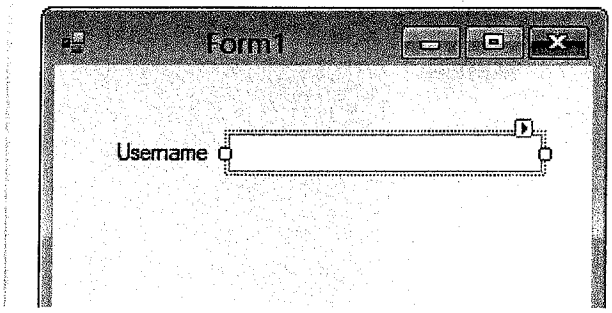
Select (if not already chosen) *Windows Form Application*. In the *Name* field type *Lab2*, then hit *OK*.

After hitting OK, you should (ideally) see a blank form called *Form1* in the middle of the screen and a toolbox of controls on the left-hand side of the screen. If you do not see the toolbox, ask the instructor(s) for help. A screenshot of what your screen should look like at this point is at the top of the next page.



Designing the program: First drag from the toolbox a *Label* to the top left of the form. Go to the Properties field in the lower right corner and change the *Name* property of this label to *lblUser* and set the *Text* property to *Username*. Then drag a *TextBox* next to the *Username* label on the form. Set the *Name* property of the TextBox to *txtUser* and ensure the *Text* property for this TextBox has no value in it. Stretch the textbox a little to take up more space on the form by grabbing the square that appears on the right side of the textbox and pulling right.

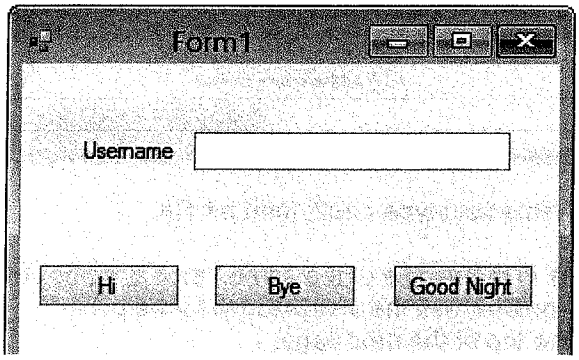
At this point, your form should look like:



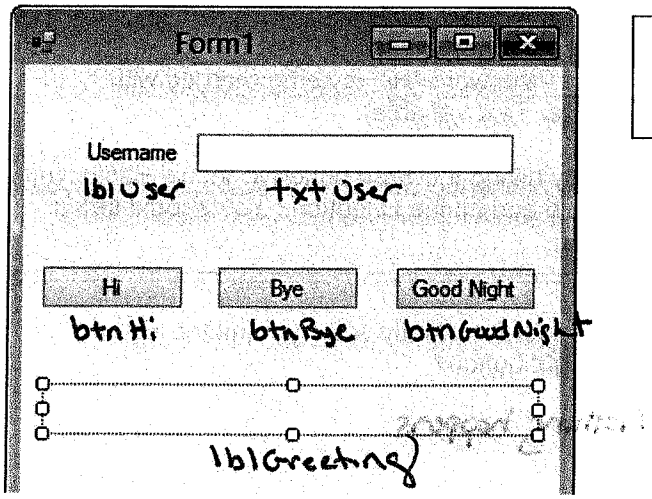
Anytime you make a change in the program, save your updated program by hitting the Save all button 

Drag a *Button* control onto the form, just under the word *Username*. For this control, set the *Name* property to *btnHi* and set the Text property to have the value *Hi*. Do the same thing two more times, but using the names *btnBye* and *btnGoodNight* and setting the text to *Bye* and *Good Night* respectively.

At this point, your form should look like:



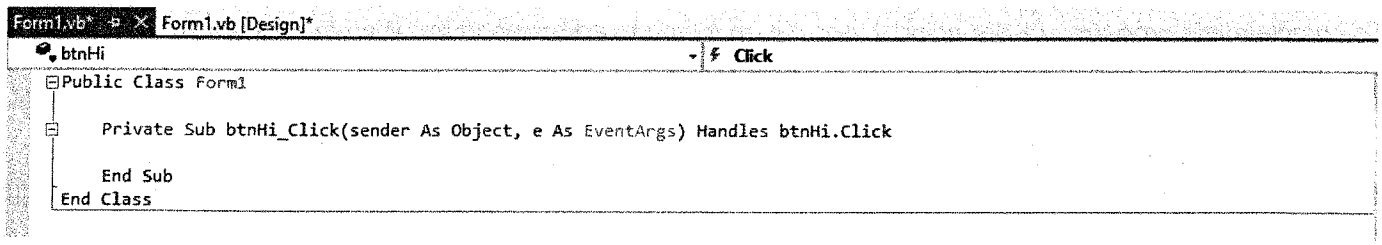
Finally, drag a Label control onto the form underneath the buttons. Change the label control's *Name* property to *lblGreeting*, the *Text* to nothing (remove whatever is in the Text field), and change the label's *Autosize* property to *false*. When the *Autosize* property has been set to false, stretch the label, by grabbing the right-most white square on the label, so that it is about as long as the three buttons' combined length.



Question 1: Underneath each of the six controls in this picture, write the value associated with the control's *Name* property.

Now our goal will be to print in the label referred to by *lblGreeting* a phrase depending on the value entered for the username and depending on the button pressed. As an example of what we want to do, assume *Bob* was entered as the username value and the *Bye* button was pressed. In this case, *lblGreeting* should show: *Bye Bob* Similarly we should be able to get the screen to show *Hi Bob* and *Good Night Bob*.

Adding event handlers: To achieve our goals listed in the previous paragraph, we need to add event handlers for each of the three buttons. To start, we'll add an event handler to the button labeled *Hi*. Double click on the *Hi* button and you should see a new screen open. This screen is the Visual Basic *code* screen, where we will add instructions – the one with the form (that we were previously on) is the *GUI design* screen.



This procedure provided to you and shown in this screen, *Private Sub bntHi_Click(sender As Object, e As EventArgs) Handles btnHi.Click*, is a place where we can put instructions that we want to run when the *Hi* button is clicked. This will support our response to the event of the "Hi" button being clicked.

What do we want to happen when the *Hi* button is clicked? Again, as an example, assume *Bob* was typed in the *Username* textbox. We want the *Text* of the greetings label (*lblGreeting*) to be updated to show "Hi Bob". To do this, we need to set the *Text* property of *lblGreeting* to the result of concatenating (merging) together two strings: the string "Hi " and the *Text* value in the *txtUser* TextBox. This can be achieved by adding the following instruction just after the words *Private Sub bntHi_Click(sender As Object, e As EventArgs) Handles btnHi.Click* and just before the words *End Sub*

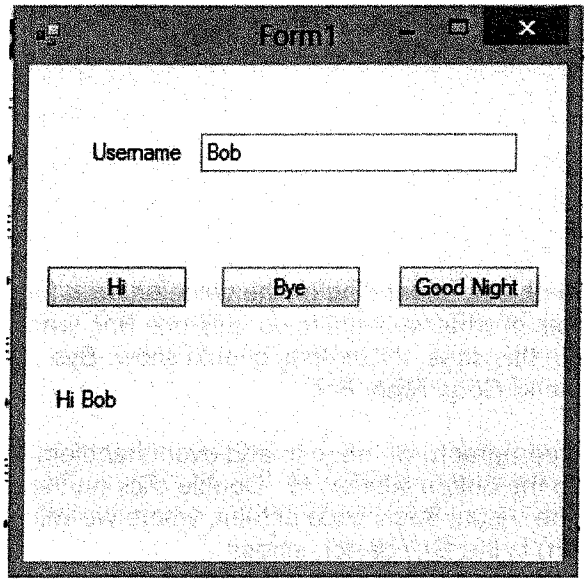
```
lblGreeting.Text = "Hi " & txtUser.Text
```

Go ahead and copy or type the line above into the space suggested (after *Private Sub...btnHi.Click*, before *End Sub*).

What is the instruction above doing? First, remember that the assignment operator (discussed last Thursday) copies whatever value is on the right side of the assignment operator into the variable on the left side. So, this is storing a value in the Text property of the greeting label (as the greeting label is represented by the variable on the left hand side of the assignment operator)

In Visual Basic, the symbol & is an operator that allows one to merge together strings (words). So, in this instruction, the & is being used to merge two words into one word – the word "Hi " is being merged with whatever the username entered was (the value sitting in the *txtUser.Text* variable).

Let's run the program and see the code written so far in action. By hitting the *Start* button in Visual Basic, you should see an application start. Type a name in the *Username* field and hit the *Hi* button. You should see a greeting appear in the label at the bottom of your programs screen.



Question 2:

A. What happens currently when you hit the *Bye* or *Good Night* button?

Nothing happens

B. Explain your understanding of why this happens:

We have not written an event handler for these events.

To complete this program, we are going to leave it up to you to add support for when the *Bye* button is pressed and the *Good Night* button is pressed – the process very similar to the code that was written to support the *Hi* button.

Answer Question 3 below before writing any code on your computer! The next page will step you through writing the actual code in Visual Basic.

Question 3:

A. Suggest the appropriate instruction to write in the *Bye* button event handler?

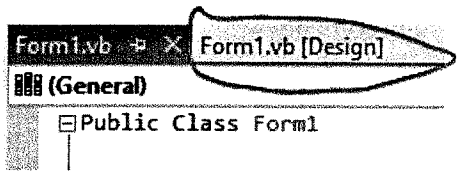
lblGreeting.Text = "Bye␣" & txtUser.Text

(␣ = space)

B. Suggest the appropriate instruction to write in the *Good Night* event handler?

lblGreeting.Text = "Good␣Night␣" & txtUser.Text

Before we can actually write these handlers in Visual Basic, we need Visual Basic to provide the (*Private Sub, End Sub*) of the event handlers for us. Close the running *Form1* program by hitting the *X* button. In Visual Basic, click on the tab that says *Form1.VB [Design]* to hide the code we have been writing and to show the design screen again.



Double-click the *Bye* button and we will see the code screen re-open. Type in the appropriate code that you wrote in your answer to Question 3, Part A in the newly generated event handler procedure.

To add the event handler for the *Good Night* button, first click on the tab that says *Form1.VB [Design]* to hide the code we have been writing and to show the design screen, and then double-click the *Good Night* button to see the code screen again. Type in the appropriate code that you wrote in your answer to Question 3, Part B in the newly generated event handler procedure.

Once your changes are made, run the program using the Start button of Visual Basic.

Question 4:

A. Did the program work as expected?

Yes or No

B. If not, what was wrong? How will you fix it?

Explanation | Fix As Needed (No answer ok if worked as expected)

If you had errors in your program, fix them (with the help of the instructors if needed) before continuing.

Buttons are not the only controls that cause events, nor is clicking the only way for an event to be triggered. Let's add one last component to this program – an event handler that responds when the user types in the *Username* textbox.

In design mode of Visual Basic, double-click on the *Username TextBox*. This will add one more procedure to the code for the program – the second procedure shown below named "txtUser_TextChanged". Note how this procedure is different from the previous ones. In particular, it responds to a different event – to the text in the txtUser box changing (TextChanged) instead of to a button being pressed (Click).

```
Private Sub btnGoodNight_Click(sender As Object, e As EventArgs) Handles btnGoodNight.Click
    lblGreeting.Text = "Good Night " & txtUser.Text
End Sub

Private Sub txtUser_TextChanged(sender As Object, e As EventArgs) Handles txtUser.TextChanged

End Sub
```

Between the *Private Sub* and *End Sub* of this new procedure, add the following statement:

```
lblGreeting.Text = "User is typing: " & txtUser.Text
```

Run the program by pressing the Start button in Visual Basic and answer the questions below.

Question 5:

A. Describe what you see in the greeting area as you type into the Username textbox.

As the user is typing, the label is being updated on the fly

B. Is that what you expected to see? Why or why not? Try to relate your answer back to the instruction that was used: `lblGreeting.Text = "User is typing: " & txtUser.Text`

any reasonable answer ok

(ideally they will discuss that the event is being responded to any change in text, so any typing in username triggers a response updating the greetings label.

Close the running program by hitting the X exit button, make sure your work is saved, and then close Visual Basic.

Submission

To submit this lab for grading, you need to do TWO things by Friday at 5pm:

- Submit the answers to the questions on this lab document on paper to Prof. Turkett
- Zip and upload your project into Sakai under the Assignments, Lab2 link.

Your grade will be based 60 points on the questions (12 points each) in this lab and 40 points on the code you submit (10 points per event handler procedure, checking each works as expected).