

Module 2 – Creating the graphical user interfaces.



To continue with this activity, you can open the HelloVisualCSharp application created in the previous week, or you can create a new project.

One main task in creating Windows desktop applications is to design and create the user interfaces through which the end-users will interact with your application to perform some tasks. When creating these visual interfaces, you will use three components in the Visual Studio Environment: (1) Toolbox, (2) Form (in the Designer view), and Properties window. Please see Figure 1 below.

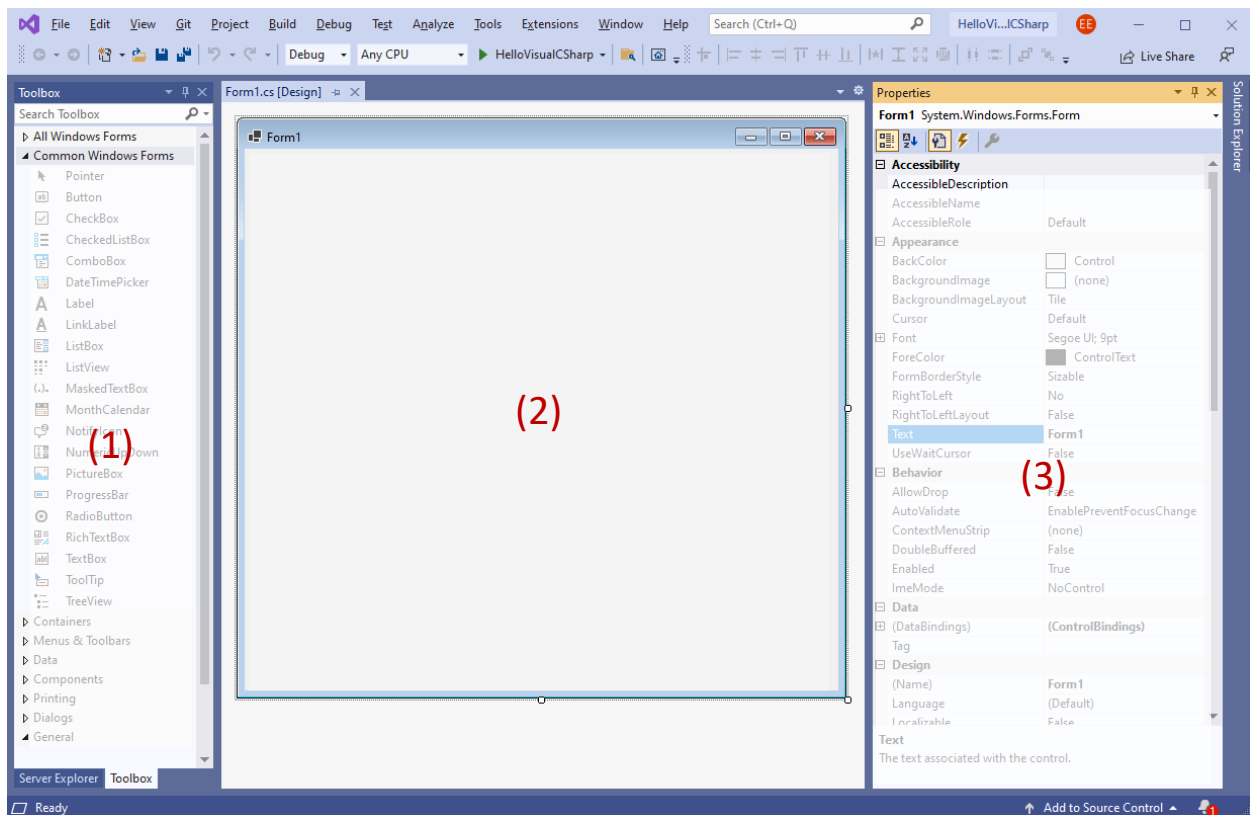


Figure 1. Visual Studio components for creating GUI.

The basic flow when creating a GUI can be described like this: Form controls (such as a button) are dragged from Toolbox and dropped on a desired location within the Form (its location can also be changed later). Then, the Properties window is used to view and change some characteristics (name, shape, colour, etc.) of the controls placed on the Form.

In the Properties window (see Figure 2), the left column displays the names of the properties, and right column shows their values. The new value for a property can be directly entered into the box where the current value of the property is displayed.

1) An example: Changing the Form title.

When a new project is loaded or created, initially the Properties window will list the properties of the Form. For example, in the figure below, the properties of **Form1** are displayed in the Properties window. Properties are by default categorized by certain sections such as Appearance or Behaviour.

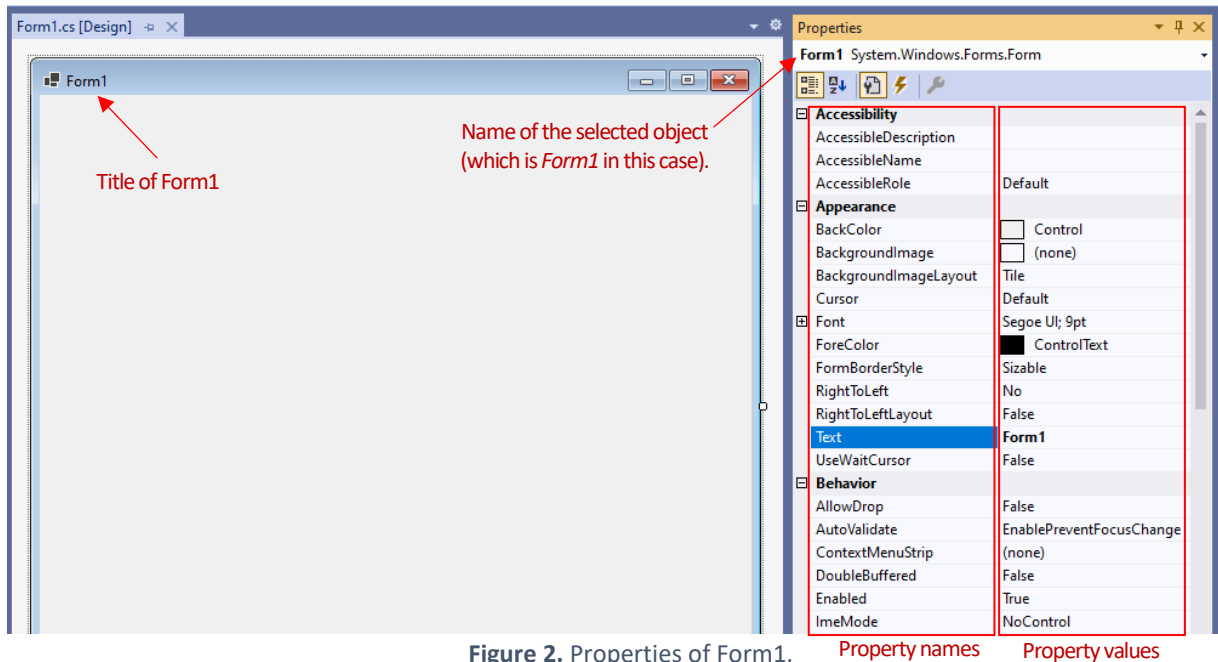


Figure 2. Properties of Form1.

In the figure above you may notice that the title of the form is *Form1*. This title will be visible to our users. However, it is not descriptive and relevant. To change this default value, we need to use the **Text** property, which is currently set as *Form1* as you may see above. When a new form is created, its Text value is set to be the same as the form name, which is a default behaviour.

Please remove the Form1 text, and instead, type *Main Form* and then press Enter. You should notice an immediate change in the title of the form. Figure 3 below shows the updated title for Form1.

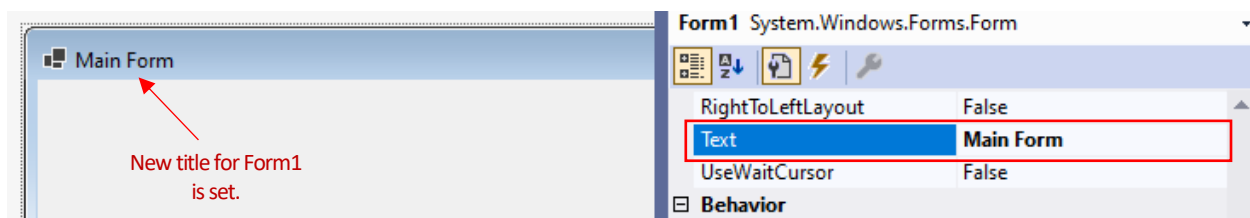


Figure 3. Setting a new title for Form1.

2) Difference between the Text property and the Name property

In the previous section, we have learned how to change the *title* of the form using the **Text** property. Actually, **Text** is a very common property that almost all popular controls possess. For example, if you want a button with “Submit” text, then you need to set its **Text** property to “Submit” (as we did for the form). However, you may notice that at the top of the Properties window (see Figure 4), **Form1** is displayed as the name (instead of “Main Form”). This is because we change the **Text** property, not the Name property.

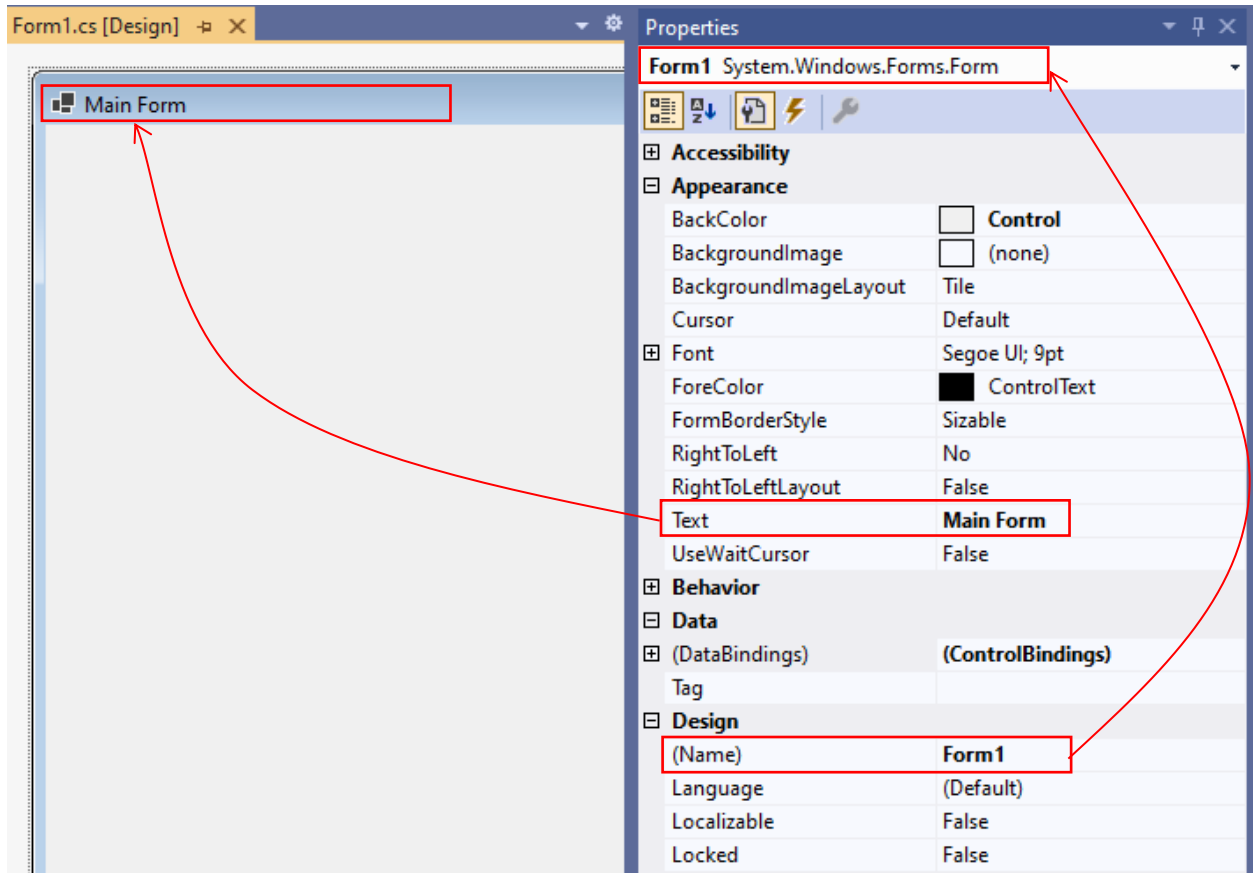


Figure 4. Setting a new title for Form1.

Each property *must* have a unique name set through the **Name** property. This name serves as a unique identifier for the form controls. When you need to refer to a control within your code, you will need to type the name of the control in order to perform any operations related to that control (e.g., clicking a specific button, reading the user input from a specific textbox).

To change the **Name** value, please delete Form1 value and type *frm_main*. Then, press enter to update the name value. As you can see in Figure 5, the new name value (*frm_main*) should appear at the top of the Properties window.

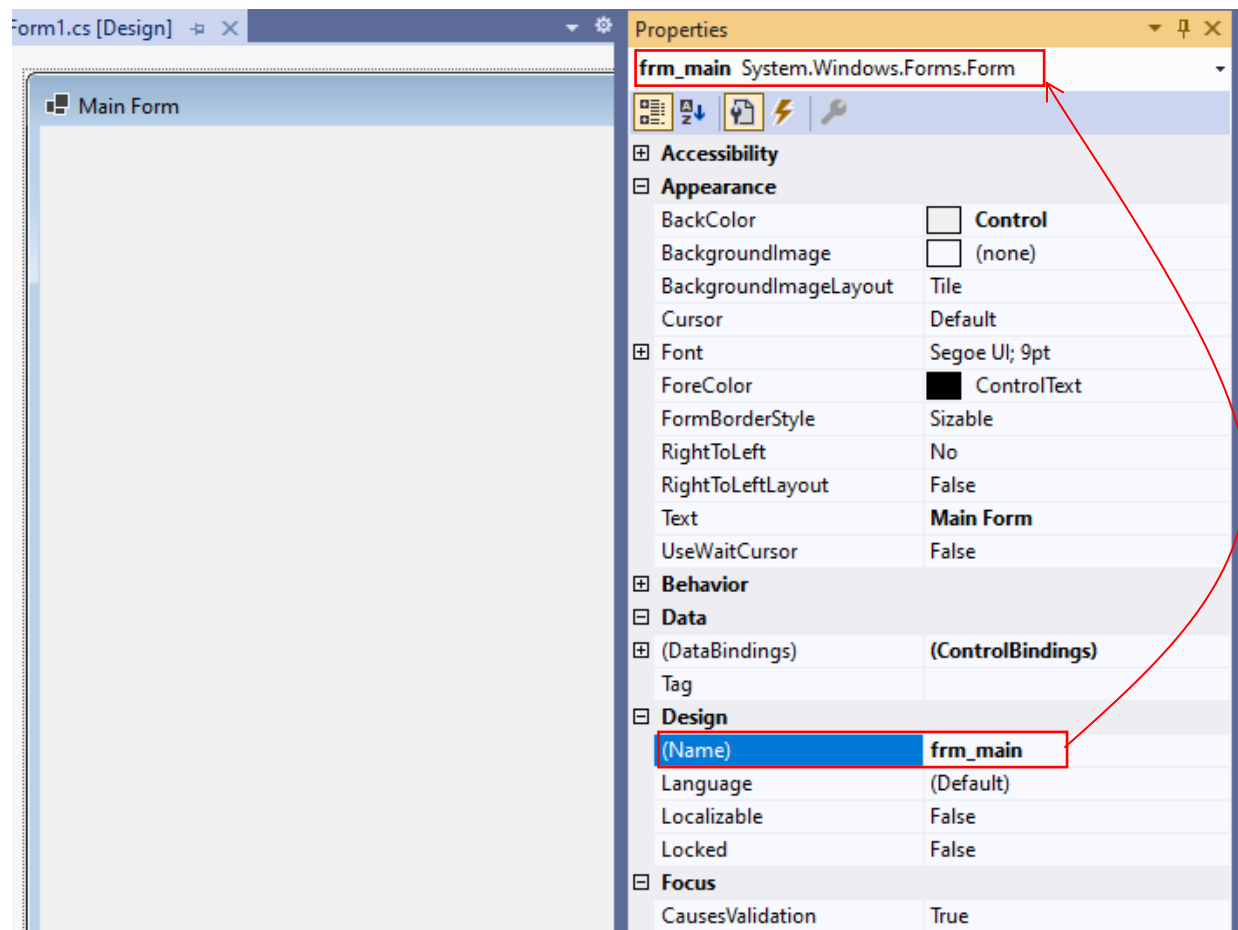


Figure 5. Setting a new name for Form1.

3) Adding controls to a form

To add controls to a form, you should first open the Toolbox window. As you may remember, The Toolbox displays all available controls divided by different sections (see Figure 6).

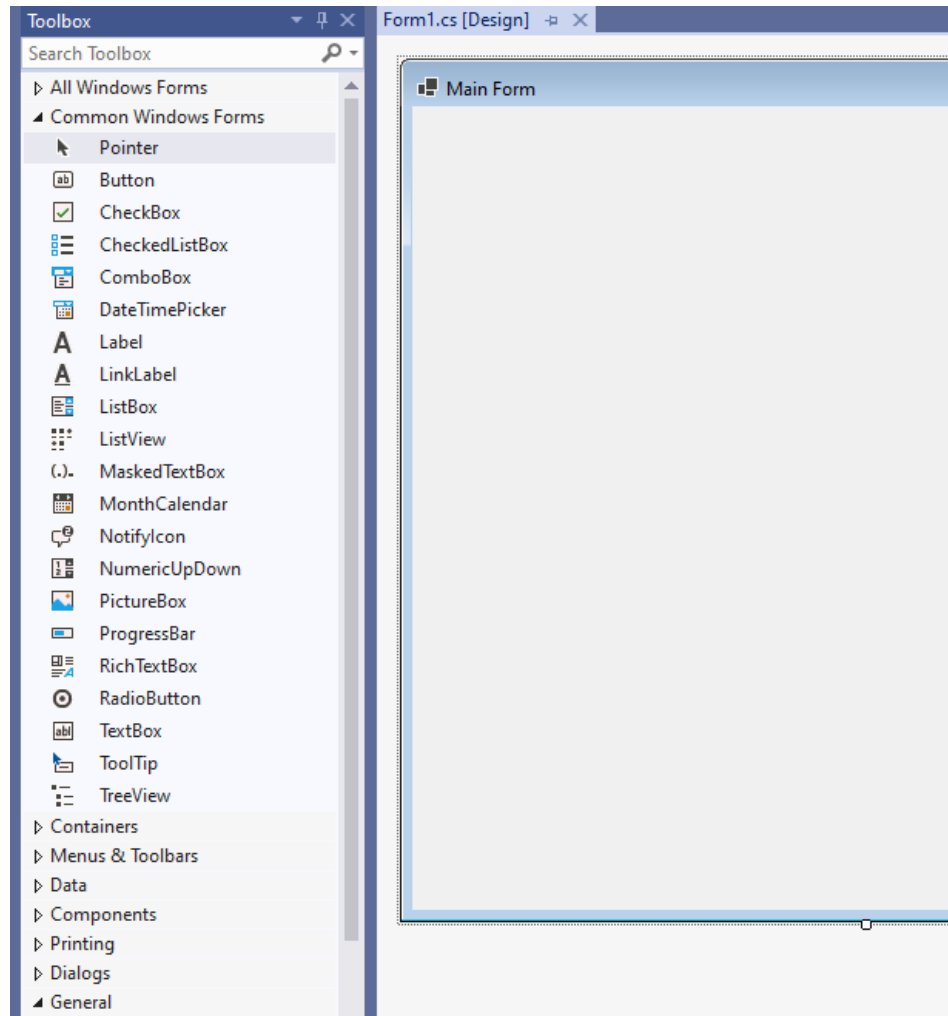


Figure 6. The Toolbox Window.

We will add two button controls to the form. You can add controls to a form in two ways: (1) double-clicking on a control in the Toolbox or (2) dragging and dropping the control on the form. Use the first approach (double-click) to add the first button and apply the second approach (drag and drop) to add the second button.

As you can see in Figure 7, the first button added through double-click will appear on the top-left corner of the form. The second button will be placed where you choose to drop it.

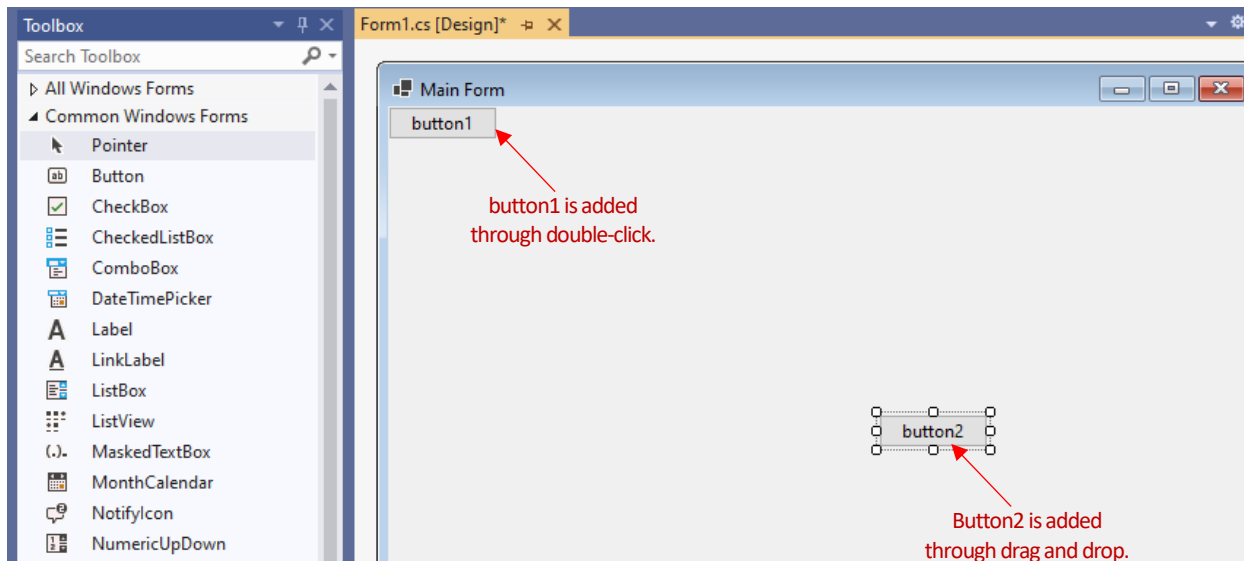


Figure 7. Adding two button controls to the form

By default, button text is initially set to the *name* of the button. First button's name is set to *button1* automatically (therefore, its text property is also "button1"), and the second button's name is set to *button2* (therefore, its text property is also "button2"). This automatic behaviour would be repeated, if you added a third button, which would have the "button3" text.

4) Resizing and moving the controls inside a form

When you select a button (by clicking on it once with your mouse), you will notice that there is a bounding box with resizing handlers (little squares on the edges of the button) around the button. For example, in Figure 7, *button2* is selected. When a control is selected, you can use the resizing handlers to change its width and height. Please resize *button1* and *button2* as shown in Figure 8.

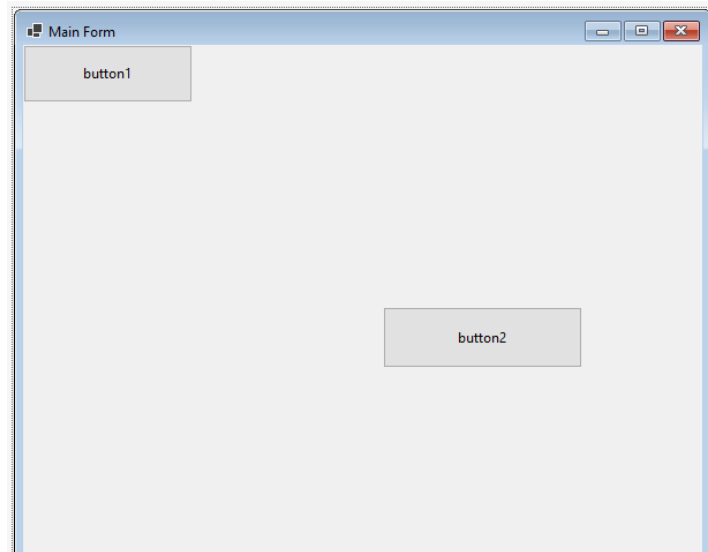


Figure 8. Resizing the buttons.

To move a control to a different part of the form, you can use your mouse. You should click inside a control and (without releasing your mouse) move your mouse to change the location of the button as desired. Use your mouse to move *button1* and *button2* to obtain the following design shown in Figure 9.

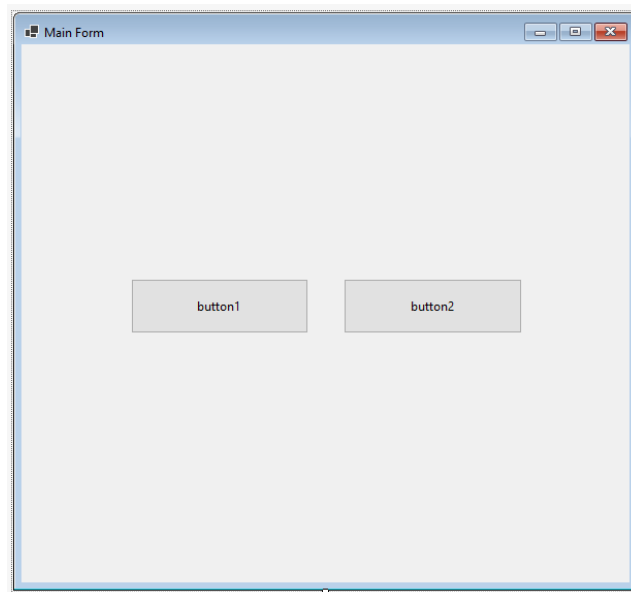


Figure 9. Placing the buttons next to each other.

Once buttons are next to each other, you may notice a slight difference in their height. If you try to change the height of a button, a blue guiding line may appear (see Figure 10). This line will help you set the button height exactly same as the other button. You can also use the Properties window to view and change the *width* and *height* of the buttons.

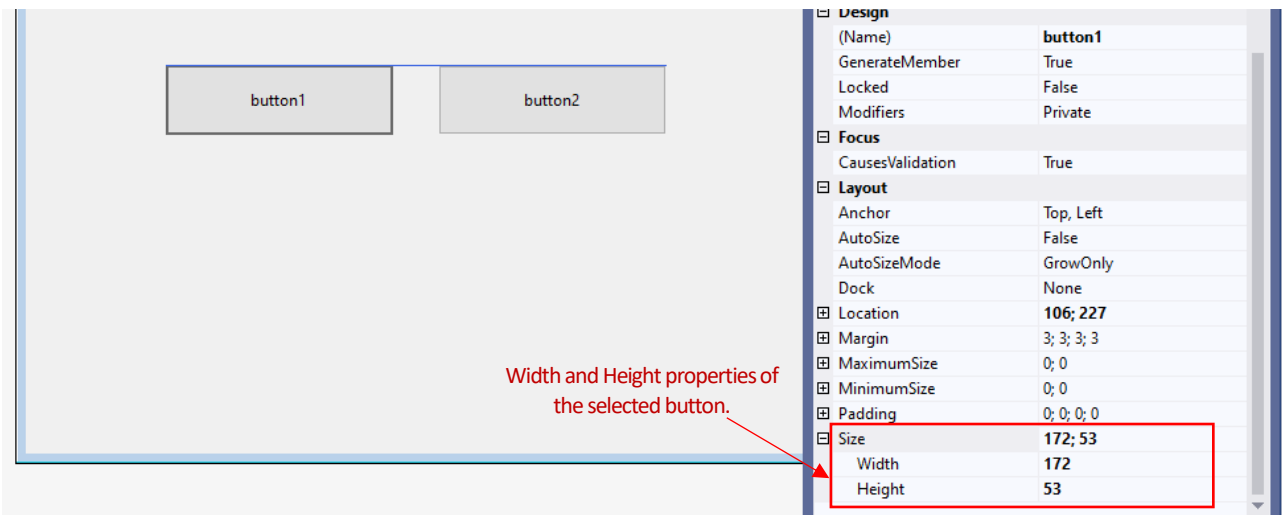


Figure 10. Placing the buttons next to each other.

5) Changing the buttons' texts and names

Currently, the buttons in our form have a default name and text (i.e., button1 and button2), which has no specific meaning. The names and texts of buttons should convey their functions and purposes. For example, let's assume that we will use button1 for displaying a message, and button2 for cancelling the message. Based on this desired functionality, we will change the buttons' names and texts.

Please change the text of the first button to *Show Message*, and its name to *btn_showMessage*, as in Figure 11.

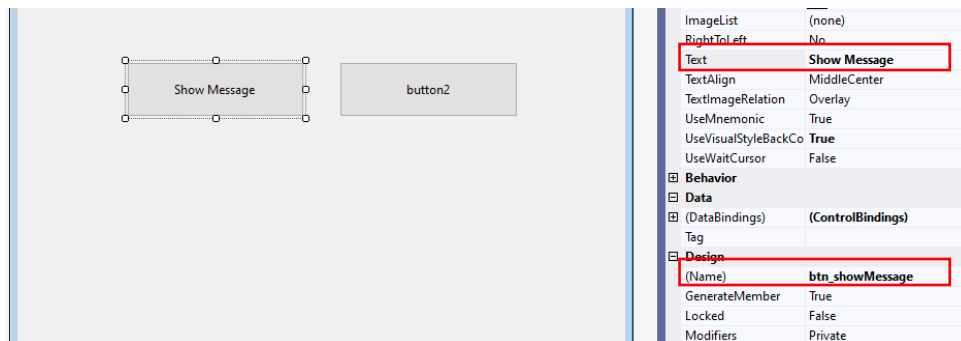


Figure 11. Changing the name and the text of the first button (button1).

Similarly, please change the text of the second button to *Cancel*, and its name to *btn_cancel*, as you may see in Figure 12.

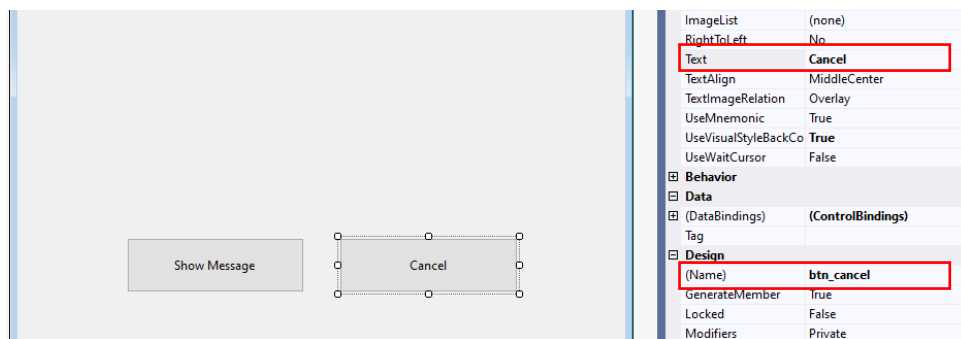


Figure 12. Changing the name and the text of the second button (button2).

5) Naming conventions for controls

For each control you add to a form, you should define a proper name that reflects the purpose of the control. These names serve as the identifiers for the controls. There are several restrictions when naming a control:

- The name can contain letters (a-z or A-Z), or an underscore character (_),
- The digits (0-9) are only allowed after the first character,
- The name cannot contain spaces.

Some examples of **valid** names : showMessageButton, _usernameTextbox1

Some examples of **invalid** names : display Error, 1textbox, label*1

It is a good practice that you choose and follow a standard procedure when naming controls throughout your project. One popular naming convention is the **camelCase** naming convention, in which the first letter should be a lowercase letter, and the first character of any subsequent word should be an uppercase. Some example names of camelCase could be *errorMessageLabel*, *cancelOperationButton*.

I strongly recommend dividing a control name into two parts using underscore (_) as seen below:

[control type] _ [control purpose]

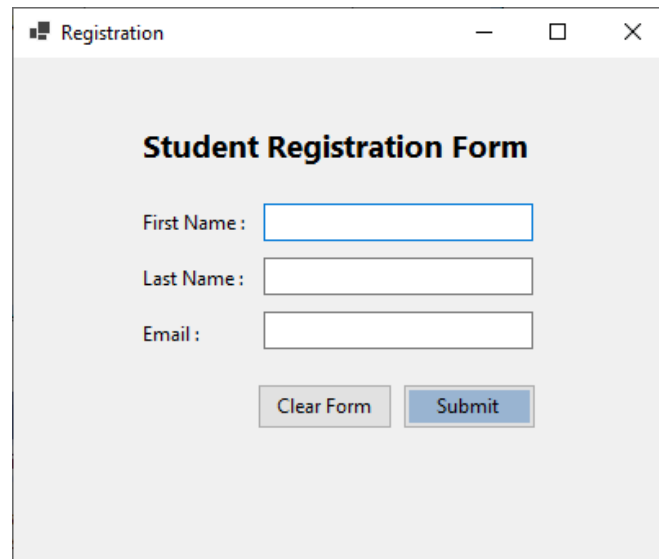
First part (before the underscore) should indicate the control type which could be button (*btn*), label (*lbl*), textbox (*txt*), and so on. The second part (after the underscore) should communicate the purpose such as *displayError* (if it is used to display the error messages). Here are some example control names following this approach:

txt_firstName, btn_submitForm, lbl_errorMessage, pic_userPhoto, check_Answer.

Please note that in these examples, the control type is shortened (e.g., label → *lbl*). You may choose to use any other shorter (or longer) text to indicate control types, but just be consistent throughout your code.

6) Creating a student registration form

We will build a student registration form by using what we have learned so far. This form will have a header text, and textbox fields to enter first name, last name, and the email of the student. There will be also two buttons (one is to clear form, and the other one is to submit the information). The final version of the form will look like Figure 13 shown below. We will add functionality to the buttons in the next section. We will only focus on creating the user interface.



The figure shows a Windows application window titled "Registration". Inside the window, there is a form titled "Student Registration Form". The form contains three text input fields labeled "First Name:", "Last Name:", and "Email:". Below these fields are two buttons: "Clear Form" and "Submit".

Figure 13. Student registration form.

6.1 Changing the form properties.

We will start with changing some features of the form. First, we want to name our form as *frm_registration* instead of Form1, which is the default name assigned by the system automatically. Please set the **(Name)** property to *frm_registration* (see Figure 14).

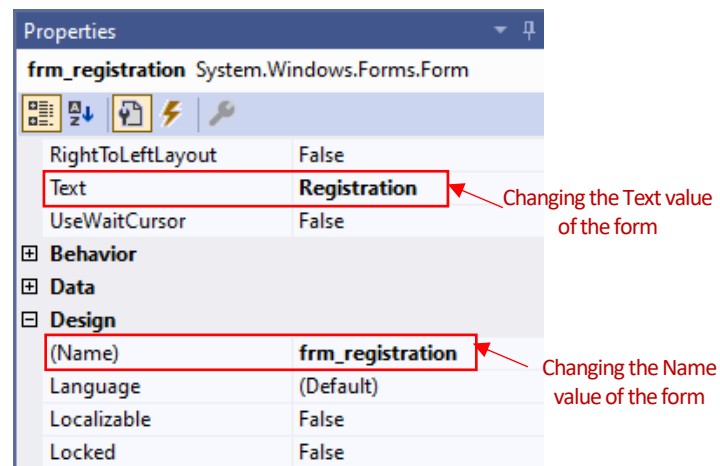


Figure 14. Changing the form name and title.

Also, we want to change the title of the form to *Registration*. We can achieve this by changing the **Text** property of the form. As seen in Figure 14, please type *Registration* for the **Text** field.

Next, we will change the size of the form. This can be in two ways. You can use the resizing handlers on the right or bottom borders of the form to shrink/enlarge the width and height as you desire. Alternatively, you can also set an exact width and height (in pixels) in the Properties window. In our case, we want our form to have a width of 600 px and height of 400 px.

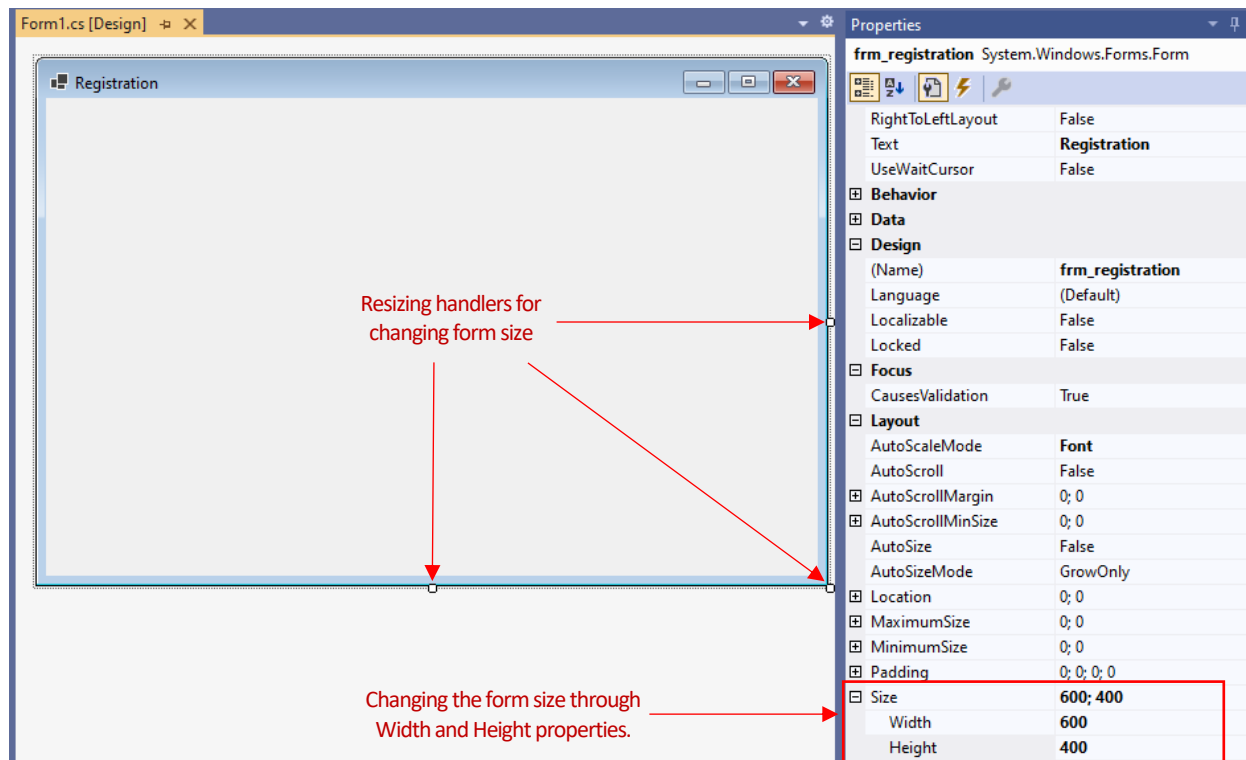


Figure 15. Changing the form size.

6.2 Adding a title (header text) using the label control.

As you see in Figure 13, at the top of the form, there is a header text saying, “Student Registration Form”. Typically, label controls are used to place some descriptive text at some parts of a form. We will add a label control and change its font style to display the intended header text.

Please drag and drop a label control around top center part of the form as shown in Figure 16. It will automatically have *label1* as its name. The text of the label will be set as its name as a default behavior. That is why, *label1* text is displayed when the label is added for the first time.

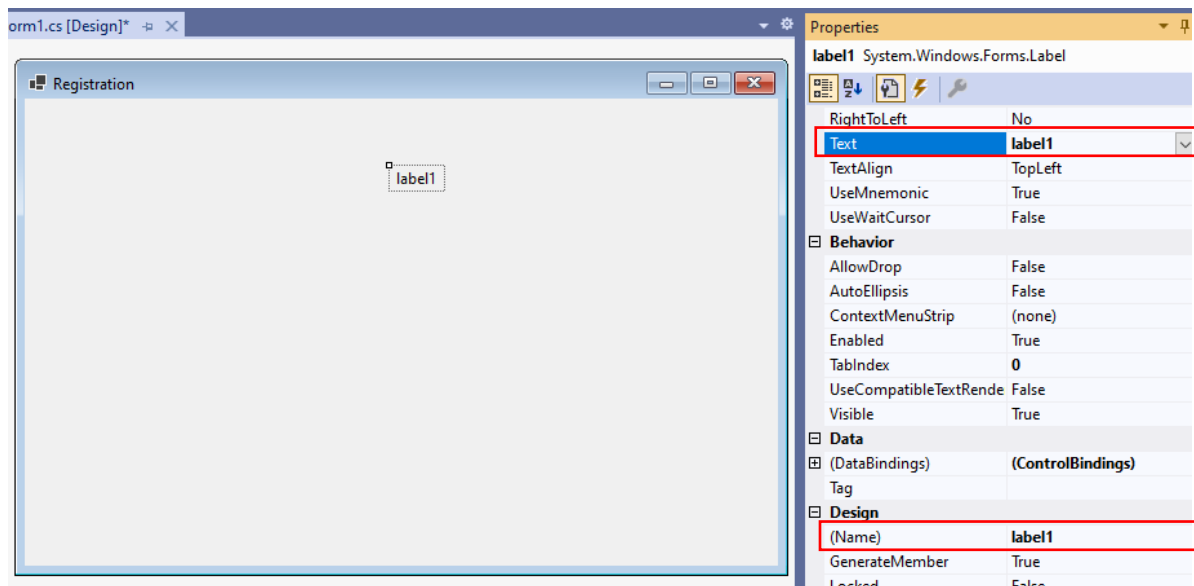


Figure 16. Adding a title.

After adding a control to a form, first thing you would want to do is mostly to change its Text and Name properties. As seen in Figure 17, please change the label text properly using the **Text** property. Similarly, please assign a descriptive **Name** for the label that reflects its purpose. Last, please change the font **Size** to 12 and set the **Bold** property to *True*.

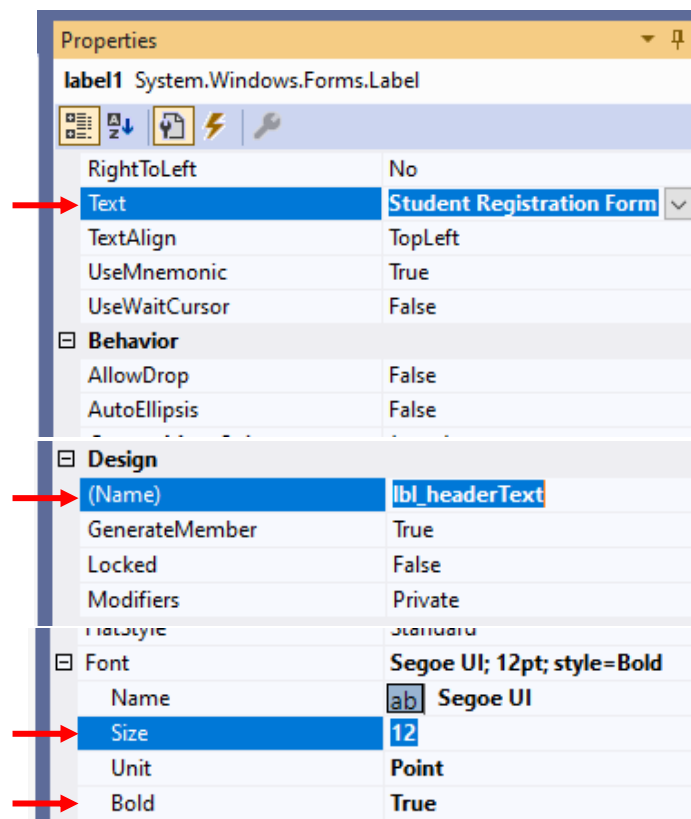


Figure 17. Changing the properties of the label control.

After all these changes your form should look like Figure 18:

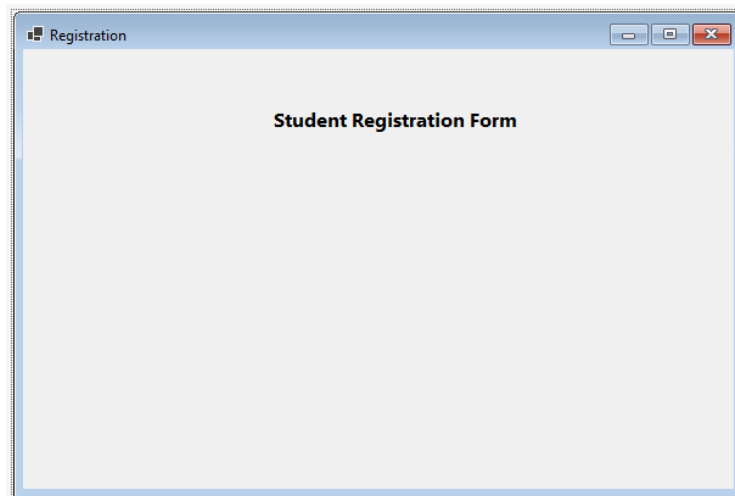


Figure 18. The form with the header text added.

6.2 Adding a label and textbox for the first name field.

To request a textual user input (such as name, address, description, etc.), we use the Textbox control. Textbox controls appear as a text field where the users enter some text. Usually, textbox controls are accompanied with a label, which includes some sort of short description to inform the users about what type of information they should enter into the text field.

As you may remember, in our form (see Figure 13), the first field requests the first name of the users. We will add a label control and then next to it, we will add a textbox. Then, we will change their text and name properties. Some suggestions for possible name and text values are provided in the Figure 19 below. Please update the text and name properties for both label and textbox control.

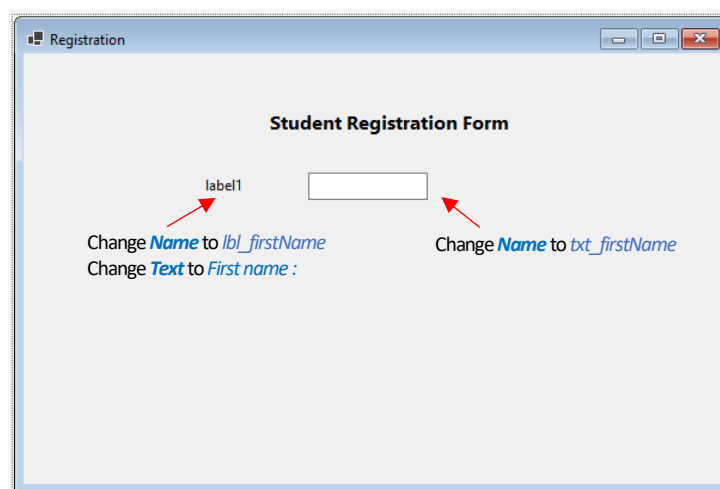


Figure 19. Adding a label and control for the first name field.

You can resize the textbox as you desire. You can also move the controls around to properly position them. After these changes, you should obtain a similar design with the form displayed in Figure 20.

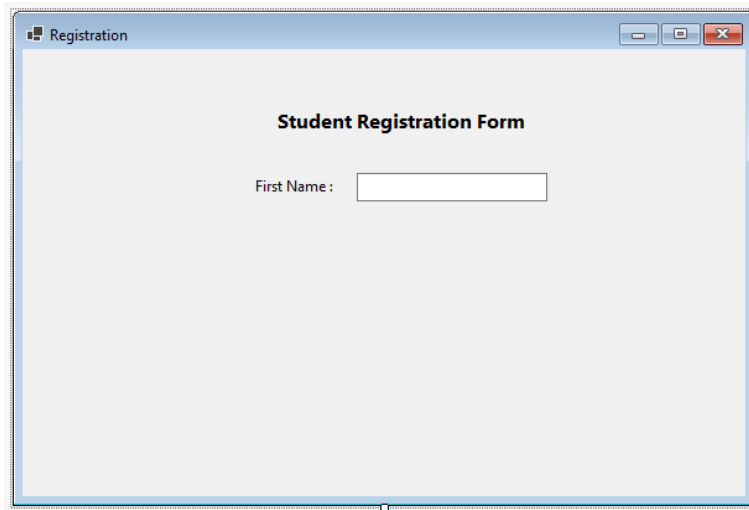


Figure 20. Registration form with the first name field added.

We will continue our design with the Last Name field. As for the First Name field, we need to add a label and a textbox. To speed up the process, instead of adding new controls, we can copy and paste the existing label and textbox controls (created for the first name field).

To do that, please select two of the controls by clicking on each of them (only once) while holding the *Shift* button in your keyboard. Alternatively, you can also use the mouse. Next, do a right-click on any of the selected controls, and click on **Copy** as shown in Figure 21 below.

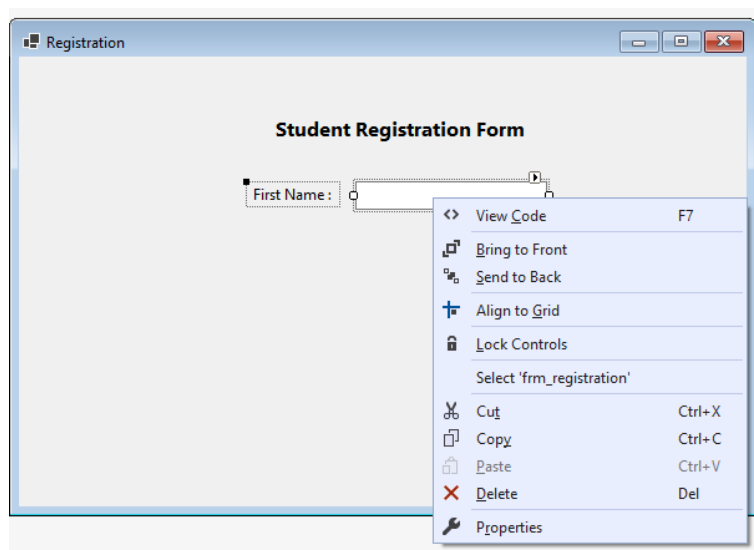


Figure 21. Copying the textbox and label controls.

To paste the copied items, please do a right-click on an empty area of the form, and in the pop-up menu, click on *Paste* (see Figure 22).

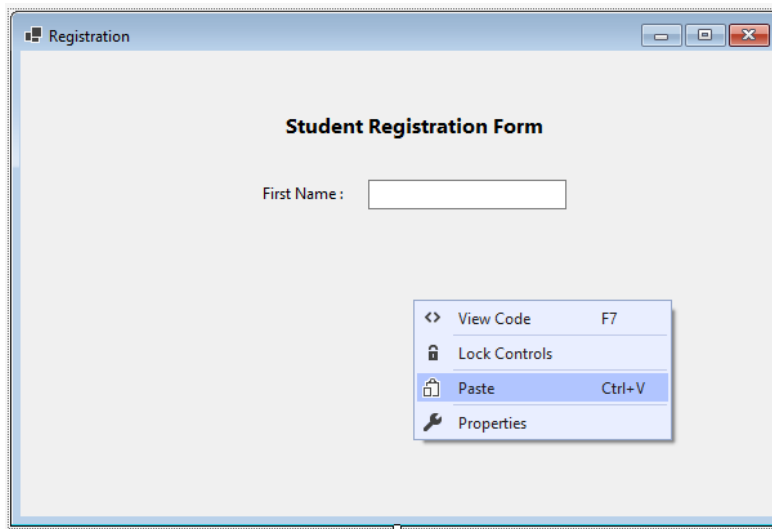


Figure 22. Pasting the textbox and label controls.

The duplicate of the label and textbox controls for first name field should appear towards the bottom of the form (see Figure 23). Please quickly change the name and the text properties of these fields as suggested in Figure 23.

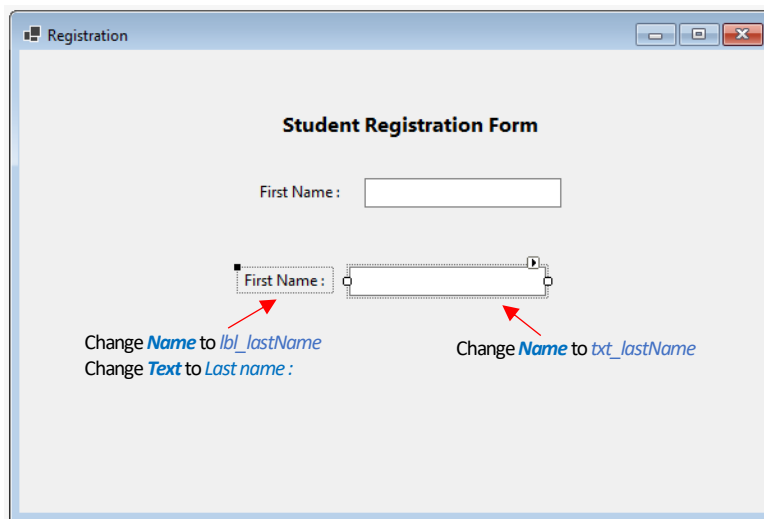


Figure 23. The textbox and label controls pasted.

After properly updating their text and the name values, please select both label and textbox controls, and move them around to align them with the First Name field. You can take advantage of the blue guiding lines as shown in Figure 24 below.

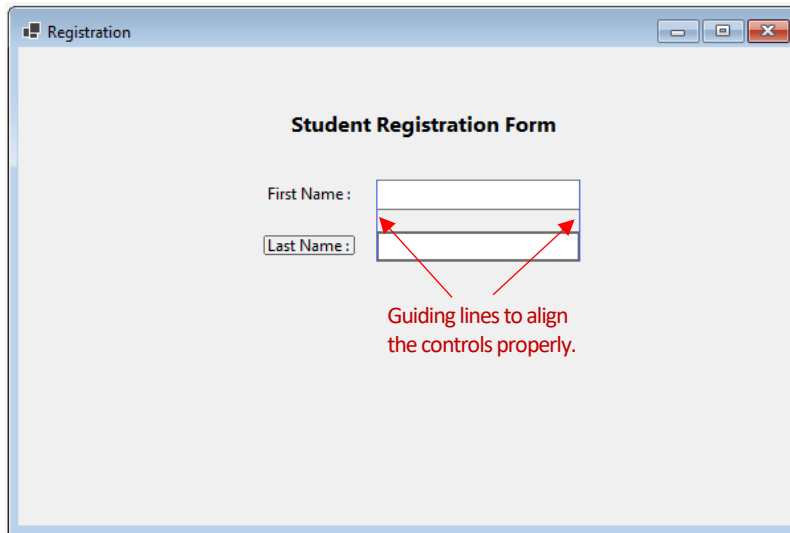


Figure 24. Aligning the last name field with the first name field.

The last field in our form will enable users to enter her/his email. We will follow the same approach as we used before when duplicating the first name field. Please select the label and textbox controls for the last name field and press first **Control + C** (to copy the controls) and then press **Control + V** (to paste) to duplicate the selected controls. Your form should look similar to Figure 25 shown below:

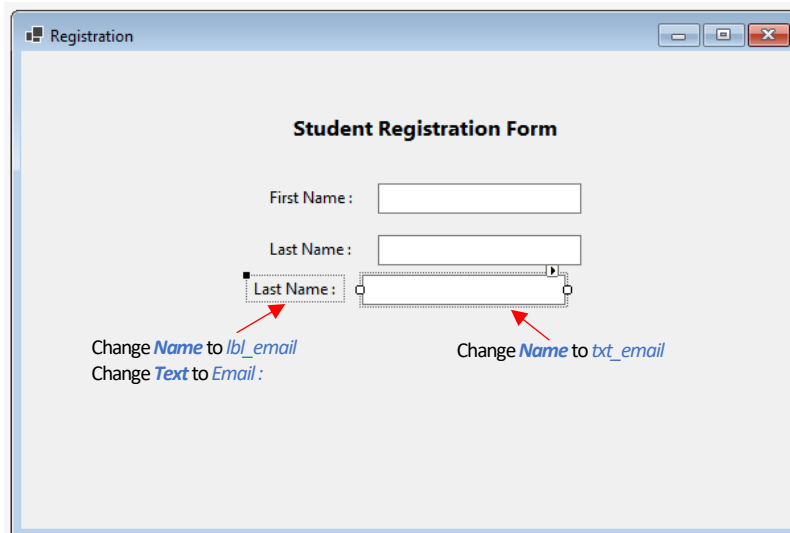


Figure 25. Copying and pasting the last name field.

Please change the **Name** and **Text** properties of the duplicated controls as suggested in Figure 25 and align them with the other fields (using the guiding lines). After these changes, you should obtain a form similar to the one shown in Figure 26.

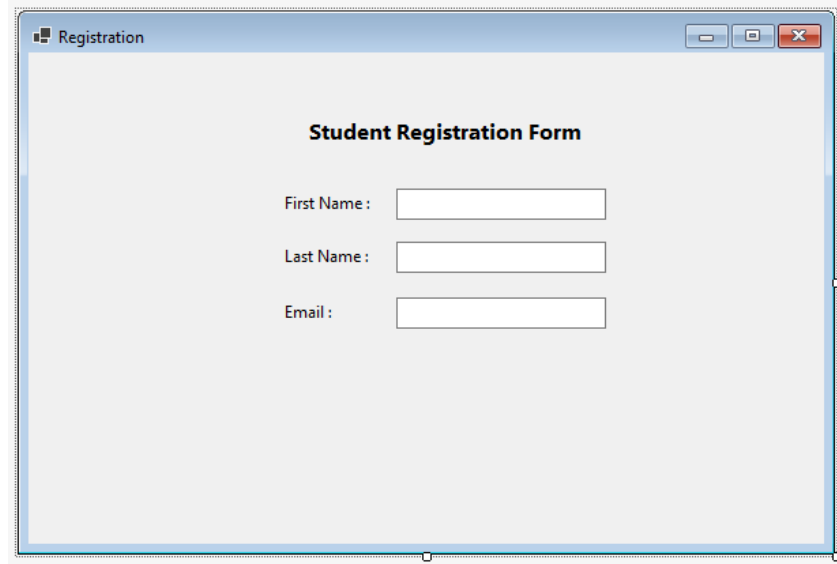


Figure 26. The form with all fields created.

6.3 Adding buttons to the form.

As you may see in Figure 13, the form will have two buttons, “Clear Form” and “Submit”, which will be placed just under the email field. Please drag and drop two buttons onto the form (see Figure 27).

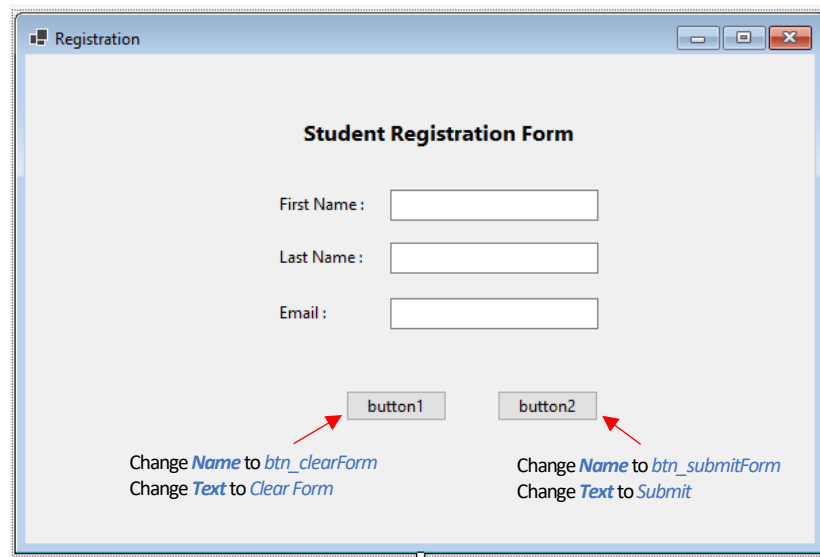


Figure 27. The form with the buttons added.

As we did for all controls so far, we need to update the **Name** and the **Text** properties of the buttons. Using the suggestions in Figure 27, please set a proper name and text for the buttons. After these changes, the form should like Figure28.

Figure 28. The complete form.

We have completed adding all the required controls to our form. Now, you can style some components of the form to make it visually more appealing and user-friendly. The properties that you can use to style a control are grouped under the **Appearance** section in the Toolbox. For example, Figure 29 shows the Appearance section for the button control. Many of these items are common to other controls. Explanations of the commonly used properties are provided in the figure.

Accessibility	
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
Appearance	
BackColor	<input type="checkbox"/> Control
BackgroundImage	<input type="checkbox"/> (none)
BackgroundImageLayout	Tile
Cursor	Default
FlatAppearance	
FlatStyle	Standard
Font	
Font	Segoe UI; 9pt
ForeColor	<input checked="" type="checkbox"/> ControlText
Image	<input type="checkbox"/> (none)
ImageAlign	MiddleCenter
ImageIndex	<input type="checkbox"/> (none)
ImageKey	<input type="checkbox"/> (none)
ImageList	(none)
RightToLeft	No
Text	
Text	Submit
TextAlign	MiddleCenter
TextImageRelation	Overlay
UseMnemonic	True
UseVisualStyleBackColor	True
UseWaitCursor	False

Figure 29. Popular properties to style a button.

The form below shows an example in which buttons are styled differently to visually communicate the user about the different functionality of the buttons.

Figure 30. Styled version of the form.

You can try to use different properties to style the different components of the form as you desire. Just keep in mind that your design should be consistent. That is, each of your textboxes should not have a distinct look as long as you do not have a strong reason for that. The form in Figure 31 is an example of a poor design.

Figure 31. A badly styled form.

7) Using Tab Control to create multipage in a single form

Tab control allows for creating a tabbed view of multiple content holders, called **tab pages**, in the same form. Tab control is located under the *Containers* section in the Toolbox. The transition among these pages is possible through clicking on the tab headers (and also through programmatically, which will cover in next weeks). Each page can have a distinct content consisting of different form controls. You can try adding a Tab control into the form, as shown in Figure 32.

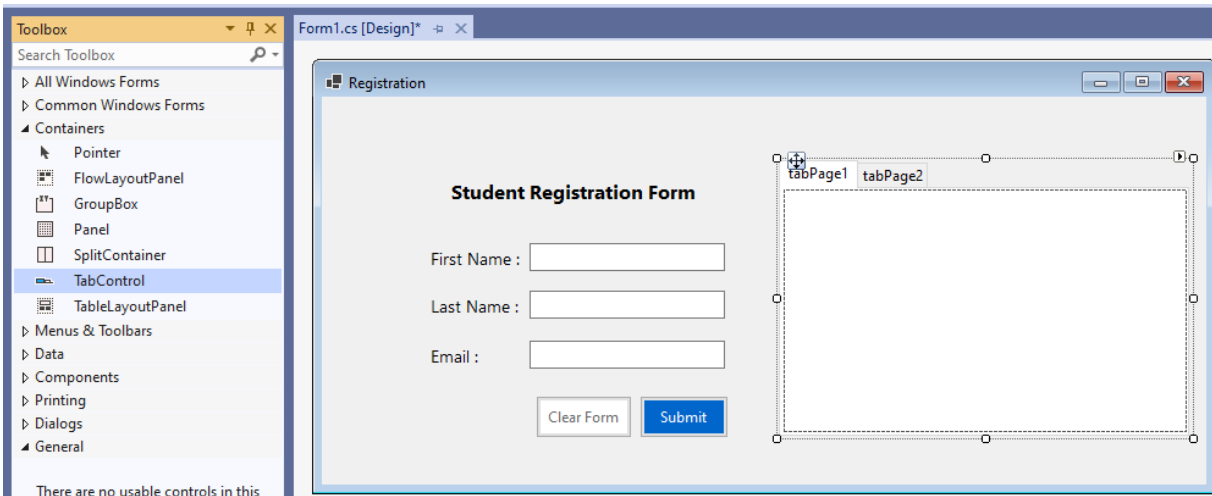


Figure 32. Adding a Tab control to the form.

A newly added Tab control always comes with two tab pages, which have default names as *tabPage1* and *tabPage2*. To add a control inside a tab page, first the page should be selected by clicking on its tab header. The selected page (and its header) will have a white background (which can be changed from the Properties window). For example, in Figure 32, initially *tabPage1* is selected.

Now we will move the existing controls inside the *tabPage1*. To do that, please select them all together using your mouse, do a right-click on them, and then choose Cut from the menu, as shown in Figure 33.

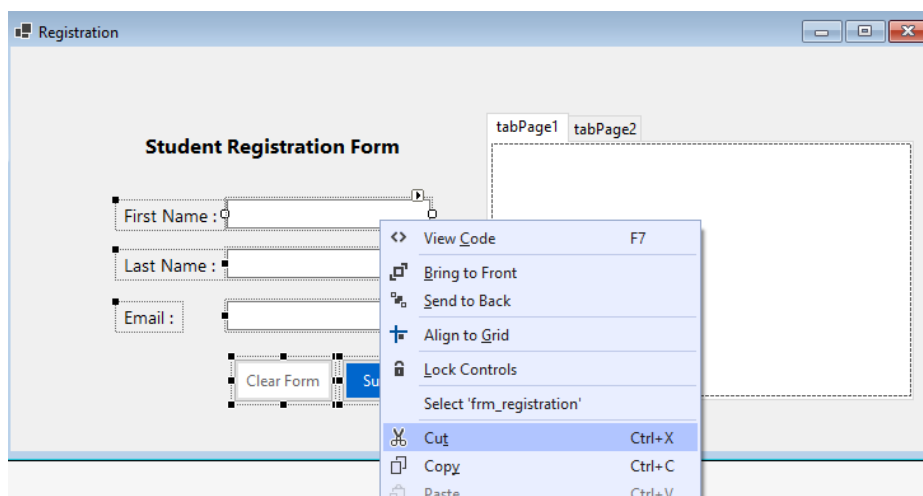


Figure 33. Cutting the controls.

Next, please do a right click inside tabPage1 and then choose Paste from the menu, as shown in Figure 34. All controls should appear inside tabPage1 (see Figure 35).

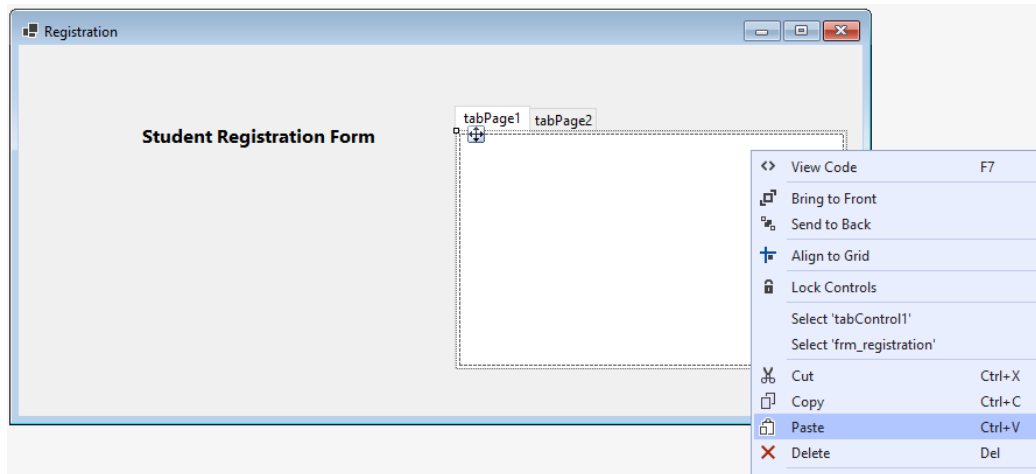


Figure 34. Pasting the controls inside tabPage1.

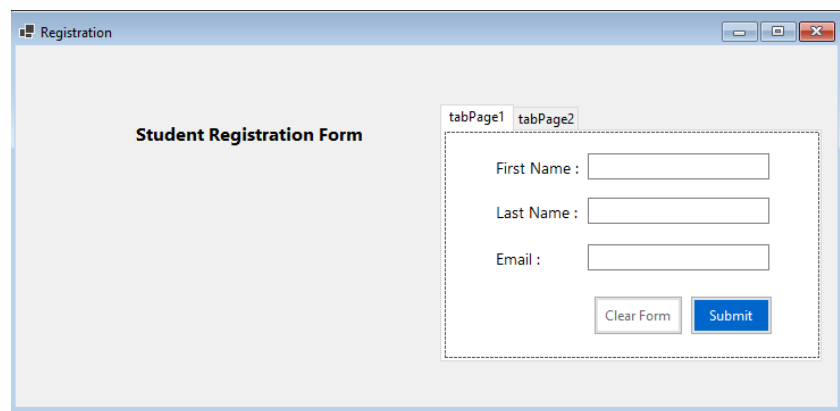


Figure 35. The controls placed inside tabPage1.

From now on, if you move the tab control, all controls inside the tab pages should also move together. Try to arrange the content of the form as shown in Figure 36 below.

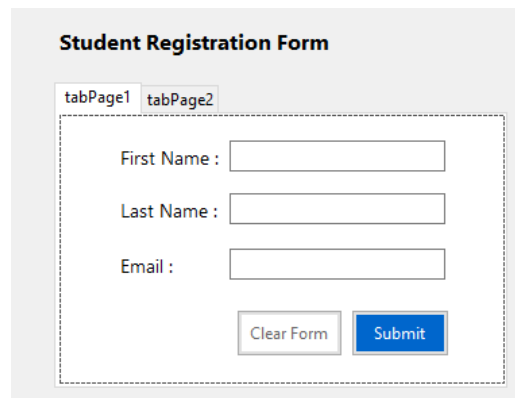


Figure 36. Rearranging the form items.

Now, we will change the titles and names of the tab pages. To change the properties of a tab page (not the entire Tab control), you should select the page by clicking on a white space inside that page. Figure 37 illustrates the Properties window when tabPage1 is selected.

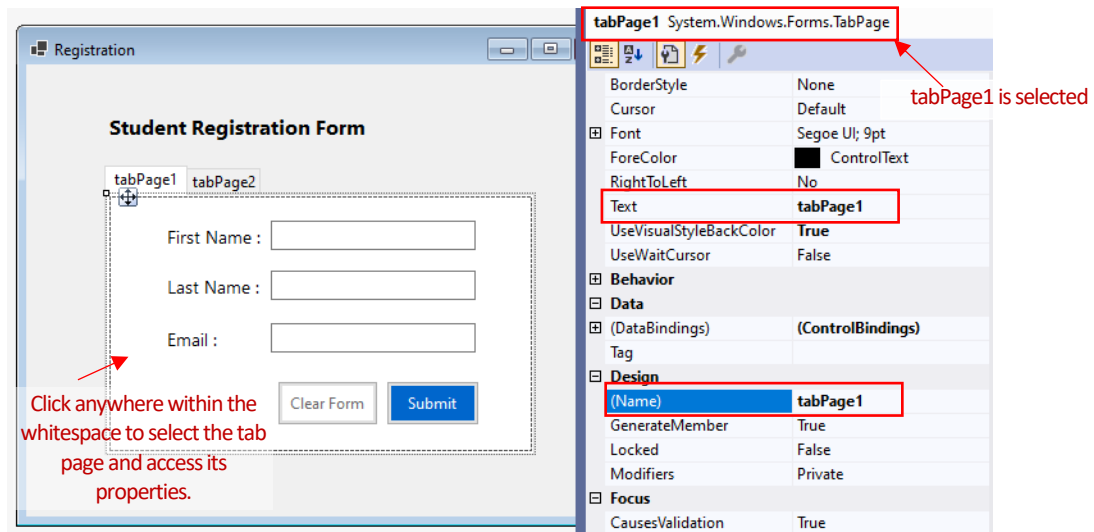


Figure 37. tabPage1 is selected and its properties are shown.

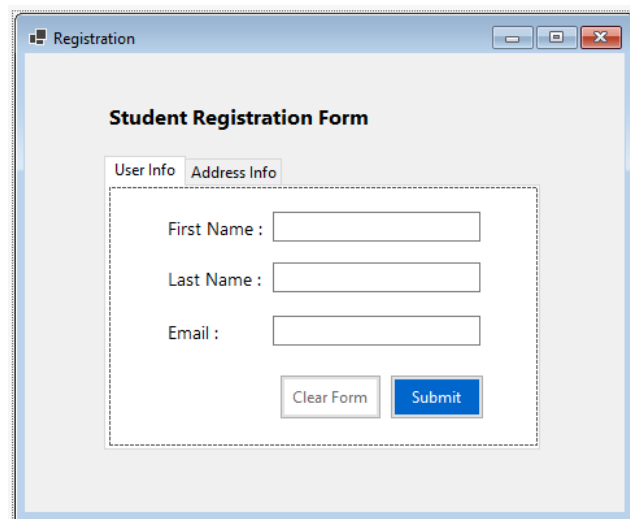


Figure 38. tabPage1 and tabPage2 are updated.

Now, we will build the Address Info tab page by adding a label, a textbox, and a button control as shown in Figure 39. Please provide a proper name for each of these controls. Note that the textbox control should allow for entering multiple lines. This is possible by setting the **Multiline** property of the textbox to True.

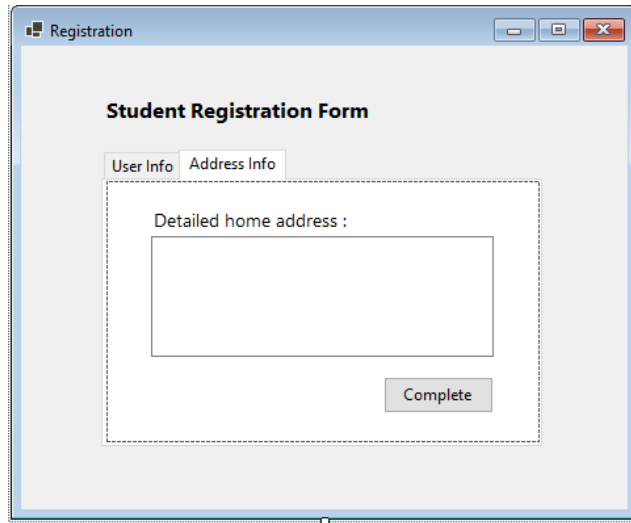


Figure 39. tabPage1 is selected and its properties are shown.

This is the end of the chapter. Please convert your project to a single zip file and upload it to OdtuClass. In the lab assignment, you will work on similar task.