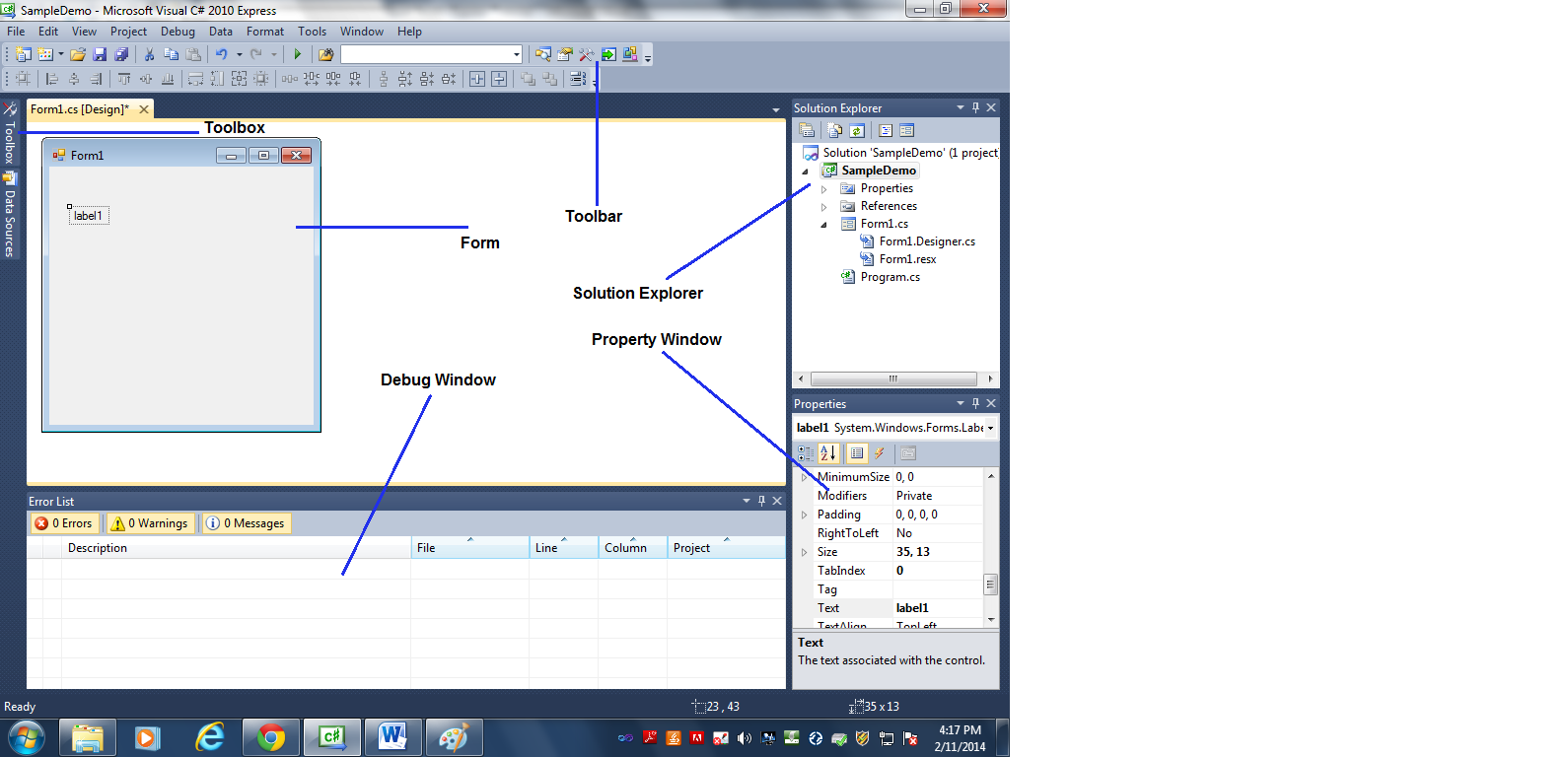
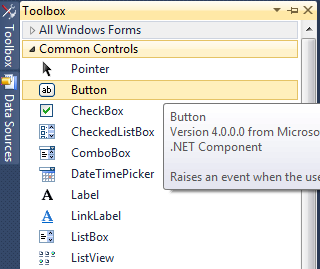
**Visual Studio Express Tutorial**

**Different Parts of your Visual Studio Express**

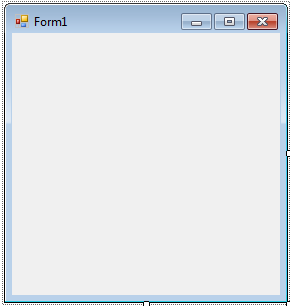
****

**Adding controls to your Form**

**Controls** are any object in your ToolBox that you could add or drop in your Form. Textbox, Label, Button, ListView etc are examples of controls in your ToolBox.

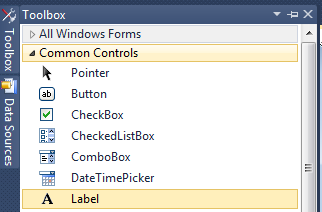


**Forms** on the other hand is the main user interface for your Visual Studio application. You could create as many forms as you like.

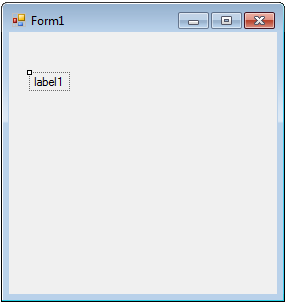


**Example**: Adding Label, Textbox and Buttons to your form

1. Point your mouse to your *ToolBox* and under *Common Controls* point to a *Label*.



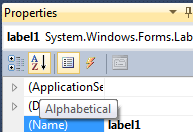
1. Drag and Drop a *Label* to your *Form*



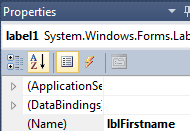
1. Open the *Property* window of your Label if it not open yet by clicking on the property window icon in the Toolbar as shown below:



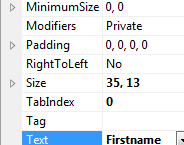
1. Make sure that the Property window Properties are in Alphabetical order if not click on the Alphabetical button (the one that says A to Z) on Property window as shown below.



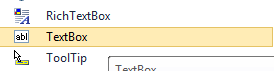
1. Change the (Name) property value from **Label1** to **lblFirstname**.

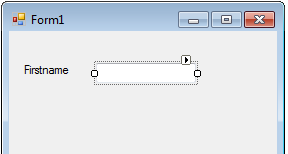


1. Change the *Text* property value from **Label1** to **Firstname**.

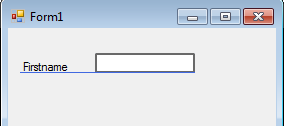


1. Drop a Textbox this time to your Form from your ToolBox as shown below.

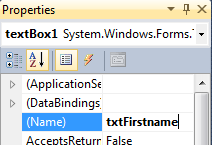




1. Move your textbox until it aligns with your Label (i.e. Firstname). It will show a blue colored line to show its alignment to Label.



1. Change the (Name) property value in Property window of your *Textbox* from **textBox1** to **txtFirstname**.



1. Repeat the process of adding a *Label* and a *Textbox* and this time for the Lastname. Follow the values below.

**Label**

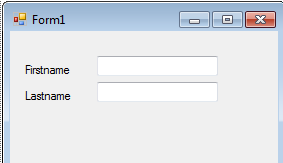
**(Name) :** lblLastname

**Text** : Lastname

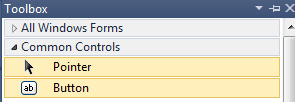
**Textbox**

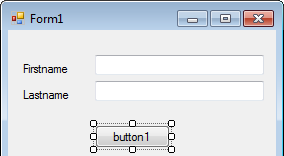
**(Name) :** txtLastname

*Example Output*:



1. Let’s add a *Button* this time. Get a *Button* from a *ToolBox* and drop it again in the form as shown below. (**Note**: Increase the width of your Textbox as shown below to match the sizes of the two buttons).



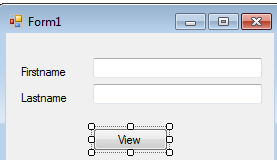


1. Name your Button with the following values:

**Button**

**(Name) :** btnView

**Text** : View

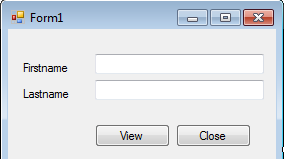


1. Create another *Button* and put it besides *btnView* with the following values:

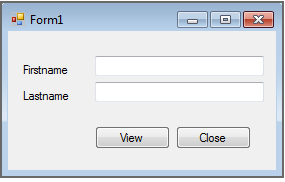
**Button**

**(Name) :** btnClose

**Text** : Close



1. Let’s make our *Form* shorter. Drag from the bottom of the *Form* (it will show a two-headed arrow if you point at the bottom of the Form)until it is near the two *Buttons* that you just created.

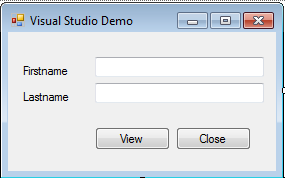


1. Now it’s time to change the property values of your *Form* as below:

**Form**

**(Name) :** frmMain

**Text** : Visual Studio Demo



1. Let’s run your application. Press F5 or Click the *Start Debugging* button at the Toolbar as shown below.



1. Try to click the Close button. And as you probably know by now it does not work. Let’s make it work next but click the X button first to close the *Form*.
2. Double Click on the *Close* Button. And you will have the following Code Behind (I will show you both VB.Net and C#).

**VB.Net**:

Private Sub btnClose\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnClose.Click

End Sub

**C#:**

private void btnClose\_Click(object sender, EventArgs e)

{

}

1. Type the word *Close()* inside the *btnClose\_Click* event (I will explain what is an event later) like shown below:

**VB.Net**:

Private Sub btnClose\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnClose.Click

Close()

End Sub

**C#:**

private void btnClose\_Click(object sender, EventArgs e)

{

Close();

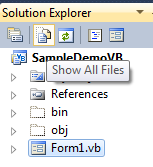
}

1. Run again (F5 or Click the Run Button) your application. (Note: If there is an error check again closely if you follow my instruction).
2. Click the Close Button and see what happens. Did it Close the *Form*?

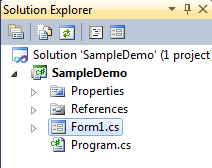
**VB.Net and C# OOP Tutorial**

VB.Net and C# (and your C++ in IS220 class) is actually what is known as Object Oriented Programming (OOP) language. The heart and soul of an OOP language is an object. An object is simply a runtime in-memory instance of a Class. A class on the other hand is a template that the object follows. For example your Button comes actually from Button Class. Go to your Solution Explorer and Click on Show All Files as shown below:

**VB.Net**

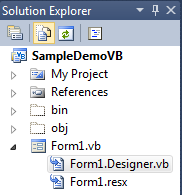


**C#:**

****

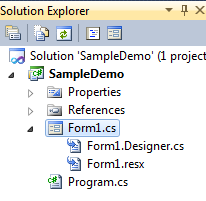
**VB.Net**

Click the arrow Form1.vb (or frmMain.vb if it is the case in yours) and then Double Click on Form1.Designer.vb



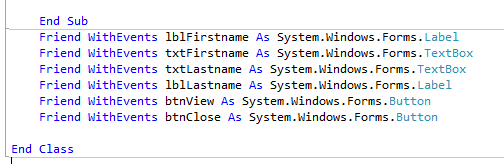
**C#:**

Click the arrow Form1.cs (or frmMain.cs if it is the case in yours) and then Double Click on Form1.Designer.cs

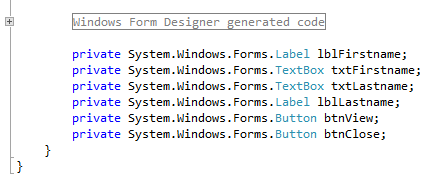


Scroll Down to the Bottom until you see the Lines shown below:

**VB.Net**



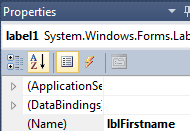
**C#:**

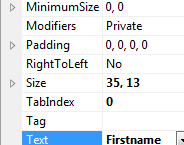
****

Notice your btnClose, btnView comes from **Button** class. The *btnView* and *btnClose* are objects that come from *Button* class. The *System.Windows.Forms* is what is known as **Namespace**. In this case the Button class belongs to *System.Windows.Forms* namespace. Imagine a Namespace like a Folder where your files in your Computer (or Windows for that matter) resides. In this case Class is organized or put inside a Namespace.

**Properties, Methods and Events**

Objects and Class for that matter has what is known as Properties, Methods and Events. Remember in our Property window earlier we changed the name on one of our Label to *lblFirstname* under (Name) property? And then we change the value under *Text* to Firstname? Both (Name) and Text are what is known as properties.





Objects and Class has Events as well. Events are things that happens or an action done towards an object. Remember we double click Close Button and it shows as the following?

**VB.Net**:

Private Sub btnClose\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnClose.Click

Close()

End Sub

**C#:**

private void btnClose\_Click(object sender, EventArgs e)

{

Close();

}

The *Click* is the event that the Button class or in this case actually the btnClose receives. So, what we did is every time the Button is Click when our program is run – we close the Form in this case using Close(). And by the way the Close() is a **method**. A method on a class or object is simply like your Function in your C++ class that is a line of codes put together that do one/single function or task.

**Handling more events in your program and Adding a Method**

We’ll add an event in your program that you have just created and then we will add also a method along with that. But first let’s save our work by Clicking on Save All as shown below and name it as **SampleDemo**:

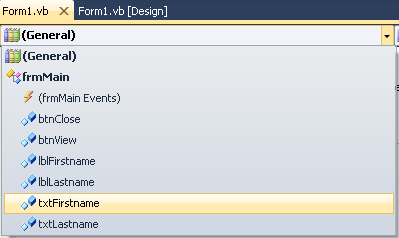


What we are going to do now is that we are going to add event name *KeyUp*. A KeyUp event is when you press any key in your keyboard and then release (or up).

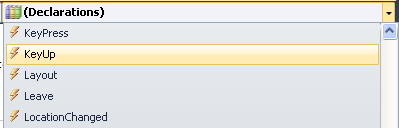
1. Right Click in your Form and select *View Code* or you could simply press F7.
2. We then create event as shown below:

**VB.Net**

* On the Drop down list (on the Left) above Click *General* and select *txtFirstname* as shown below:



* Then on the right Drop down list under *Declarations* scroll down and select KeyUp event as shown below:



* You will have the following Code behind:



**C#:**

* Click inside frmMain() constructor and place your cursor after InitializeComponent();
* Press Enter and Type the following:

txtFirstname.KeyUp += new KeyEventHandler(txtFirstname\_KeyUp);

* Copy paste the code below after frmMain() {} but before btnClose\_Click() {} :

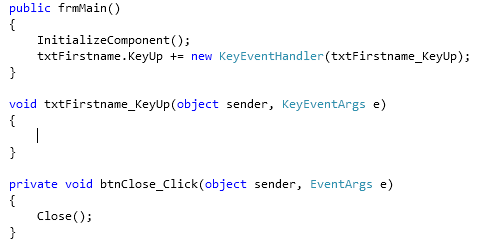
void txtFirstname\_KeyUp(object sender, KeyEventArgs e)

{

throw new NotImplementedException();

}

* Delete the line throw new NotImplementedException(); and it should like as shown below:



1. We are now ready to add some more code. What we are going to do here is that when a user Press and enter key we are going to trap it and do something as you will see next.

**VB.Net:**

If (e.KeyCode = Keys.Enter) Then

End If

**C#:**

if (e.KeyCode == Keys.Enter)

{

}

1. So what we are going to do now after we know that the user press an Enter Key we are move the cursor to the next Textbox, namely the Textbox for the *Lastname* which is *txtLastname.*

**VB.Net:**

If (e.KeyCode = Keys.Enter) Then

txtLastname.Focus()

End If

**C#:**

if (e.KeyCode == Keys.Enter)

{

txtLastname.Focus();

}

1. Repeat steps 2 to 4 and this time instead of *txtFirstname* it would be for *txtLastname*. And instead of txtLastname.Focus() it would be *btnView.Focus()* as shown in the example code below:

**VB.Net:**

If (e.KeyCode = Keys.Enter) Then

btnView.Focus()

End If

**C#:**

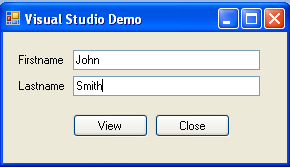
if (e.KeyCode == Keys.Enter)

{

btnView.Focus();

}

1. Run your application (F5) again and enter information and see what happen when you presses enter key(s).

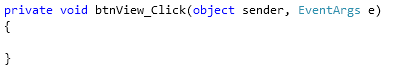


1. Now, we are going to add some code on the *View* button. Double Click on the View Button and it will show the code below:

**VB.Net:**



**C#:**

****

1. What are going to do next is that when the user clicks the btnView we are going check if the txtFirstname and txtLastname is not empty.

**VB.Net:**

If (Not String.IsNullOrEmpty(txtFirstname.Text) And Not String.IsNullOrEmpty(txtLastname.Text)) Then

End If

**C#:**

if (!String.IsNullOrEmpty(txtFirstname.Text) && !String.IsNullOrEmpty(txtLastname.Text))

{

}

1. If the Lastname and Firstname is not empty then we are going to display the name on a Window using *MessageBox.Show()* as shown below:

**VB.Net:**

If (Not String.IsNullOrEmpty(txtFirstname.Text) And Not String.IsNullOrEmpty(txtLastname.Text)) Then

Dim Fullname As String

Fullname = txtFirstname.Text & " " & txtLastname.Text

MessageBox.Show("Hello " & Fullname, "Greetings", MessageBoxButtons.OK, MessageBoxIcon.Information)

End If

**C#:**

if (!String.IsNullOrEmpty(txtFirstname.Text) && !String.IsNullOrEmpty(txtLastname.Text))

{

String Fullname;

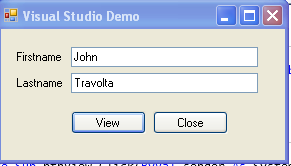
Fullname = txtFirstname.Text + " " + txtLastname.Text;

MessageBox.Show("Hello " +

Fullname, "Greetings", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

1. Run your application type whatever name you want and then Click the View button and it should show like one below:





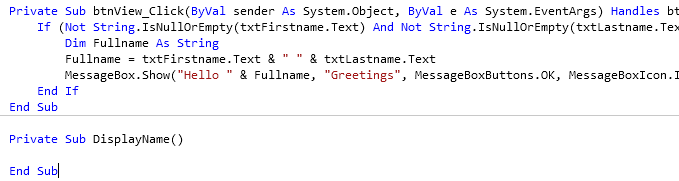
1. We’ll add a Method this time. We are going to create a method name *DisplayName()*.

**VB.Net:**

Private Sub DisplayName()

End Sub

It should be after the End Sub of *btnView\_Click* as shown below:



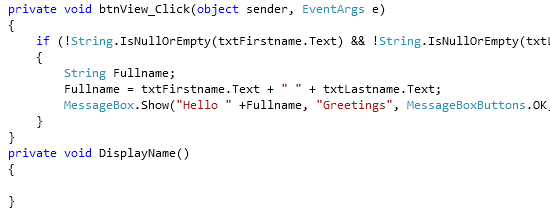
**C#:**

private void DisplayName()

{

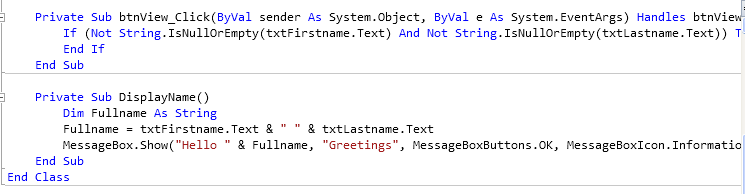
}

It should be after the closing brace } of *btnView\_Click* as shown below:

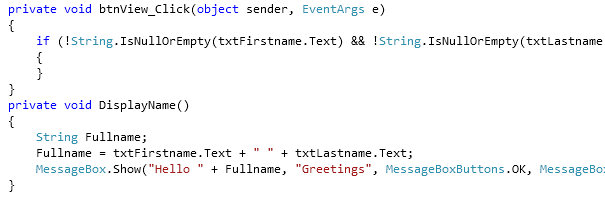


1. Cut all the code inside the If condition and paste it inside the DisplayName() method as shown below:

**VB.Net**

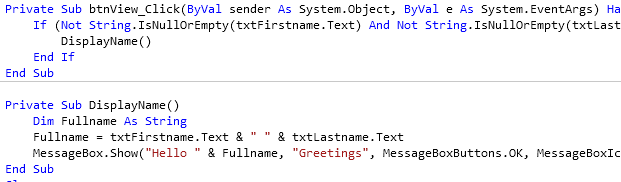


**C#:**

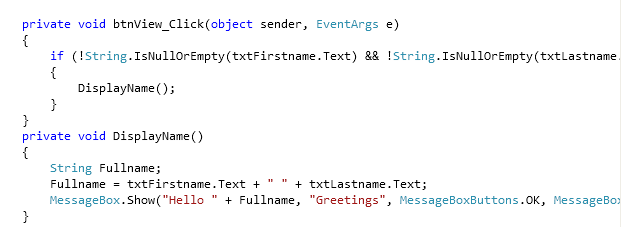


1. Now type the word DisplayName() inside the if the condition. This one will call the method that we have just created named DisplayName().

**VB.Net**



**C#:**



1. Run again your application and you will have the same result as before. However, as you see we make our program more manageable because we create a Method which is its sole purpose or task is to display the entered name and that in essence is the use of a method in a class or an object.