Lab 4 – User Interface Design Using VB.NET

Human Computer Interfacing

(BESE 14)

Objectives

This is the third lab activity on designing a user interface using VB.net. In the previous labs we have seen how to work with:

Labels, Text boxes and Buttons, Menus, Picture Box, Combo Box, Check Boxes, Radio Buttons and Calendar.

In today's lab, you will learn:

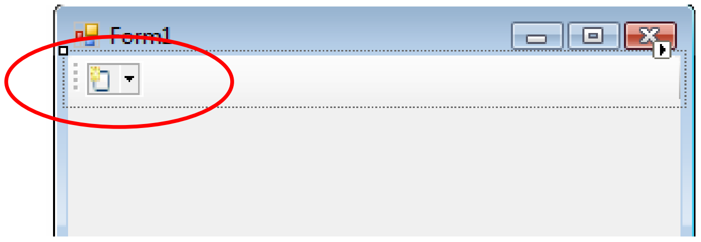
- Toolbars
- Creating multiple Forms
- Modal and Non Modal Forms
- Passing Values between Forms

Task 1

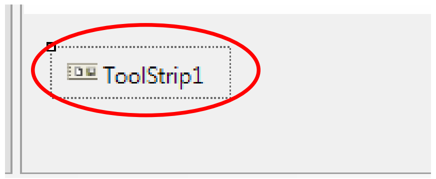
Note: This task comprises practice exercises and you do NOT need to submit any lab report/program code.

Adding a Toolbar

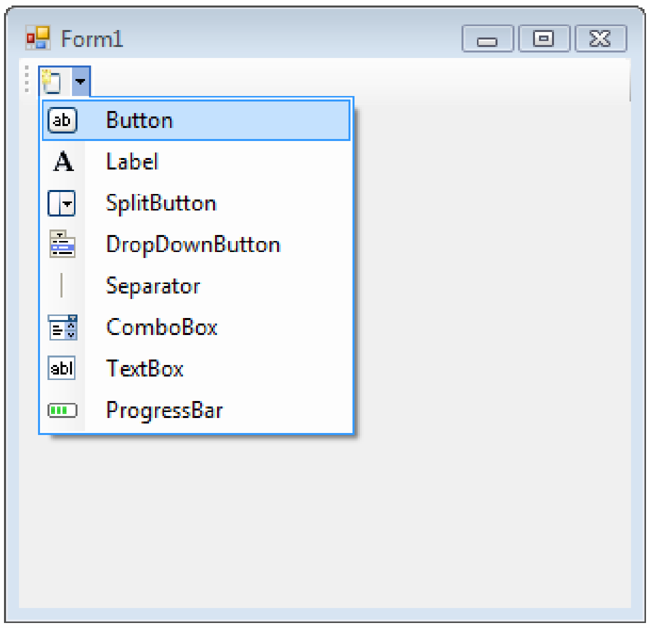
- Start a new project, expand the Toolbox and locate the ToolStrip control and add it to your form by double clicking it.
- On the top of the Form you will see some thing like this :



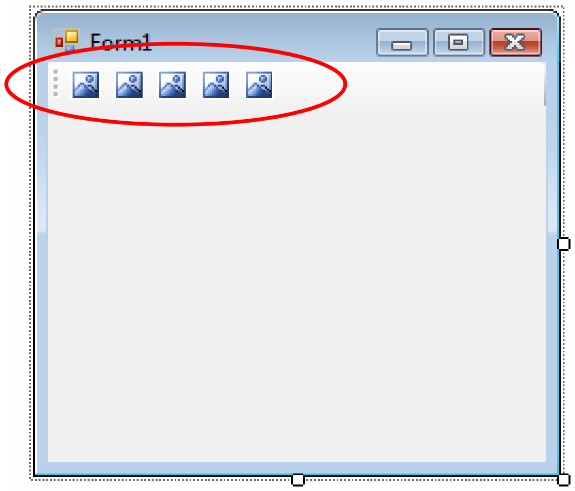
- While on the bottom you will have the ToolStrip control:



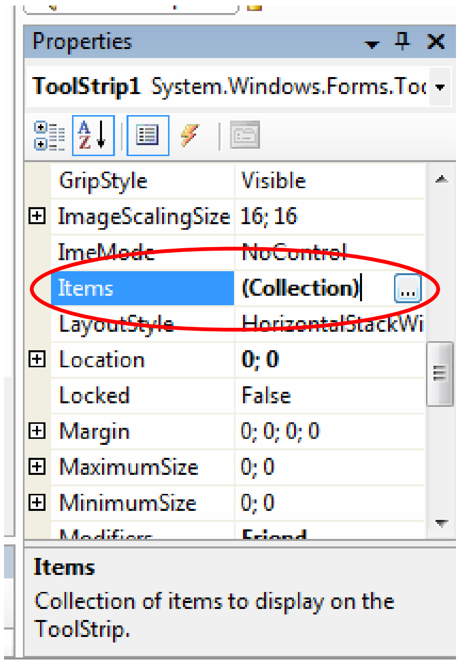
- Clicking on the top strip, add few buttons to the toolbar:



- Your form should look like:



- Select the ToolStrip control from the bottom, inspect the properties dialog box and find the properties 'Items':

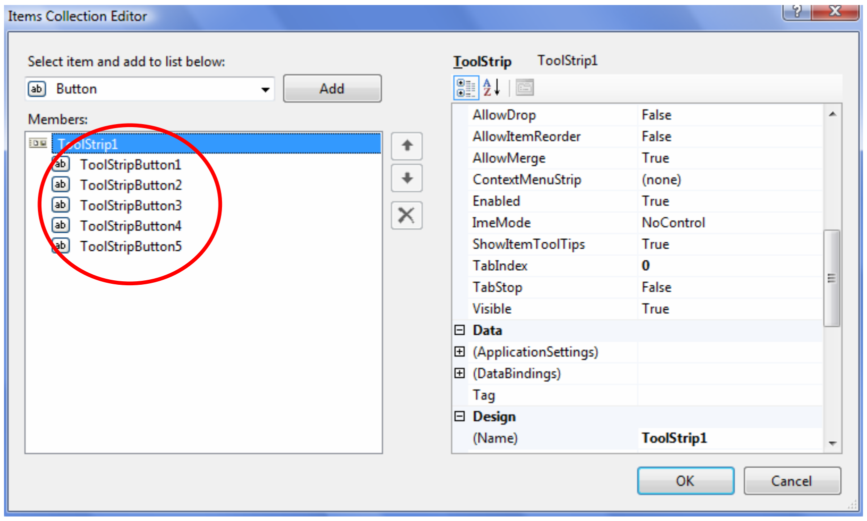

Properties

ToolStrip1 System.Windows.Forms.Toc

GripStyle	Visible
ImageScalingSize	16; 16
ImeMode	NoControl
Items	(Collection)
LayoutStyle	HorizontalStackWi
Location	0; 0
Locked	False
Margin	0; 0; 0; 0
MaximumSize	0; 0
MinimumSize	0; 0
Modifiers	Friend

Items
Collection of items to display on the ToolStrip.

- Expand the 'Items' Property and you will have a dialog box like this:


Items Collection Editor

Select item and add to list below:
Button — Add

Members:
ToolStrip1
ToolStripButton1
ToolStripButton2
ToolStripButton3
ToolStripButton4
ToolStripButton5

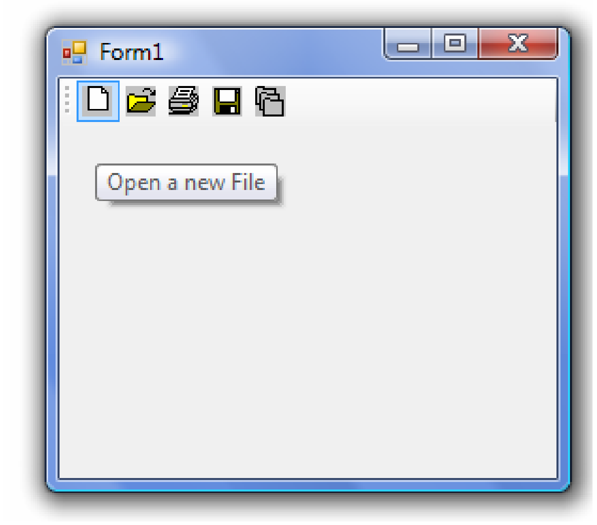
ToolStrip ToolStrip1

AllowDrop	False
AllowItemReorder	False
AllowMerge	True
ContextMenuStrip	(none)
Enabled	True
ImeMode	NoControl
ShowItemToolTips	True
TabIndex	0
TabStop	False
Visible	True
Data	
(ApplicationSettings)	
(DataBindings)	
Tag	
Design	
(Name)	ToolStrip1

OK Cancel

- The buttons added to the Toolbar are visible in the Dialog box. We want our toolbar to contain the following buttons:
    - File Open

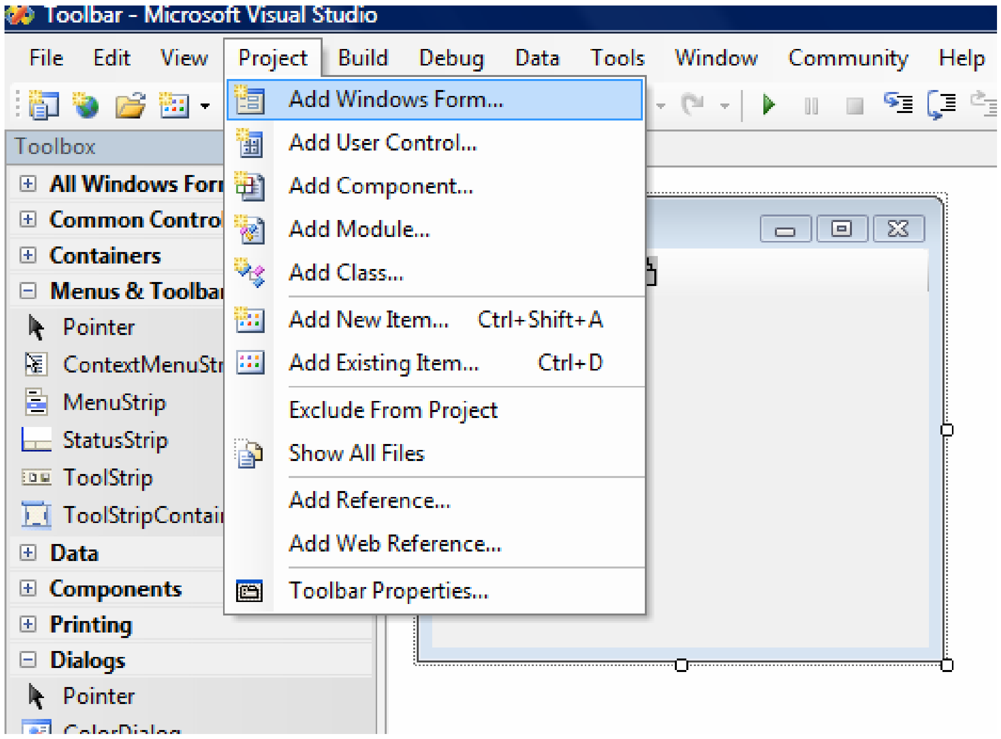
  - New File
  - Save
  - Copy
  - Print
- Select each of these buttons one by one and change the following properties:
  - Name: Give a meaningful name to each button.
  - ToolTip Text: A small text message that would appear on the button once you move mouse over it (a help message).
  - Image: Assign an image (from the images provided) to each button.
- Performing these steps, your Form will look something like:



- If you move your mouse on the buttons (in the Run mode) you will see the message that you added to the ToolTip Text property.
- Add a simple message box to each of these buttons and verify the working of your program.

Creating Multiple Forms

- It's a rare programme that only has one ‘Form’ in it. Most programmes will have multiple forms.
- To add another form to your existing project, Click on the ‘Project’ Menu and add a new Form ‘Add Windows Form’



- We want the Form2 to be displayed when the user clicks any of the buttons on Form1.
- To any of the buttons on the first Form, add the following functionality:

```
Dim secondForm As New Form2
secondForm.Show()
```

- We have created a new object of the class Form2 and called its 'Show' method.
- Run your programme and test it out. When you click your button, you should see the second form appear.
- However, there's a problem with this code. Click the button again and another copy of **Form2** appears. Keep clicking the button and your screen will be filled with the second form!
- To prevent this from happening, you can move the code that creates the form object. Move it right to the top, just below where it says **Public Class Form1** – (outside of the button, in other words).
- The only code left in the button is the line that Shows the form. A new form object will now not be created every time the button is clicked. If you try it out,

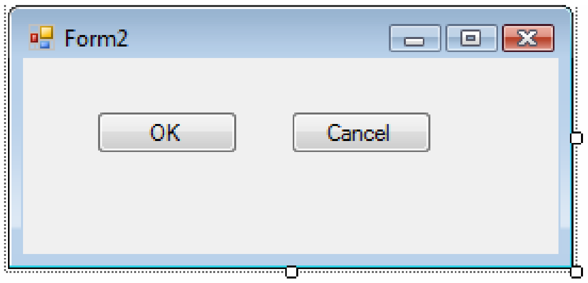
you should see only one form appear when the button is clicked, and not multiple forms.

Modal and Non Modal Forms

- A modal from is one that has to be dealt with before a user can continue. An example is the Format->Paragraph dialogue box in Microsoft Word. If you try to click away from the dialogue box, you'll here a beep to indicate an error. Until you click either the Cancel or OK buttons, the programme won't let you click anywhere else.
- The second form you've just created is called a **Modeless** form. These are forms than can be sent to the taskbar. You can then return to the main form or programme and do things with it.
- A Modal form is sometimes called a dialogue box. And we'll see how to create one of these now.
- To a second button in your Form1 add the following code:

```
Dim secondForm As New Form2
secondForm.ShowDialog()
```

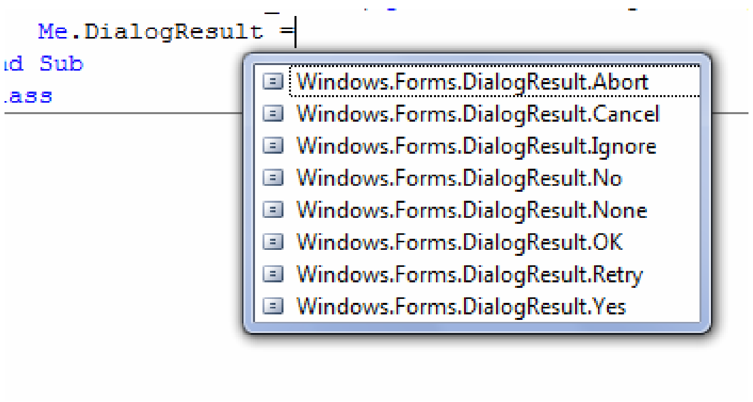
- To display a form as a Modal dialogue box, you use the ShowDialog method. If you use the Show method, the form is displayed as a Modeless form
- Run your programme and Click the button to display the second Form. Move it out the way and try to click a button on Form1. You won't be able to. The second form has to be dealt with before you can access Form1.
- Now add two buttons to the second Form to have something like this :



- Add the following functionality to the OK button

```
Me.DialogResult = Windows.Forms.DialogResult.OK
```

- The **Me** keyword refers to the current form. When you type 'Me.DialogResult =' you will see a popup list appear :



```
Me.DialogResult =
nd Sub
ass
```

Windows.Forms.DialogResult.Abort
Windows.Forms.DialogResult.Cancel
Windows.Forms.DialogResult.Ignore
Windows.Forms.DialogResult.No
Windows.Forms.DialogResult.None
Windows.Forms.DialogResult.OK
Windows.Forms.DialogResult.Retry
Windows.Forms.DialogResult.Yes

- The value **Windows.Forms.DialogResult.OK** indicates that you want to use this button as an OK button. When the button is clicked, VB.NET will return a result of OK for this button.
- Similarly, the Cancel button add the following code:

```
Me.DialogResult = Windows.Forms.DialogResult.Cancel
```

- When the Cancel button is clicked, VB.NET will return a result of Cancel for this button.
- Now access your Form1 code, and locate the lines that display the second form. The two lines should be these:

```
Dim secondForm As New Form2
secondForm.ShowDialog()
```

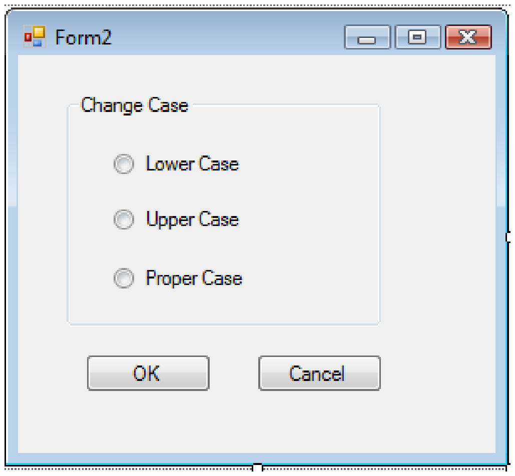
- Now remove the second line and add the following code:

```
Dim secondForm As New Form2
Dim result As Integer
result = secondForm.ShowDialog()

If result = Windows.Forms.DialogResult.OK Then
    MsgBox("OK button clicked in the second form")
ElseIf result = Windows.Forms.DialogResult.Cancel Then
    MsgBox("Cancel button clicked in the second form")

End If
```

- Run and test your code.

Getting Values from other forms

- The form with OK and Cancel buttons on is not doing much work so let's turn the form into a Change Case dialogue box.
- Change the design of your second form to look like the following :



- When you've designed your form, click back on Form1 and add a Textbox to it. When a button on Form1 is clicked, the dialogue box above will be displayed. You can then select an option button to change the case to Upper, Lower or Proper case. This will happen when the OK button is clicked. Whatever text is in Texbox1 on Form1 will be changed accordingly.
- Double click the OK button on **Form2** to access the code. You should have the following:

```
Me.DialogResult = Windows.Forms.DialogResult.OK
```

- If you want to refer to Texbox1 on Form1, one way could be :

Form1.Textbox1.Text

- In some version of VB, that code would be all right. You're saying "Access the Text property of Textbox1 on Form1." However, a better option to achieve the same has been presented in the following.

- Create a textbox object variable on Form1, and assign Textbox1 to this variable. But this variable has to be something that all Classes in the project can see. So add the following to the top of your code window for Form1

```
Public tb As TextBox
```

- We're setting up a variable which we've called **tb**. A Textbox object is going to be stored in this variable. But notice that the variable is **Public**. This way, **Form2** will be able to see the variable.
- Double click anywhere on Form1. This will open the Form1_Load function that is called immediately once you execute the program and the Form is loaded. Add the following line:

```
tb = TextBox1
```

- When Form1 loads, the textbox called Textbox1 will be assigned to the tb variable. Now Textbox1 can be seen by Form2.
- Go back to your code for the OK button on Form2 and add the following at the top:

```
 Dim ChangeCase As String

ChangeCase = Form1.tb.Text
```

- We're setting up a String variable called ChangeCase. Whatever text is in Textbox1 of Form1 will then be assigned to the ChangeCase variable.
- We now only need to add the code that does the actual converting. So your code should look some thing like :

```
Dim ChangeCase As String
ChangeCase = Form1.tb.Text

If RadioButton1.Checked Then
    ChangeCase = ChangeCase.ToLower
ElseIf RadioButton2.Checked Then
    ChangeCase = ChangeCase.ToUpper
ElseIf RadioButton3.Checked Then
    ChangeCase = StrConv(ChangeCase, VbStrConv.ProperCase)
End If

Form1.tb.Text = ChangeCase

Me.DialogResult = Windows.Forms.DialogResult.OK
```

- When you're finished adding the code, run your programme. Enter some text into Textbox1. Then click the button that brings up the Change Case Dialogue box. Select an option from the three available, and the click OK. The text in Textbox1 should be converted.

Task 2

This task will be considered as an **Assignment.** Start it in the lab session and submit it **Before the start of next lab.**

- You have to design a flight search interface. An example interface could be:

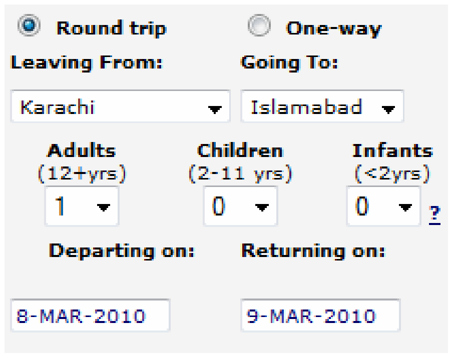
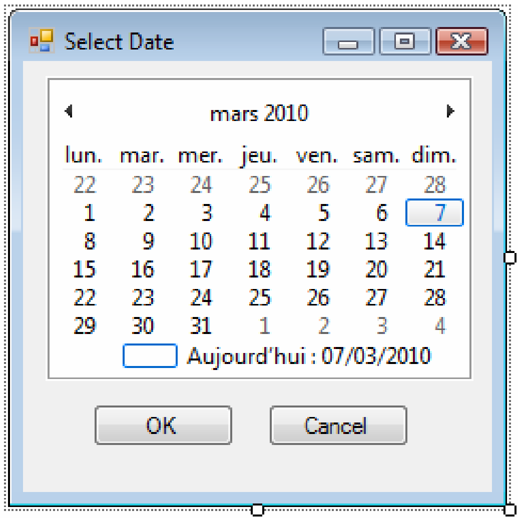


Figure 1 An example flight search interface (Source: Airblue.com)

- Note that the interface presented is just an example, you are free to add your personal preferences of colors, fonts, layouts etc.
- With each of the TextBoxes for entering the date, provide a small button.
- Clicking on the button, another form pops up which contains a Calendar and two buttons:



- The user selects a date and when clicks the OK button, the date is displayed in the respective text box.
- Finally, on the Main form, display the summary of user travel options on clicking a given button.
- Important: For this particular exercise, you can use Two different forms with Calendars. Each button for date selection invokes its respective form. We will learn how to do it with one single calendar from in a later lab activity.