3. Introduction to windows control

Control is an Object. In the aspect of VB.Net, Control is a User Interface Element which is used to place in the VB.Net Form to perform some action. To design the user interface, you can add controls to the form **at design time or at runtime. Control class** is the base class of all the windows controls**.** All controls are unique with through its features.

We can add controls from the toolbox **at design time** as given below:

1. Dragging and Dropping
2. Double click on the control

We can add controls **at run time** as given steps below:

1. Create an object of control
2. Set the properties of it.
3. Add that controls to the form using Me.controls.Add(ctlobj).

Every control has three features in VB.Net.

1. **Properties** describe the object.
2. **Methods** are used to make the object do something
3. **Events** describe what happens when the user/Object takes any action**.**

Some Common Properties of the controls are as given below:

Properties	**Description**
Background Color	It is used to change the back color of the control.
Fore Color	It is used to change the fore color of the Control.
Visible	It is used to enable or disable the visibility of the controls. It has Boolean value either True or False. By default it is True.
Enabled	This Property is similar to Visible property but the difference is that, In this property the components are visible on both True and False condition, but it not perform function on False condition.
Text	Name (Text) of the control (i.e. Name which are display on the form) we can change by using this property.
Dock	By using this property we can arrange the controls (i.e. Top, Bottom, Left, Right, and Fill) in a proper manner on the form.
Size	It is used to set the size of the controls.
(Name) :	This property is like a Text property. This property is used to set the name of the control (i.e. the name which is used to writing a code on control).
Font	It is used to set the font style of the name of the control.
Image	It is used to set the image of the controls.
Tab Stop	It is declared in the form of Boolean i.e. in True or False value. This allows the control focus on pressing Tab Key in Key Board.
Tab Index	It is used to declare the index or the number on which control focus by pressing Tab Key.

Working with common controls

3.1.1. Label

- It is used to display text on the form which user can not edit.
- However user can edit contents of the label control **at run time using text property**.
- The Prefix for the Label is “lbl”.

Properties:

Property	Description
BackColor	It is used to get or set background color of the label.
Font	It is used to set Font Face, Font Style, Font Size and Effects of the text associated with Label Control.
ForeColor	It is used to get or set Forecolor of the text associated with Label Control.
Enabled	It is used to specify whether label control is enabled or not at run time. It has Boolean value. Default value is true.
Image	It is used to specify an image that is displayed in Label Control.
ImageAlign	It is used to get or set alignment of the image that is displayed in the Label control.
Text	It is used to get or set text associated with the Label control.
TextAlign	It is used to get or set alignment of the text associated with the Label control.
Visible	It is used to specify whether label control is visible or not at run time. It has Boolean value. Default value is true.

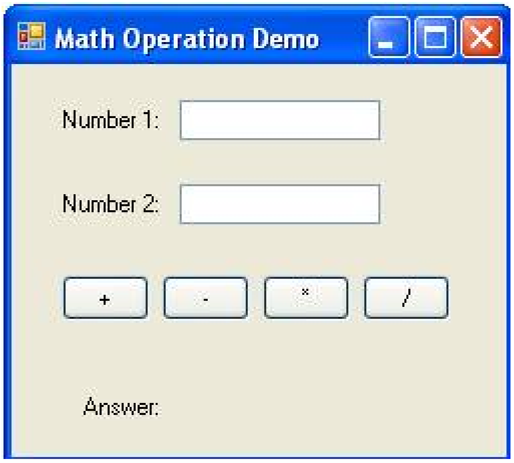
Methods:

Method	Description
Show	It is used to show label control at run time.
Hide	It is used to hide label control at run time.

Events:

Event	Description
Click	It is the default event of Label Control. It fires each time user clicks on Label Control.
DoubleClick	It fires each time user double clicks on Label Control.
TextChanged	It fires each a text associated with Label Control is changed.

Example: Program that accepts two numbers from user and performs addition, subtraction, multiplication and division of two numbers using Label, Textbox and Button Control.



Source Code:

```
Public Class Form4

    Private Sub btnadd_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnadd.Click
        Try
            MsgBox(txtno1.Text + txtno2.Text)
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

    Private Sub btnsub_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnsub.Click
        Try
            MsgBox(txtno1.Text - txtno2.Text)
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

    Private Sub btnmulti_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnmulti.Click
        Try
            MsgBox(txtno1.Text * txtno2.Text)
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

    Private Sub btndiv_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btndiv.Click
        Try
            MsgBox(txtno1.Text / txtno2.Text)
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub
End Class
```

3.1.2. Textbox

- It is used to accept textual input from the user. The user can add strings, numerical values and a combination of those, but Images and other multimedia content are not supported.
- By default, it takes a single line of text, however, you can make it accept multiple texts and even add scroll bars to it.The prefix of TextBox is **"txt"**

Properties:

Property	Description
Name	Indicates the name used in code to indentify the object. **Ex. txtname,txtaddress**
Text	Gets or sets the current text in the TextBox.
PasswordChar	Indicates the character as a password.We can enter *? Etc character for password
TextAlign	It is used for setting text alignment such as Left,Right,Center
ScrollBars	For adding scrollbars. ThisProperty has values • None • Horizontal • Vertical • Both
Multiline	It is used to set the TextBox Control to allow multiple lines
MaxLength-	It is used for specifying the maximum character number the TextBox Control will accept.
Enabled	It is used to specify whether textbox control is enabled or not at run

	time. It has Boolean value. Default value is true.
Readonly	Gets or sets a value indicating whether text in the text box is read-only. if set to **true**, you will be able to use the TextBox Control, if set to **false**, you won't be able to use the TextBox Control.
WordWrap	Indicates whether a multiline text box control automatically wraps words to the beginning of the next line when necessary.
TabIndex	Gets or sets the tab order of the control within its container.

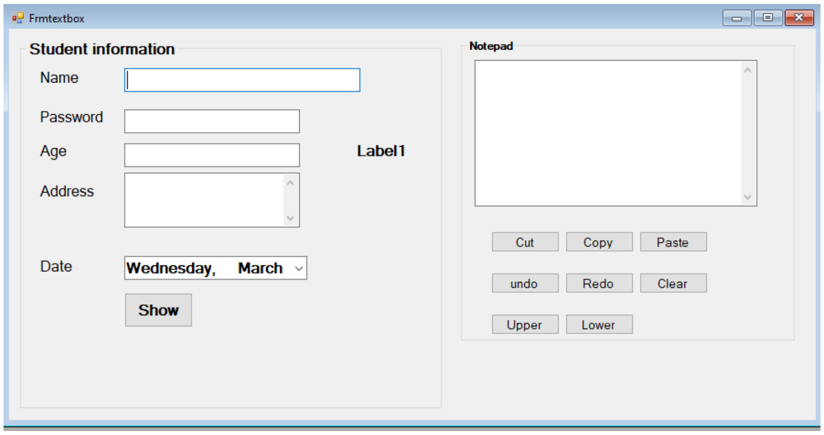
Methods:

Method	**Description**
Clear	Clears all text from the text box control.
Cut	Moves the current selection in the text box to the Clipboard.
Copy	Copies the current selection in the text box to the Clipboard.
Paste	Replaces the current selection in the text box with the contents of the Clipboard.
Undo	Undoes the last edit operation in the text box.
Focus	Sets focus to the TextBox.
ToString()	Returns a string that represents the TextBox control.

Events:

Event	**Description**
TextChanged	It is the default event of TextBox Control. It fires each time a text in the TextBox control is changed
GotFocus	It fires each time a focus is set on TextBox
LostFocus	It fires each time a focus is lost from TextBox

Example:



Source Code:

```
Public Class Frmtextbox

    Private Sub btncut_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btncut.Click
        Try
            txtdata.Cut()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try

    End Sub

    Private Sub btncopy_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btncopy.Click
        Try
            txtdata.Copy()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try

    End Sub

    Private Sub bntpaste_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles bntpaste.Click
        Try
            txtdata.Paste()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try

    End Sub

    Private Sub btnundo_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnundo.Click
        Try
            txtdata.Undo()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

    Private Sub btnredo_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnredo.Click
        Try
            If txtdata.CanUndo = True Then
                txtdata.Undo()
                txtdata.ClearUndo()
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Critical, "Error")
        End Try

    End Sub

    Private Sub btnclear_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnclear.Click
        Try
            txtdata.Clear()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

    Private Sub btnupper_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnupper.Click
        Try
            txtdata.Text = UCase(txtdata.Text)
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

    Private Sub btnlower_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnlower.Click
        Try
            txtdata.Text = LCase(txtdata.Text)
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try

    End Sub

    Private Sub btnshow_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnshow.Click
        Try
            MsgBox("Your Name is" & txtname.Text & vbCrLf & "you are " & txtage.Text
& " year Old" & vbCrLf & "your are from" & txtadd.Text)
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try

    End Sub
    Private Sub txtname_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles txtname.TextChanged
        MsgBox(txtname.Text)
    End Sub

End Class
```

3.1.3. Button

- Button is a widely used control in application. It is used to perform an action.
- Whenever user clicks on a button the click event associated with the button is fired and the action associated with the event is executed.
- The prefix for button control is **"btn".**
- We can set **accelerator** key for the button. It is also known as **mnemonics**. It is represented by underlined characters.
- For example set **text property** of the btnans(button) as **&Ans**. It will display underline under the character A now we call btnans button's event by **ALT +A.**

Properties:

Property	**Description**
Name	Indicates the name used in code to indentify the object. **Ex. btnans,btnshow.**
Text	Gets or sets the current text in the Button.
TextAlign	It is used for setting text alignment such as Left,Right,Center
Enabled	It is used to specify whether textbox control is enabled or not at run time. It has Boolean value. Default value is true.
TabIndex	Gets or sets the tab order of the control within its container.
BackColor	It is used to get or set background color.
Font	It is used to set Font Face, Font Style, Font Size and Effects of the text.
Visible	It is used to specify whether the control is visible or not at run time. It has Boolean value. Default value is true.
FlatStyle	It is used to get or set appearance of the Button Control when user moves mouse on it or click on it. It has following 4 options: **System, Popup, Standard, Flat**
Image	It is used to specify an image that is displayed on it.
ImageAlign	It is used to get or set alignment of the image that is displayed on it. control.
TextImageRelation	It is used to get or set position of text in relation with image. It has following 5 options: (1) Overlay (2) ImageAboveText

	(3) TextAboveImage (4) ImageBeforeText (5) TextBeforeImage It is used when user wants to display both text and image on Button Control.

Methods:

Method	**Description**
Show	It is used to show Button control at run time.
Hide	It is used to hide Button control at run time
Focus	It is used to set input focus on Button Control at run time

Example: Refer the example of label or textbox

3.1.4. Checkbox

- It is used to **set true/false or yes/no type options**. The user can **select** or **deselect** it. When a check box is **selected** it has the value **True**, and when it is **deselected**, it has the value **False**.

- The prefix of the checkbox is **"chk".**

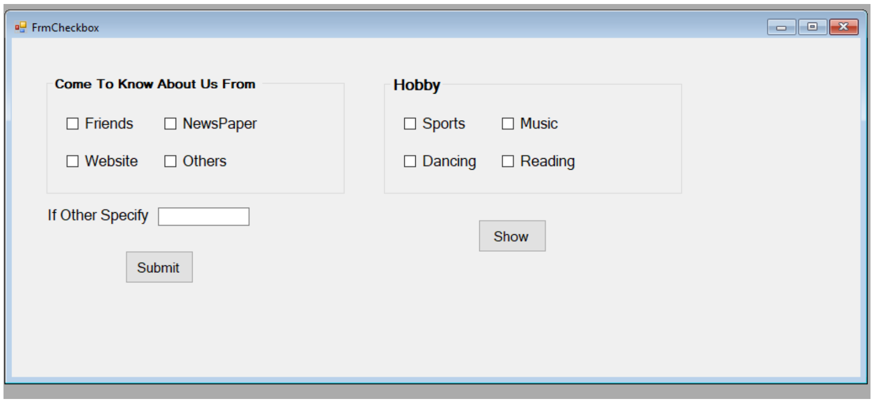
Properties:

Property	**Description**
Text	Gets or sets the current text of it.
Appearance	It is used to determining the appearance of the check box.
Autocheck	When check box is clicked it automatically change state (**True/False**). By default it is true.
CheckAlign	It Determine the location of the checkbox inside the control. By default it is Middleleft.
Checked	Indicating whether the check box is in checked state(True/False).
ThreeState	Indicates whether check box will allow three check states rather than two.
CheckState	Indicates the state of the checkbox. There are 3 types of chechstate: -checked -unchecked -indeterminate.

Events:

Event	**Descrption**
CheckedChanged	Raised when Checked property of the CheckBox control is changed

Example:



Source Code:

```
Public Class FrmCheckbox

    Private Sub btnsubmit_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnsubmit.Click
        Dim str As String
        str = " "
        If chkfrd.Checked = True Then
            str &= chkfrd.Text
            str &= " "
        End If
        If chknp.Checked = True Then
            str &= chknp.Text
            str &= " "
        End If
        If chkwebsite.Checked = True Then
            str &= chkwebsite.Text
            str &= " "
        End If
        If chkother.Checked = True Then
            str &= TextBox1.Text
            str &= " "
        End If

        If str <> Nothing Then
            MsgBox(str + vbLf + "Thank you")
        End If
    End Sub
    Private Sub btnshow_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnshow.Click
        If chksport.Checked = True Then
            MsgBox("Sports checked", MsgBoxStyle.OkCancel)
        Else
            MsgBox("Sports unchecked", MsgBoxStyle.OkCancel)
        End If
    End Sub
    Private Sub chkmusic_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles chkmusic.CheckedChanged
        If chkmusic.Checked = True Then
```

```
            MsgBox("Music checked")
        Else
            MsgBox("Music unchecked")
        End If
    End Sub
End Class
```

3.1.5. Radio Button

It is also known as an option button. We can check one radio button at a time. Radio button are generally use for **one in multiple options like Gender, Stream etc.** We can checked or unchecked the radiobutton.

- The prefix of the radiobutton is either **"opt" or "rad".**

Properties:

Property	Description
Text	Gets or sets the current text of it.
TextAlign	It is used for setting text alignment such as **Left,Right,Center**
Appearance	It is used to determining the appearance of it (**normal or button**).
Enabled	It is used to specify whether the control is enabled or not at run time. It has Boolean value. Default value is true.
Visible	It is used to specify whether the control is visible or not at run time. It has Boolean value. Default value is true.
Autocheck	When check box is clicked it automatically change state (True/False). By default it is true.
CheckAlign	It Determine the location of the checkbox inside the control. By default it is Middleleft.
Checked	Indicating whether the check box is in checked state (True/False).

3.1.6. Date Time Picker

It is used to select date and time.

Properties

Property	Description
BackgroundImage	It is used to set background Image for the DateTimePicker.
CalendarFont	It is used to set the font style for the calendar in the DateTimePicker control.
CustomFormat	It is used to set the custom date and time format string in the DateTimePicker control. dd/mm/yy
Checked	It is used to check whether the value property is checked with a valid date and time in the DateTimePicker control.
Format	It is used to set the format for the Date and time displayed in the Windows Form. dd/mm/yyyy
MaxDate	It is used to set the maximum date value.
Name	Indicates the name used in code to indentify the object.
MinDate	It is used to set the minimum date value.

Value	The current date and time value for this control.

Methods:

Method	Description
Select()	It is used to start or activate the DateTimePicker control.
Show()	It is used to display the control.
ToString()	It is used to return string that represent date and time.

Example:

3.1.7. List Box

- It is used to display list of items. It allows user to select one or more items from the listbox.we can add or design the list box by using the properties and events at runtime.
- -It is used for displaying larger number of choices.
- A scroll bar is automatically appears when many items in the ListBox. By default we can select only one item from ListBox.-The Prefix is "lst".

Properties:

Property	Description
Name	It represents unique name of control. It is used in the coding.
DataSource	It is used to bind a collection of items to a ListBox
DisplayMember	It gets or sets the field to be displayed in the ListControl.
Items	It gets the items of the ListBox.
Multicolumn	It allows multiple items to be displayed in the ListBox. It has Boolean Value (True/False). By default it is false.
SelectionMode	It defines how items are selected in the Listbox.The values of selectionMode are given below: - None: No item can be selected - One: Only one item can be selected (by default) - Multisimple:Multiple Item can be selected - MultiExtended: Multiple Item can be selected and user can use SHIFT,CTRL and arrow keys to make selections
SelectedIndex	It returns the index of the selected item. If more than one item is selected, it represents the lowest index. If no item is selected then it returns -1.
SelectedIndices	It returns the indices of all selected items. It is used when more than one item is selected.
SelectedItem	It returns the text of selected item. If more than one item is selected, then it returns the 1st Selected Item.
SelectedItems	It returns the text of all selected items.
Sorted	Set the ListBox item in sorted order either in ascending or descending. It has Boolean value (True/False). By default false.

ValueMember	It is used to get or set the actual value for the items in the control.

Methods

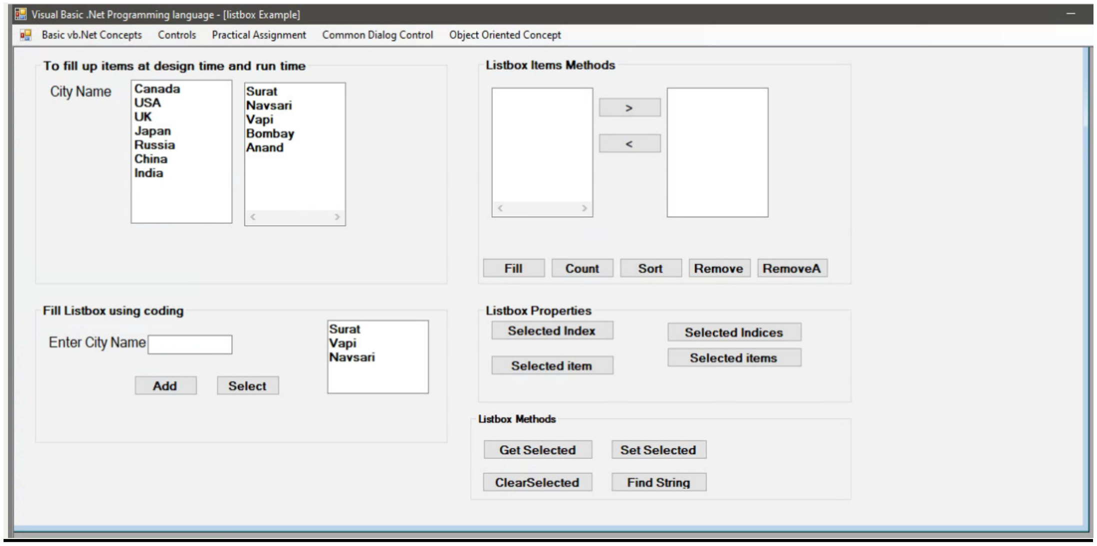
Method	Description
ClearSelected	Unselects all the items in the ListBox.
FindString	It returns the index of specified string. If it does not match then returns -1.
FindStringExact	It returns the index of exactly matches of the specified string.
GetSelected	It checks whether given index of item is selected or not. It returns Boolean value. If item is selected then it returns true otherwise false.
SetSelected	It selects or deselects the item of the given index.

Events

Event	Description
Click	It occurs when ListBox is selected
SelectedIndexChanged	It occurs when SelectedIndex property is changed

The Items in the list boxes are stored in the Item Collection. Properties and methods of Item collection are given below.

Example:



Source Code

```
Public Class FrmListbox

    Private Sub btnadd_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnadd.Click
        lstcitynamebutton.Items.Add(txtname.Text)
        txtname.Clear()
        txtname.Focus()
    End Sub
    Private Sub btnshiftright_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnshiftright.Click
        lstyourmenu.Items.Add(lstmymenu.SelectedItem) 'select from mymenu and
add in yourmenu
        lstmymenu.Items.Remove(lstmymenu.SelectedItem) 'remove from selected
item from menu
    End Sub

    Private Sub btnShiftLeft_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnShiftLeft.Click
        lstmymenu.Items.Add(lstyourmenu.SelectedItem) 'select from yourmenu
and add in menu
        lstyourmenu.Items.Remove(lstyourmenu.SelectedItem) 'remove from
selected item from yourmenu
    End Sub

    Private Sub lstname_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles lstcitynamerun.SelectedIndexChanged
        MsgBox(lstcitynamerun.SelectedItem)
        MsgBox(lstcitynamerun.SelectedIndex)
    End Sub

    Private Sub FrmListbox_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load
        ' Set the caption bar text of the form.
        'add items in listbox using coding
        Me.Text = "listbox Example"
        lstcitynamerun.Items.Add("Canada")
        lstcitynamerun.Items.Add("USA")
        lstcitynamerun.Items.Add("UK")
        lstcitynamerun.Items.Add("Japan")
        lstcitynamerun.Items.Add("Russia")
        lstcitynamerun.Items.Add("China")
        lstcitynamerun.Items.Add("India")
    End Sub

    Private Sub btnfill_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnfill.Click
        lstmymenu.Items.Add("Eductaion")
        lstmymenu.Items.Add("Sports")
        lstmymenu.Items.Add("Music")
        lstmymenu.Items.Add("Reading")
    End Sub

    Private Sub btnremove_Click_1(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnremove.Click
        lstmymenu.Items.Remove(lstmymenu.SelectedItem)

    End Sub
    Private Sub btncount_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btncount.Click
        MsgBox(lstmymenu.Items.Count)
        MsgBox(lstmymenu.Items.Item(0))
    End Sub

    Private Sub btnsort_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnsort.Click
        lstmymenu.Sorted = True
        lstcitynamerun.Items.Clear()
    End Sub

    Private Sub btnselectindex_Click(sender As Object, e As EventArgs)
Handles btnselectindex.Click
        MsgBox(lstcitynamedeg.SelectedIndex)
    End Sub
    Private Sub btnselectindice_Click(ByVal sender As Object, ByVal e As
EventArgs) Handles btnselectindice.Click
        Dim i As Integer
        For i = 0 To lstcitynamedeg.SelectedIndices.Count - 1 'used when more
than one selected items
            MsgBox(lstcitynamedeg.SelectedIndices.Item(i))
        Next
    End Sub

    Private Sub btnselectitem_Click(sender As Object, e As EventArgs) Handles
btnselectitem.Click
        MsgBox(lstcitynamedeg.SelectedItem)
    End Sub

    Private Sub btnselectitems_Click(sender As Object, e As EventArgs)
Handles btnselectitems.Click
        Dim i As Integer
        For i = 0 To lstcitynamedeg.SelectedItems.Count - 1 'use when more
than one selected items
            MsgBox(lstcitynamedeg.SelectedItems.Item(i))
        Next
    End Sub

    Private Sub btngetsec_Click(sender As Object, e As EventArgs) Handles
btngetsec.Click
        MsgBox(lstcitynamedeg.GetSelected(0)) 'true/false
        lstcitynamedeg.GetSelected(2)
    End Sub

    Private Sub btnsetsec_Click(ByVal sender As Object, ByVal e As EventArgs)
Handles btnsetsec.Click
        lstcitynamedeg.SetSelected(2, True)
        lstcitynamedeg.SetSelected(4, True)
        lstcitynamedeg.SetSelected(3, True)
    End Sub

    Private Sub btnclearsec_Click(sender As Object, e As EventArgs) Handles
btnclearsec.Click
        lstcitynamedeg.ClearSelected()
    End Sub

    Private Sub btnfindstring_Click(sender As Object, e As EventArgs) Handles
btnfindstring.Click
        MsgBox(lstcitynamedeg.FindString("Vapi")) 'it returns of the index of
the particular string
        MsgBox(lstcitynamedeg.FindString(lstcitynamedeg.SelectedItem))
    End Sub

    Private Sub btnselect_Click(sender As Object, e As EventArgs) Handles
btnselect.Click
        Dim i As Integer
        For i = 0 To lstcitynamebutton.SelectedItems.Count - 1 'use when more
than one selected items
            MsgBox(lstcitynamebutton.SelectedItems.Item(i))
        Next
    End Sub

    Private Sub btnremoveat_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnremoveat.Click
        lstmymenu.Items.RemoveAt(1)
    End Sub
End Class
```

3.1.8. Combo Box

- The ComboBox control is used to display a **drop-down list of various items**. It is a combination of a **text box in which the user enters an item and a drop-down list** from which the user selects an item.
- You can populate the list box items either from the **properties window or at runtime**.
- To add items to a ComboBox, select the ComboBox control and go to the properties window for the properties of this control. **Click the ellipses (...)** button next to the Items property.
- This opens the **String Collection Editor dialog box**, where you can enter the values one at a line.

Properties

Property	**Description**
DataBindings	It gets the data binding for the control.
Name	It represents unique name of control. It is used in the coding.
DataSource	It is used to bind a collection of items to a ComboBox
DisplayMember	It gets or sets the field to be displayed in the ComboBox.
ValueMember	It is used to get or set the actual value for the items in the control.
Items	It gets the items of the ComboBox.
SelectedIndex	It returns the index of the selected item. If more than one item is selected, it represents the lowest index. If no item is selected then it returns -1.
SelectedItem	It returns the text of selected item. If more than one item is selected, then it returns the 1st Selected Item.
SelectedText	Gets or sets the text that is selected in the editable portion of a ComboBox
SelectedValue	Gets or sets the value of the member property specified by the ValueMember property.
Sorted	Gets or sets a value indicating whether the items in the combo box are sorted
IntegralHeight	Indicates whether the combo box should resize to avoid showing partial items.(true(Default)/false)
MaxDropDownItems	The maximum number of entries to be display in the drop down list.

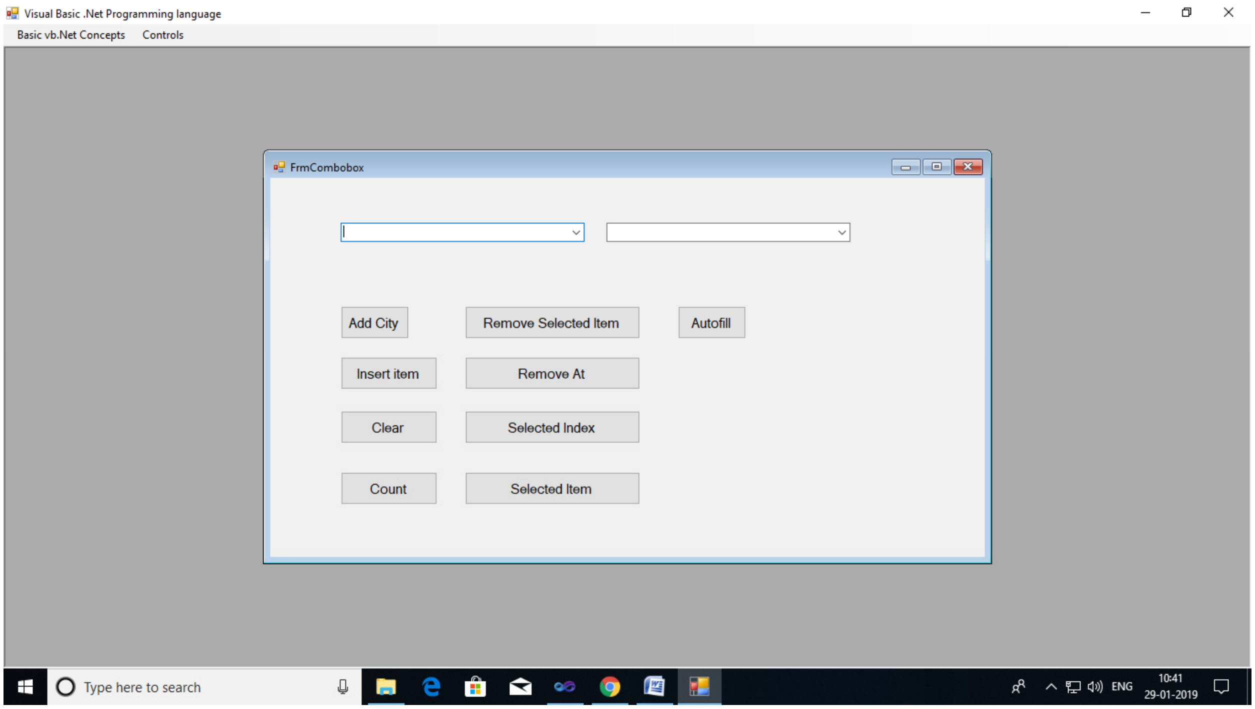
Methods

Method	**Description**
SelectAll	Selects all the text in the editable area of the combo box.
FindString	It returns the index of specified string. If it does not match then returns -1.
FindStringExact	It returns the index of exactly matches of the specified string.

Events

Event	**Description**
SelectedIndexChanged	It occurs when SelectedIndex property is changed

Example:



```
Public Class FrmCombobox

    Private Sub FrmCombobox_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        'Fill combo when form is load
        'cmbcity.Items.Add("India")
        'cmbcity.Items.Add("UK")
        'cmbcity.Items.Add("Canada")
    End Sub

    Private Sub cmbautofill_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs)
        MsgBox(cmbcity.SelectedItem)
    End Sub

    Private Sub btnadd_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnadd.Click
        Dim str As String
        str = InputBox("Enter City Name")
        If cmbcity.FindString(str) > -1 Then
            MsgBox("Already in the list", MsgBoxStyle.Critical, "error")
        Else
            cmbcity.Items.Add(str)
        End If
    End Sub

    Private Sub btninsertitem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btninsertitem.Click
```

```
        Dim pos, str As String
        str = InputBox("Enter City Name")
        pos = InputBox("Enter Position")
        If cmbcity.FindString(str) > -1 Then
            MsgBox("Already in the list", MsgBoxStyle.Critical, "error")
        Else
            cmbcity.Items.Add(str)
        End If
    End Sub

    Private Sub btnremoveat_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnremoveat.Click
        cmbcity.Items.RemoveAt(cmbcity.SelectedIndex)
    End Sub

    Private Sub btnclear_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnclear.Click
        cmbcity.Items.Clear()
    End Sub

    Private Sub btnselectindex_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnselectindex.Click
        MsgBox(cmbcity.SelectedIndex)
    End Sub

    Private Sub btncount_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btncount.Click
        MsgBox(cmbcity.Items.Count)
    End Sub

    Private Sub btnselectitem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnselectitem.Click
        MsgBox(cmbcity.SelectedItem)
    End Sub

    Private Sub btnremoveselect_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnremoveselect.Click
        cmbcity.Items.Remove(cmbcity.SelectedItem)
    End Sub

    Private Sub cmbcity_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cmbcity.SelectedIndexChanged
        MsgBox(cmbcity.SelectedItem)
    End Sub

    Private Sub btnautofill_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnautofill.Click
        'Fill combo when autofill button is click
        cmbautofill.Items.Add("India")
        cmbautofill.Items.Add("UK")
        cmbautofill.Items.Add("Canada")
    End Sub
End Class
```

3.1.9. Picture Box

– It is used for displaying images on the form. The Image property of the control allows you to set an image both at design time and at run time.

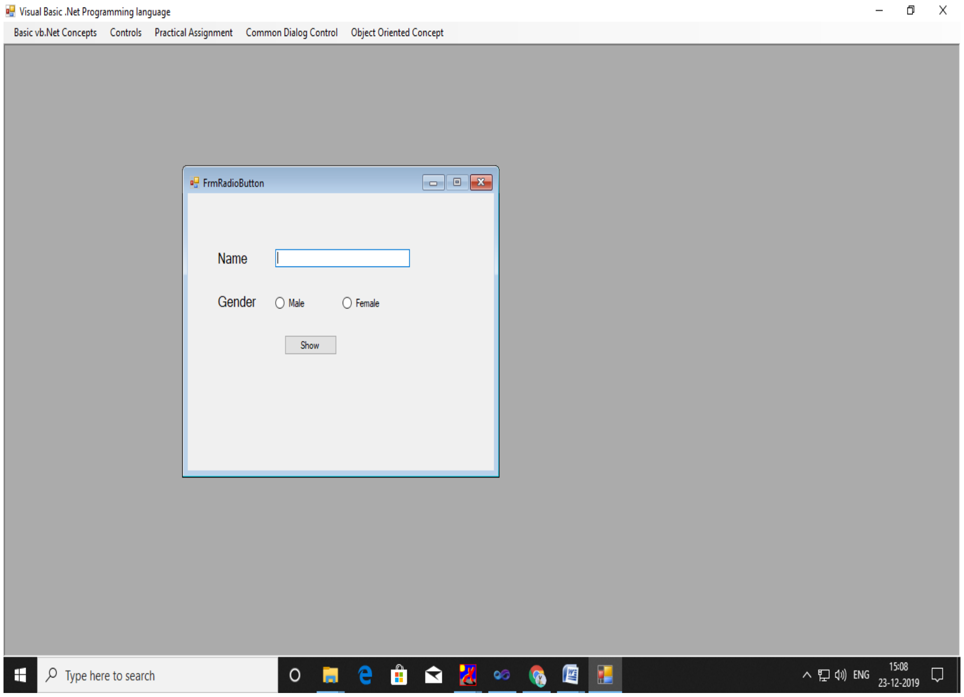
Properties

Property	**Description**
Name	It represents unique name of control. It is used in the coding.
Image	Gets or sets the image that is displayed in the control.
SizeMode	Determines the size of the image to be displayed in the control. This property takes its value from the PictureBoxSizeMode enumeration, which has values – • Normal – the upper left corner of the image is placed at upper left part of the picture box • StrechImage – allows stretching of the image • AutoSize – allows resizing the picture box to the size of the image • CenterImage – allows centering the image in the picture box • Zoom – allows increasing or decreasing the image size to maintain the size ratio.
Height	Gets or sets the height of the image
Width	Gets or sets the width of the image

Events

Event	**Description**
Click	Raise when user clicks on it.

Example



```
Public Class FrmRadioButton

    Private Sub btnshow_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnshow.Click
        Dim str As String
        str = "your name is " & txtname.Text
        If rdbmale.Checked = True Then
            str &= " your gender is male "
        Else
            str &= " your gender is Female "
        End If
        MsgBox(str, MsgBoxStyle.Information, "Information")
    End Sub

    Private Sub rdbmale_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles rdbmale.CheckedChanged
        PictureBox1.Image = Image.FromFile("D:\NMODI.jpg")
    End Sub

    Private Sub rdbfemale_CheckedChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles rdbfemale.CheckedChanged
        PictureBox1.Image = Image.FromFile("D:\SG.jpg")
    End Sub
End Class
```

```
Private Sub PictureBox1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles PictureBox1.Click
        Me.Text = "Picture Box Example" 'Set the title name for the form
        btnshow.Text = "Show"
        Label1.Text = "Click to display the image"
        Label1.ForeColor = Color.ForestGreen
    End Sub
```

3.1.10.Rich Text Box

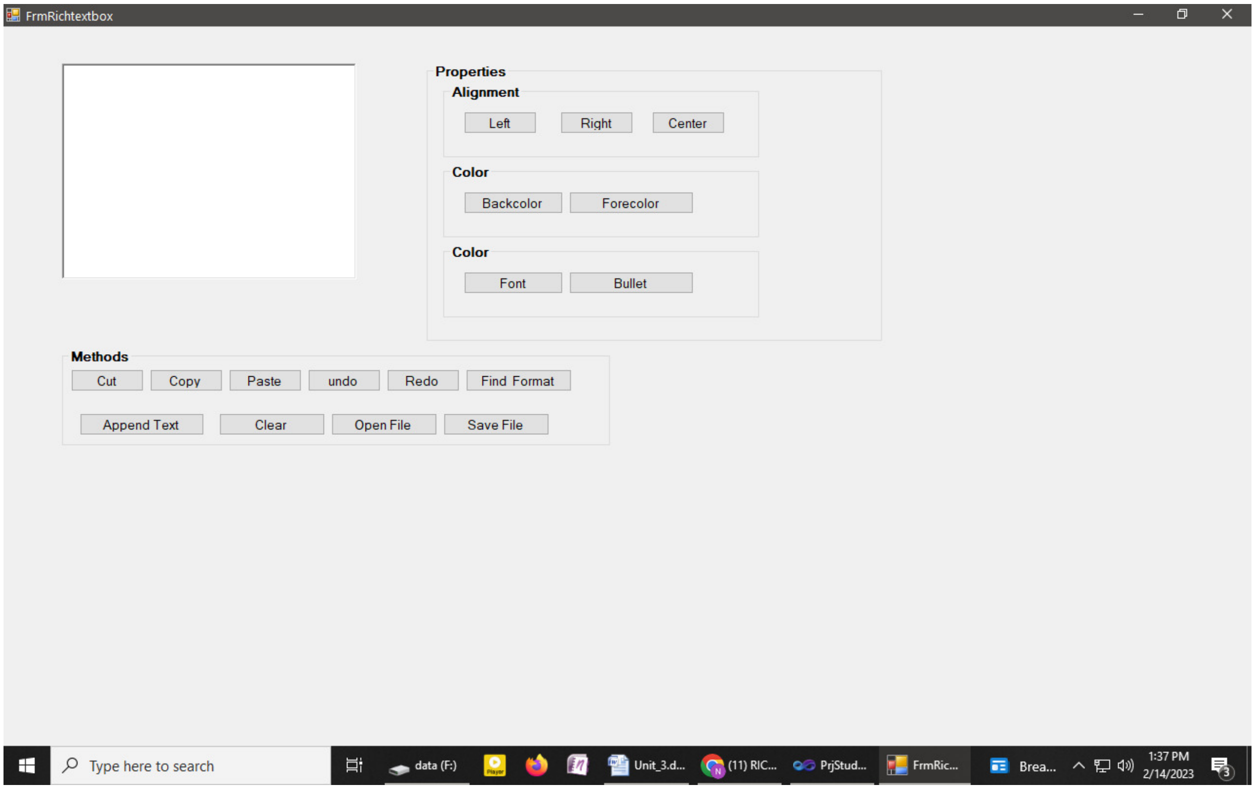
- It is a text editing control that allows users to enter and format text using a variety of fonts, colors, and styles. It is an advance version of textbox. It can load RTF (Rich Text Format), TXT format files for reading or editing. For example **WordPad**.
- **Properties**

Property	Description
Text	Gets or sets current text of the control.
SelectionAlignment	Gets or sets the Alignment to apply to current selection
SelectionBackColor	Gets or sets background color of text.
SelectionBullet	Gets or sets bullet list
SelectionFont	Gets or sets font of current selection
SelectionColor	Gets or sets text color of the current text selection.
SelectedText	Gets or sets the selected text
ZoomFactor	Define the scaling factor of it.

Methods:

Method	Description
AppendText	Appends text to the end of the control's text.
Cut	Cuts the selected text to the Clipboard.
Copy	Copies the selected text to the Clipboard.
Paste	Replaces the current selection in the text box with the contents of the Clipboard
Undo	Undoes the last edit operation in the control.
Find	Finds the specified text within the control.
LoadFile	Loads the contents of a file into the control.
SaveFile	Saves the contents of the control to a file.
Clear	Clears all text from the control.

Example:



Source Code

```
Public Class FrmRichtextbox

    Private Sub btncut_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btncut.Click
        richtext.Cut()
    End Sub

    Private Sub btncopy_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btncopy.Click
        richtext.Copy()
    End Sub

    Private Sub bntpaste_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles bntpaste.Click
        richtext.Paste()
    End Sub

    Private Sub btnundo_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnundo.Click
        richtext.Undo()
    End Sub
    Private Sub btnredo_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnredo.Click
        richtext.Redo()
    End Sub

    Private Sub btnclear_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)

    End Sub
    Private Sub btnselectall_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        richtext.SelectAll()
    End Sub
    Private Sub btnbaccolor_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnbaccolor.Click
        richtext.SelectionBackColor = Color.Red
    End Sub

    Private Sub btnforcolor_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnforcolor.Click
        richtext.SelectionColor = Color.PowderBlue
    End Sub

    Private Sub btnfont_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnfont.Click
        Dim f As New Font("Arial", 20)
        richtext.SelectionFont = f
    End Sub
    Private Sub btnbullet_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnbullet.Click
        richtext.SelectionBullet = True
    End Sub
    Private Sub btnappenttext_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnappenttext.Click
        richtext.SelectionStart = richtext.Text.Length
        richtext.ScrollToCaret()
        richtext.AppendText("This is some additional text that will be added
to the bottom of the RichTextBox control.")
    End Sub

    Private Sub btnleft_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnleft.Click
        richtext.SelectionAlignment = HorizontalAlignment.Left
    End Sub

    Private Sub btnright_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnright.Click
        richtext.SelectionAlignment = HorizontalAlignment.Right
    End Sub
    Private Sub btnCenter_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCenter.Click
        richtext.SelectionAlignment = HorizontalAlignment.Center
    End Sub

    Private Sub btnclear_Click_1(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnclear.Click
        richtext.Clear()
    End Sub

    Private Sub btnopen_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnopen.Click
        richtext.LoadFile("F:\p1.txt", RichTextBoxStreamType.PlainText)
    End Sub

    Private Sub btnsave_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnsave.Click
        richtext.SaveFile("F:\p1.txt", RichTextBoxStreamType.PlainText)
    End Sub

    Private Sub btnfind_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnfind.Click
        richtext.SelectionStart = richtext.Find("SY")
        richtext.SelectionColor = Color.RoyalBlue
        richtext.SelectionStart = richtext.Find("BCA")
        richtext.SelectionColor = Color.Red
    End Sub
End Class
```

3.1.11.Tree View

- It is a powerful tool that allows you to display **hierarchical data** in a tree structure. The control can be used to display a variety of data, including files and folders, organizational charts, family trees, and more.
- It is collection of nodes. The main starting node is called **root node.** Under the root, a real tree is made of branches and leaves. A node can have a node as a child. We can expand and collapse these nodes by clicking them.
- Tree view has nodes collection.
- **Properties of Nodes Collection**
- **Properties**

Property	Description
Count	It returns total number of elements.
Items	It returns the nodes of the given index.

Methods

Method	Description
Add	It adds nodes in the Tree view
Clear	It clear all the nodes from the Tree view
Contains	It checks whether the given node is a part of Tree view or not. It returns Boolean value.
Insert	We can insert node in the Tree View in any place by using this method. It contains two arguments. Index and node
Remove	It removes the node from the Tree View. The argument of Remove () method is the node.
RemoveAt	It removes the node from the Tree View by index.

Properties of Tree View

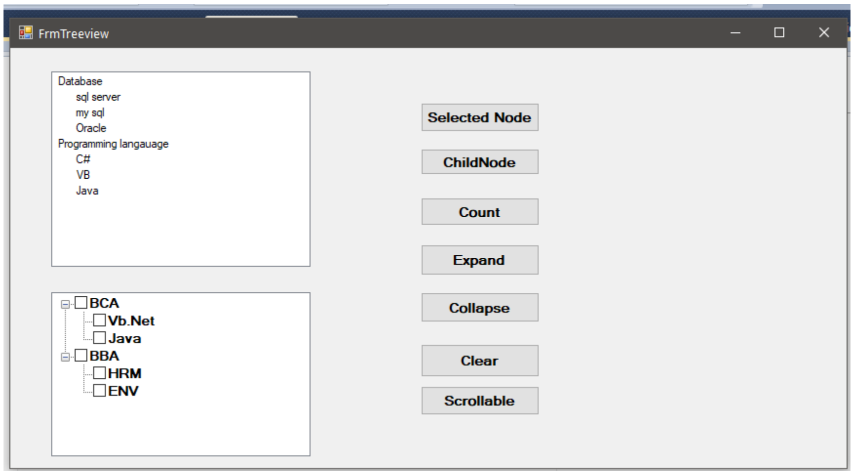
Property	Description
Nodes	It is used to gather all the nodes used in the tree.
SelectedNode	It is used to get or set the tree node that is selected in the tree view control.
ShowRootLines	It gets or sets a value that represents whether you want to draw lines between the trees nodes connected with the root of the tree view.
Path Separator	The Path Separator property of the Tree View Control is used to set a delimiter string between the tree node paths.
ShowPlusMinus	It is used to get or set a value representing whether you want to display the plus (+) or minus sign button next to tree nodes containing the child nodes.
ShowLines	It takes a value representing whether you want to draw lines between the tree nodes of the tree view control.
TopNode	It is used to get or set full visible tree nodes on top of other nodes in the tree view control.
VisibleCount	It is used to obtain the fully visible tree node in the tree view control.
ItemHeight	The ItemHeight property is used to set the height of each tree node in control.
Scrollable	The Scrollable property is used in the tree-view to display the scroll bar by setting the value in control.

Methods:

Method	Description
ExpandAll()	As the name suggests, an ExpandAll method is used to expand all the tree nodes.
CollapseAll	It is used to collapse all tree nodes, including all child nodes in the tree view control.
Sort()	A Sort method is used to sort the tree nodes that are available in the tree view control.

GetNodeCount	It is used to count the number of nodes that are available in the tree view control.
ToString	ToString method is used to return the name of the string that is in the tree view control.

Example:



Source Code:

```
Public Class FrmTreeview

    Private Sub btnsel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnsel.Click
        MsgBox(tvcourse.SelectedNode.Text)
        MsgBox(tvcourse.SelectedNode.FullPath)
    End Sub

    Private Sub btnchildnode_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnchildnode.Click
        MsgBox(tvcourse.SelectedNode.Nodes(0).Text)
        MsgBox(tvcourse.SelectedNode.Nodes(1).Text)
        MsgBox(tvcourse.SelectedNode.Nodes(2).Text)
    End Sub
    'Method
    Sub filltreeview()
        tvcourse.Nodes.Add("Database") 'this is root node
        tvcourse.Nodes(0).Nodes.Add("sql server")
        tvcourse.Nodes(0).Nodes.Add("my sql")
        tvcourse.Nodes(0).Nodes.Add("Oracle")
        tvcourse.Nodes.Add("Programming langauage") 'this is root node
        tvcourse.Nodes(1).Nodes.Add("C#")
        tvcourse.Nodes(1).Nodes.Add("VB")
        tvcourse.Nodes(1).Nodes.Add("Java")
    End Sub
    Private Sub FrmTreeview_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load
        filltreeview()
    End Sub

    Private Sub btncount_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btncount.Click
        MsgBox(tvcourse.Nodes.Count)
    End Sub

    Private Sub btnexpand_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnexpand.Click
        tvcourse.ExpandAll()
        Dim nodeCount As Integer = TreeView1.GetNodeCount(True)
        MessageBox.Show("Total nodes in the TreeView control: " &
nodeCount.ToString())
    End Sub

    Private Sub btncollapse_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btncollapse.Click
        tvcourse.CollapseAll()
    End Sub

    Private Sub btnclear_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnclear.Click
        tvcourse.Nodes.Clear()
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnscroll.Click
        tvcourse.Scrollable = True
    End Sub
End Class
```

3.1.12.Tool Tip

- A tooltip is a small pop-up window that displays some information when you rollover on a control.
- Tooltip class represents a tooltip control. Once a Tooltip object is created, we need to call **SetToolTip** method and pass a control and text. The following code snippet creates a Tooltip and attach to a Button control using **SetToolTip** method.

Properties

Property	Description
Active	A tooltip is currently active.
AutomaticDelay	Automatic delay for the tooltip.
AutoPopDelay	The period of time the ToolTip remains visible if the pointer is stationary on a control with specified ToolTip text.

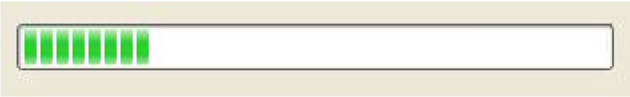
InitialDelay	Gets or sets the time that passes before the ToolTip appears.
IsBaloon	Gets or sets a value indicating whether the ToolTip should use a balloon window.
ReshowDelay	Gets or sets the length of time that must transpire before subsequent ToolTip windows appear as the pointer moves from one control to another.
ShowAlways	Displays if tooltip is displayed even the parent control is not active.
ToolTipIcon	Icon of tooltip window.
ToolTipTitle	Title of tooltip window.
UseAnimation	Represents weather an animation effect should be used when displaying the tooltip.
UseFading	Represents weather a fade effect should be used when displaying the tooltip.

Example:

```
Dim buttonToolTip As New ToolTip()
buttonToolTip.ToolTipTitle = "Button Tooltip"
buttonToolTip.UseFading = True
buttonToolTip.UseAnimation = True
buttonToolTip.IsBalloon = True
buttonToolTip.ShowAlways = True
buttonToolTip.AutoPopDelay = 5000
buttonToolTip.InitialDelay = 1000
buttonToolTip.ReshowDelay = 500
buttonToolTip.SetToolTip(Button1, "Click me to execute.")
```

3.1.13.Progress bar

- It is used **to graphically display** the progress of particular task. Thus using Progress Bar control you can display **how much task has been completed and how much task is remaining.**
- It shows the progress of any **background activity**. It is a good idea to show the progress to the end user when an application is performing complex or time consuming background tasks.



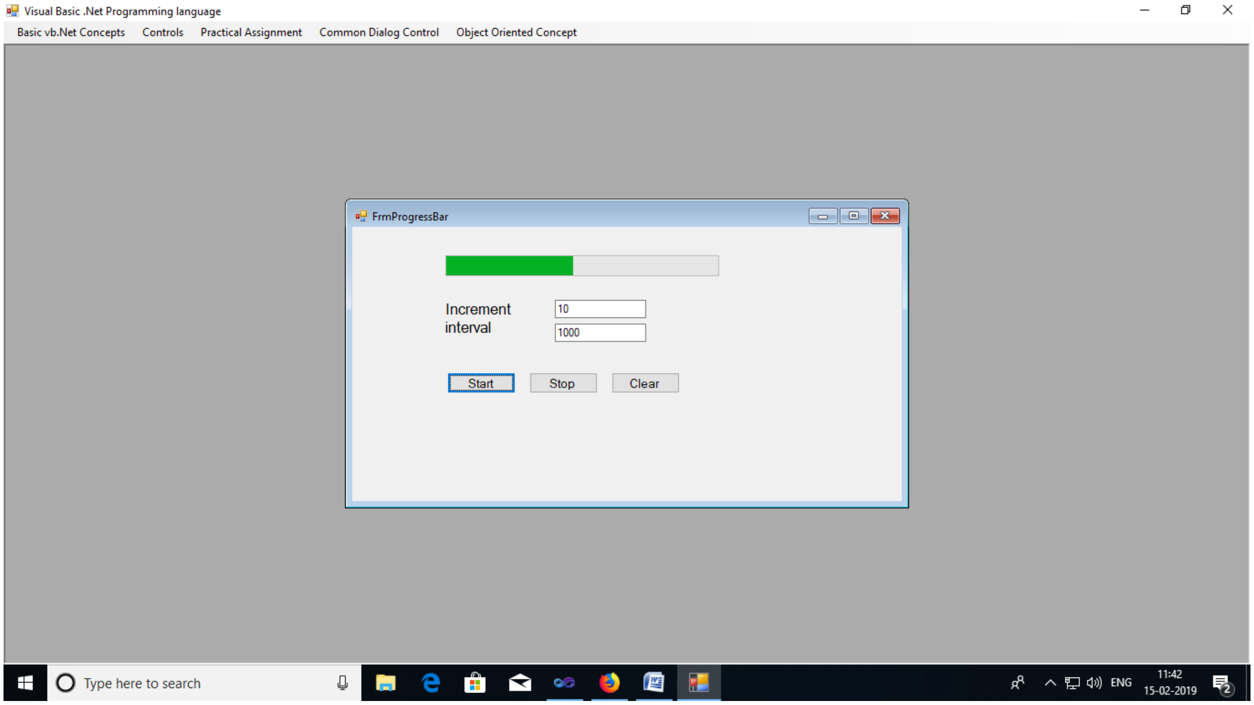
Properties

Property	Description
Minimum	It Get or Set Lower Bound of the range within which ProgressBar Control works.
Maximum	It Get or Set Upper Bound of the range within which ProgressBar Control works.
Value	It Get or Set current value of the ProgressBar within range

	specified using Minimum and Maximum property.
Step	It Get or Set Step value by which the current value of ProgressBar control is Increment.
Style	It is used to set the Style of ProgressBar Control. It can have one of the following value: Blocks,Continuous,Marquee
Visible	It is used to set weather ProgressBar control is visible on the form or not. It has boolean value true or false. Default value is true.
Enabled	It is used to set weather ProgressBar control is enabled or not. It has boolean value true or false. Default value is true.
MarqueeAnimationSpeed	It Get or Set speed of marquee animation when Style property of ProgressBar Control is set to marquee. The speed is in milisecond. Default value is 100 ms.

Method Name	**Description**
Increment	It is used to increment the current value of ProgressBar Control by specific value. Syntax: **ProgressBar1.Increment(value)**
PerformStep	It is used to increment the current value of ProgressBar Control by the value specified in the Step property of ProgressBar. Syntax: **progressBar1.PerformStep()**

Example:



Increment 10

interval 1000

```
Public Class FrmProgressBar
```

```
    Private Sub btnstart_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnstart.Click

        Timer1.Start()

    End Sub


    Private Sub btnstop_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnstop.Click

        Timer1.Stop()

    End Sub

    Private Sub btnclear_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnclear.Click

        Timer1.Stop()
        ProgressBar1.Value = 0

    End Sub

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer1.Tick

            ProgressBar1.Step = 5
        ProgressBar1.PerformStep()

        'ProgressBar1.Increment(TextBox1.Text)
        'Timer1.Interval = TextBox2.Text 'interval in minisecond .1000
milisecond =1 second'
  End Sub
End Class
```

3.1.14. Timer Control

– It is used when user wants to perform some task or action continuously at regular interval of time.

Properties

Property Name	Description
Name	It is used to specify name of the Timer Control.
Enabled	It is used to determine whether Timer Control will be enabled or not. It has Boolean value true or false. Default value is false.
Interval	It is used to specify interval in millisecond. Tick event of Timer Control generates after the time which is specified in Interval Property.

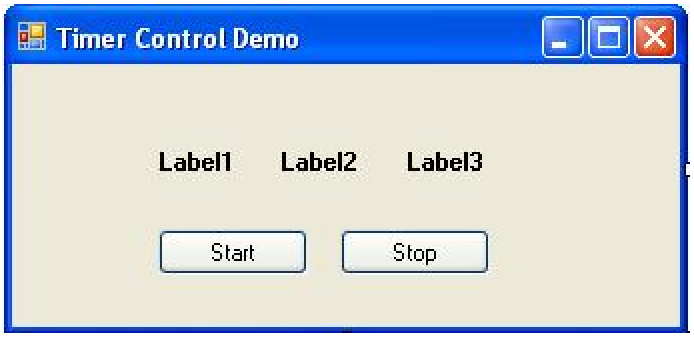
Method

Method Name	Description
Start	This method is used to start the Timer Control.
Stop	This method is used to stop the Timer Control.

Event

Event Name	Description
Tick	Tick event of the Timer Control fires continuously after the time which is specified in the Interval property of Timer Control.

Example



```
Private  Sub  btnstart_Click(ByVal  sender  As  System.Object,  ByVal  e  As
System.EventArgs) Handles btnstart.Click
        Timer1.Start()
        lblHour.Text = Now.Hour
        lblMinute.Text = Now.Minute
        lblSecond.Text = Now.Second
    End Sub


Private  Sub  btnstop_Click(ByVal  sender  As  System.Object,  ByVal  e  As
System.EventArgs) Handles btnstop.Click
        Timer1.Stop()
    End Sub
```

3.1.15. Masked Text box

It s used to **validate user input** on a form. For example, if you need a Textbox that should accept number in a certain format. It is similar to the textbox control. But it provides access to **mask** or change the format of input as well as output.

Properties:

Property	Description
Mask	It is used to get or set format string which determines whether characters entered in MaskedTextBox are valid or not.
AllowPromptAsInput	It is used to specify whether Prompt character can be entered as valid input character in MaskedTextBox or not. It has Boolean value. Default value is true.
AsciiOnly	It is used to specify whether only ASCII characters can be entered as valid input character in MaskedTextBox or not. It has Boolean value. Default value is false.
BeepOnError	It is used to specify whether control will generate system beep sound on each invalid character input or not. It has Boolean value. Default value is false.
PromptChar	It is used to get or set Prompt character for MaskedTextBox Control. This character is displayed in MaskedTextBox when user has not entered any character
Text	It is used to get or set text associated with it.
TextAlign	It is used to get or alignment of the text associated with it.
TabIndex	It is used to get or set Tab order of it.
TabStop	It is used to specify whether user can use TAB key to set focus on it or not. It has Boolean value. Default value is true.
Visible	It is used to specify whether the Control is visible or not at run time. It has Boolean value. Default value is true.

Methods:

Method	Purpose
Append Text	It is used to append text at the end of current text in it.
Clear	It is used to clear all text from it.
Cut	It is used to move current selection of it into clipboard.
Copy	It is used to copies selected text of it into clipboard.
Paste	It is used to replace current selection of MaskedTextBox by contents of clipboard. It is also used to move contents of Clipboard to MaskedTextBox control where cursor is currently located.
Select	It is used to select specific text from it.
SelectAll	It is used to select all text of it.

DeselectAll	It is used to deselect all text selected in it.
Show	It is used to show the control at run time.
Hide	It is used to hide the control at run time.
Focus	It is used to set input focus on the control at run time.

Events:

Event	Description
MaskChanged	It fires each time a mask property is changed.
TextChanged	It fires each time a text in the MaskedTextBox control changed.

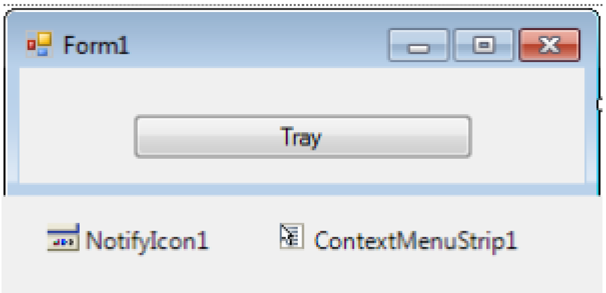
Example:

Mask	
0 or 9	Number only
?	Letter only
A	Alphanumeric character

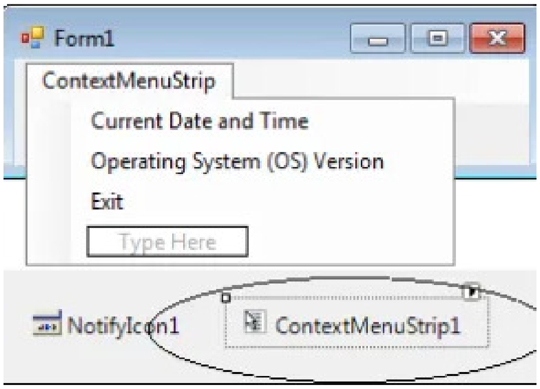
3.1.16. Notify Icon

It is used to add system tray notification functionality to a Windows Forms application. When an application is run, an icon will be added to the **system tray** and we can add double click or menus to the icon to take some actions.

Drag a **"ContextMenuStrip"**, **"NotifyIcon"**, **"Button"** and it will look like this.



After that, click the ContextMenuStrip and add the following items into it.



After adding the items, double click the **"Current Date and Time"** to fire the click event handler of it. Set the current date and time in the pop-up message.

After that, go back to the Form Design and double click the **"Operating System (OS) Version"** to fire the click event handler of it. Set the current OS version in the pop-up message.

Go back to the Form Design again and double click the **"exit"** to fire the click event handler of it. Put this code for closing the application.

Then set the following 2 properties for the NotifyIcon.

Property	**Description**
Icon	Set the icon for the NotifyIcon.
ContentMenuStrip	The short cut menu to show when the user right clicks the icon.

```
Public Class Form1
    Private Sub NotifyIcon1_MouseDoubleClick(ByVal sender As System.Object, ByVal e
As System.Windows.Forms.MouseEventArgs) Handles NotifyIcon1.MouseDoubleClick
        'WHEN YOU DOUBLE CLICK THE ICON IT WILL DISAPPEAR AND THE FORM WILL BE SHOWN.
        NotifyIcon1.Visible = False
        Me.Show()
    End Sub

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        'DISAPPEARING THE ICON IN THE SYSTEM TRAY ON THE FIRST LOAD.
        NotifyIcon1.Visible = False
        Me.Hide()
    End Sub

    Private Sub CurrentDateAndTimeToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
CurrentDateAndTimeToolStripMenuItem.Click
        'GET THE CURRENT DATE
        MsgBox("Today's date is " & Now.ToLongDateString() & ".")
    End Sub
    Private Sub OperatingSystemOSToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
OperatingSystemOSToolStripMenuItem.Click
        'GET THE CURRENT OPERATING SYSTEM INFORMATION
        MsgBox(My.Computer.Info.OSFullName & vbCrLf & "Version " &
My.Computer.Info.OSVersion)
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        'HIDE THE FORM AND VISIBLE THE NOTIFYICON SO THAT IT WIIL APPEAR IN THE
SYSTEM TRAY
        NotifyIcon1.Visible = True
        NotifyIcon1.Text = "System Information"
        Me.Hide()
    End Sub

    Private Sub ExitToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ExitToolStripMenuItem.Click
        End
    End Sub
End Class
```

3.1.17.Link Label

– It is a label control that can display a hyperlink.

– **Properties**

Property	Description
AutoSize	Gets or sets a value indicating whether the control automatically adjusts its size to fit its contents.
LinkArea	Gets or sets the area of the control that is treated as a link.
LinkBehavior	Gets or sets a value that specifies the behavior of the link.
Links	Gets the collection of links in the control
LinkColor	Gets or sets the color used for links in the control.
Text	Gets or sets the text displayed by the control.

Methods:

Method	Description
DoDragDrop	Initiates a drag-and-drop operation.
Focus	Sets input focus to the control.
OnLinkClicked	Raises the LinkClicked event.
ToString	Returns a string representation of the control.

Event:

Event	Description
LinkClicked	Triggered when the link is clicked.

3.1.18. Checked List Box

- It is a ListBox control with CheckBox displayed in the left side where user can select a single or multiple items.
- **Properties**

Property	**Description**
Items	It gets or sets the items contained in the CheckedListBox control.
CheckedItems	It gets a collection of the checked items in the CheckedListBox control.
CheckedIndices	It gets a collection of the indices of the checked items in the CheckedListBox control.
CheckOnClick	It gets or sets a value indicating whether the checkboxes are automatically checked or unchecked when the item is clicked.
SelectionMode	It gets or sets a value indicating whether the user can select multiple items or only one item.
ThreeState	It gets or sets a value indicating whether the checkboxes have three states: checked, unchecked, and indeterminate.
DisplayMember	It gets or sets the property to display for each item in the CheckedListBox control.
ValueMember	It gets or sets the property to use as the actual value of each item in the CheckedListBox control.
DataSource	It gets or sets the data source for the CheckedListBox control.
ScrollAlwaysVisible	It gets or sets a value indicating whether the vertical scrollbar is always visible or not.

Methods

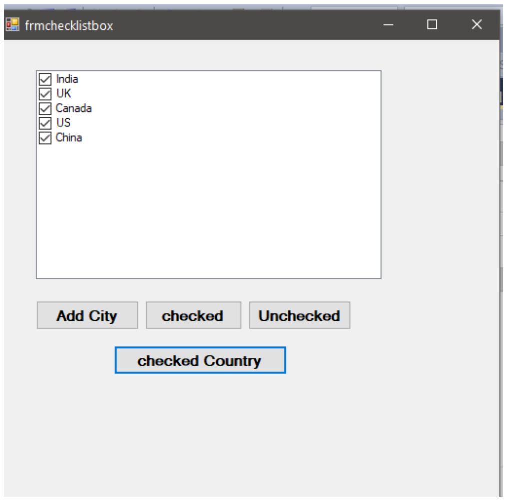
Method	**Description**
SetItemChecked(index, value)	It sets the check state of the item at the specified index in the CheckedListBox control to the specified value.
GetItemChecked(index)	It gets the check state of the item at the specified index in the CheckedListBox control.
GetItemCheckState(index)	It gets the check state of the item at the specified index in the CheckedListBox control and returns it as a CheckState enumeration value.
SetItemCheckState(index, state)	It sets the check state of the item at the specified index in the CheckedListBox control to the specified CheckState enumeration value.
GetItemText(index)	It gets the text of the item at the specified index in the CheckedListBox control.
FindStringExact(value)	It searches for the item with the specified text and returns the index of the first item found.
FindStringExact(value, startIndex)	It searches for the item with the specified text, starting from the specified index, and returns the index of the first item

	found.
GetSelected(index)	It gets a value indicating whether the item at the specified index is selected.
SetSelected(index, value)	It sets a value indicating whether the item at the specified index is selected.
ClearSelected()	It clears the selection in the CheckedListBox control.

Event:

Event	Description
SelectedIndexChanged	It occurs when SelectedIndex property is changed

Example:



Source Code:

```
Public Class frmchecklistbox

    Private Sub btnadd_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnadd.Click
        Dim str As String
        str = InputBox("Enter City Name")
        If clbcity.FindString(str) > -1 Then
            MsgBox("Already in the list", MsgBoxStyle.Critical, "error")
        Else
            clbcity.Items.Add(str)
        End If
    End Sub

    Private Sub btncheck_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btncheck.Click
        Dim i As Integer
        For i = 0 To clbcity.Items.Count - 1
            clbcity.SetItemChecked(i, True)
        Next
    End Sub

    Private Sub btnunchecked_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnunchecked.Click
        Dim i As Integer
        For i = 0 To clbcity.Items.Count - 1
            clbcity.SetItemChecked(i, False)
        Next
    End Sub
    Sub autofill()
        clbcity.Items.Add("India")
        clbcity.Items.Add("UK")
        clbcity.Items.Add("Canada")
        clbcity.Items.Add("US")
        clbcity.Items.Add("China")
    End Sub
    Private Sub frmchecklistbox_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load
        autofill()
    End Sub

    Private Sub btnchkcity_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnchkcity.Click
        Dim j As Integer
        Dim strans As String = ""
        For j = 0 To clbcity.Items.Count - 1
            If clbcity.GetItemChecked(j) = True Then
                strans &= clbcity.Items.Item(j) & vbCrLf
            End If
        Next
        MsgBox(strans)
    End Sub
End Class
```

3.1.2 Container control

- It is a control that can contain other controls within it. They are used to organize and group related controls, and to provide layout and positioning of controls within a form or other container.
- **Panel control:** It is a simple container control that can contain other controls, such as buttons, labels, text boxes, and other controls. You can use the Panel control to group related controls, and to provide a simple layout for controls within a form.

- **Group Box control**: It is similar to the Panel control, but it includes a border and a caption, which can be used to group related controls and to provide a visual separation between different groups of controls.

  Common Properties of Panel and Group Box

Property	**Description**
BackColor	Gets or sets the background color of the control.
BorderStyle	Gets or sets the border style of the control.
Controls	Gets the collection of controls contained within the control.
Padding:	Gets or sets the padding within the control.
AutoScroll:	Gets or sets a value indicating whether the control enables automatic scrolling when the control is resized.
AutoSize	Gets or sets a value indicating whether the control is automatically resized to fit its contents.
Enabled	Gets or sets a value indicating whether the control is enabled.
Visible	Gets or sets a value indicating whether the control is visible.

- Both the Panel and GroupBox controls have similar properties, such as BackColor, Controls, Padding, AutoSize, Enabled, and Visible. However, **the GroupBox control also has a Text property that allows you to set the caption or title of the group box.**

3.1.3 Data –Data set, Data Grid

3.1.4 Component

3.1.4.1 Image List

- It is a simple control that stores images used by other controls at runtime.
- **Properties**

Property	**Description**
Images	Gets the collection of images in the image list.
ColorDepth	Gets or sets the color depth of the images in the image list.
TransparentColor	Gets or sets the color that is treated as transparent.
ImageSize	Gets or sets the size of the images in the image list.
Tag	Gets or sets an object that contains data about the control.

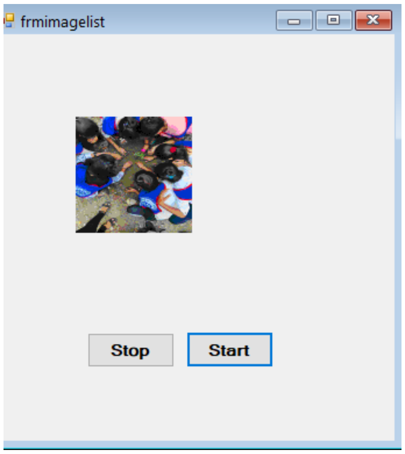
Methods:

Method	**Description**
Clear()	Removes all the images from the image list.
Add(key As String, image As Image)	Adds a new image to the image list, using the specified key to identify it.
IndexOfKey(key As String)	Returns the index of the image with the specified key.

RemoveByKey(key As String)	Removes the image with the specified key from the image list.

Method	Description
Disposed	Occurs when the control is disposed of by a call to the Dispose method.
ItemAdded	Occurs when an item is added to the image list.
ItemRemoved:	Occurs when an item is removed from the image list.

- you can create an image list by using the **ImageList** control. Here are the basic steps:
- Add an **ImageList** control to your form.
- Add images to the ImageList control by either:
  - a. Setting the **Images** property in the Properties window and manually adding images.
  - b. Adding images programmatically using the **ImageList.Images.Add** method.
- Use the **ImageList** control in other controls that support images, such as **PictureBox** and **ListView**.
- **Example**



Source Code:

```
Public Class frmimagelist
    Dim n As Integer = 0
    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer1.Tick
        PictureBox1.Image = ImageList1.Images(n)
        If (n = ImageList1.Images.Count - 1) Then
            n = 0
        Else
            n += 1
        End If
    End Sub
    Private Sub btnstart_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnstart.Click
        Timer1.Start()
    End Sub

    Private Sub btnstop_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnstop.Click
        Timer1.Stop()
    End Sub
End Class
```

3.1.4.2 Error Provider

It is used to get or set error message for particular control on the form.

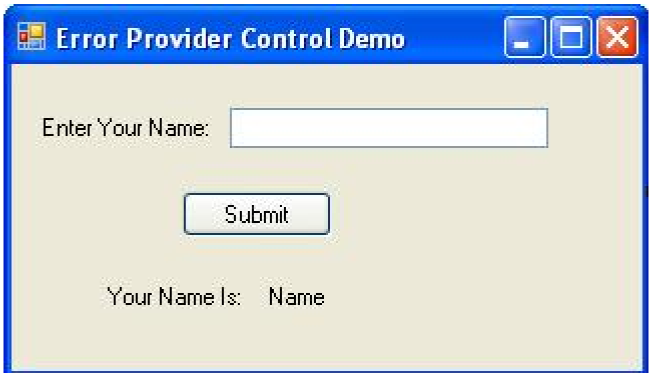
Properties

Property	Description
Name	It is used to specify name of ErrorProvider Control.
BlinkRate	It is used to specify rate in milisecond at which error icon blinks.
BlinkStyle	It is used to specify Blink Style for Error Icon. It can be: AlwaysBlink, NeverBlink, BlinkIfDifferentError
Icon	It is used to specify ICON to be displayed near to the control when error is set for that control.

Methods

Method	Description
SetError	It is used to set error message for particular control. **Ex:**ErrorProvider1.SetError(ControlName,"ErrorMessage")
GetError	It is used to retrieve current error message of particular control. **Ex:** ErrorProvider1.GetError (ControlName).
Clear	It is used to clear all the settings of ErrorProvider Control. **Ex:**ErrorProvider1.Clear()

Example:



Source Code:

Now Double click on Submit Button and write following code in Click event.

```
 ErrorProvider1.Clear()
If txtName.Text = "" Then
    ErrorProvider1.SetError(txtName, "Please Enter Name")
    txtName.Focus()
    lblname.Text = ErrorProvider1.GetError(txtname)
Else
    lblName.Text = txtName.Text
End If
```

3.1.4.3 Help Provider

- It provides popup or online help for a control.

Properties and Methods:

Properties and Methods:	**Description**
SetShowHelp	It specifies whether Help is displayed for the specified control
SetHelpString	It specifies the Help string associated with the specified control
SetHelpNavigator	It specifies the Help command to use when retrieving Help from the Help file for the specified control
SetHelpKeyword	It specifies the keyword used to retrieve Help when the user invokes Help for the specified control
HelpNamspace	gets or sets a value specifying the name of the Help file associated with this HelpProvider

Example:

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Help.ShowHelp(Label1, "f:/help.html")
    End Sub

    Private Sub frmhelprovider_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim hlpProvider As HelpProvider
        hlpProvider = New System.Windows.Forms.HelpProvider()
        hlpProvider.SetShowHelp(TextBox1, True)
        hlpProvider.SetHelpString(TextBox1, "Enter a valid text here.")
        hlpProvider.SetShowHelp(Button1, True)
        hlpProvider.SetHelpString(Button1, "Click this button.")
        ' Help file
        hlpProvider.HelpNamespace = "f:/help.html"
        hlpProvider.SetHelpNavigator(TextBox1, HelpNavigator.TableOfContents)
    End Sub
```

3.2. Working with Menus and Dialogue Boxes

MDI: Multiple Document Interface

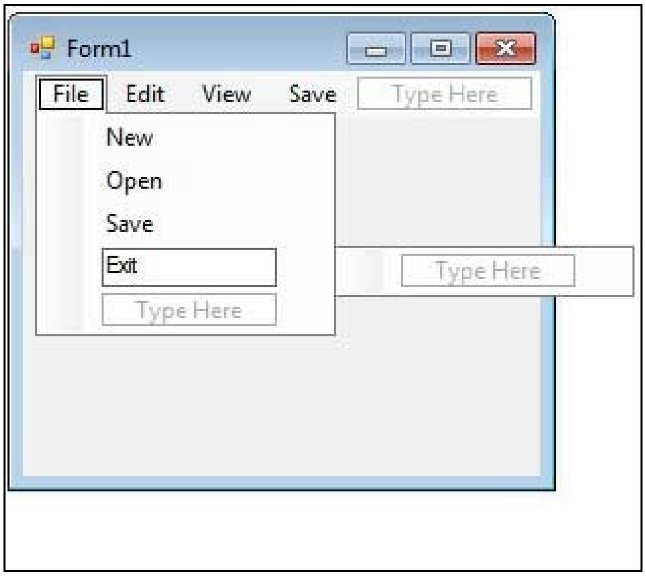
- Multiple-document interface (MDI) applications enable you to display multiple documents at the same time, with each document displayed in its own window. MDI applications often have a Window menu item with submenus for switching between windows or documents.
- **Create an MDI form**
- Create a new VB.Net project, and then you will get a default form **Form1** then open the Properties window, set the following properties
  - **IsMdiContainer= true.**
  - **Name=frmMDI**
  - **Text=MDI form Text(for example Employee Management System)**
  - **WindowState=Maximized**
- Then add two more forms in the project (Form2 , Form 3)
- Create a Menu on your form (Form1) and call these two forms on menu click event.
- Then write down the following code on menu click event

```
Dim frm2 As New Form2
frm2.MdiParent = Me
frm2.Show ()
```

- Then run your MDI form.

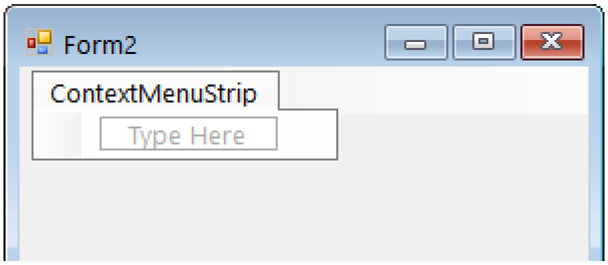
Menus

- A menu control allows **hierarchal organization of elements** associated with commands and event handlers. In a typical Microsoft Windows application, a menu bar contains several menu buttons (such as **File**, **Edit**, and **Window**), and each menu button displays a menu.
- A menu contains a collection of menu items (such as **New**, **Open**, and **Close**), which can be expanded to display additional menu items or to perform a specific action when clicked.
- **Let us add menu and sub-menu items. Perform the following steps**
  - Drag and drop or double click on a MenuStrip control, to add it to the form.
  - Click the **Type Here text** to open a text box and enter the names of the menu items or sub-menu items you want. When you add a sub-menu, another text box with 'Type Here' text opens below it.
  - Complete the menu structure shown in the diagram above.
  - Add a sub menu Exit under the File menu.

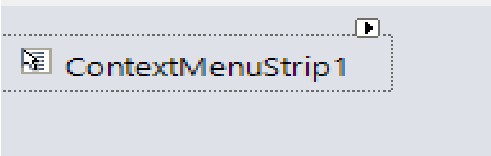


Context Menu

- Menu items which are available when you right click are called Context Menus.
- Steps of Context Menu
- Drag and drop context menu strip control to the textbox control.
- When you do, you will notice two things. At the top of your form, you will see this. And type cut copy and paste options. Then set the **ContextMenuStrip** Property of the object.



- And at the bottom of your page



-Set the ContextMenustrip Property of the object.

<u>Common Dialogue Boxes (Refer From book (Dr. Shyam N.Chawda) Page No 230)</u>

3.3. Exception Handling

Error is mistake which is found during compilation and execution of the program. It is known as Exception.

There are 3 types of errors

1) Syntax: when the syntax is wrong then we get syntax error. This error is indentifying by compiler during compilations.

2) Runtime: it occurs during program execution.

3) Logical: when there is a problem in program logic then logical error will occur. This error will raise during program execution. It can be corrected by modifying the logic of the program.

Exception Handling

It is in built mechanism in .net framework to detect and handle error. All the exceptions are directly or indirectly inherited from the **Exception class**.

-There are 2 types of exception handling

- Structured exception handling
- Unstructured exception handling

3.3.1. Structured Error Handling

Try…Catch…Finally block is used for structured exception handling.

Syntax:

```
Try
        'code to be executed
Catch
        'Exception handling
Finally
        'This code should execute, if exception occurred or
not.
End Catch
```

Try: It is used to identify the code block for which a specific exception will be activated. It is follow by one or more catch blocks.

Catch: Catching of the Exception is done in this block. It is an exception handler in which the Exception is handled.

Finally: It is used to run a set of statements whether an exception has occurred or not. It is optional.

Example:

```
Private Sub btnwithexp_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnwithexp.Click
        Try
            Dim n, m, ans As Integer
            n = TextBox1.Text
            m = TextBox2.Text
            ans = n / m
            MsgBox(ans)
        Catch ex As DivideByZeroException
            MsgBox(ex.Message)
        Catch ex As InvalidCastException
            MsgBox(ex.Message)
        Catch ex As Exception
            MsgBox(ex.Message)
        Finally
            MsgBox("Hi in finally")
        End Try
  End Sub
```

OR

```
Private Sub btnwithexp_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnwithexp.Click
        Try
            Dim n, m, ans As Integer
            n = TextBox1.Text
            m = TextBox2.Text
            ans = n / m
            MsgBox(ans)
        Catch ex As Exception
            MsgBox(ex.Message)
        Finally
            MsgBox("Hi in finally")
        End Try
  End Sub
```

3.3.2. Unstructured Error Handling

- it is implemented with **On Error GoTo** statements. The syntax is as follow.

On Error {GoTo [line | 0 | -1] | Resume Next}

GoTo line | Label—Calls the error-handling code that starts at the line specified at line. Here, line is a **line label or a line number**. If a runtime error occurs, program execution goes to the given location. **GoTo 0**—Disables the enabled error handler in the current procedure and reset it to nothing. It clears the error object.

GoTo -1—Same as GoTo 0.

Resume Next—specifies that when an exception occurs, execution skips over the statement that caused the problem and goes to the statement immediately following and Execution continues from that point.

Example :(on Error Goto Label)

```
Private Sub btngotolabel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btngotolabel.Click

        On Error GoTo Help
        MsgBox(TextBox1.Text \ TextBox2.Text)
        Exit Sub
Help:
        MsgBox("Some Error")
    End Sub
```

Example: :(on Error Goto 0)

```
Private Sub btngoto0_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btngoto0.Click

        On Error GoTo Help
        MsgBox(TextBox1.Text \ TextBox2.Text)
        Exit Sub
Help:
        On Error GoTo 0 'turn off error handling
        MsgBox("Program completed")
        Exit Sub
    End Sub
```

Example: :(Resume Next)

```
  Private Sub btnresumenext_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnresumenext.Click
        Dim ans, flag As Integer
        On Error GoTo Help
        If flag = 0 Then
            ans = 0
        End If
        MsgBox(TextBox1.Text \ TextBox2.Text)
        MsgBox(ans)
        Exit Sub
Help:
        MsgBox(Err.Number & Err.Description)
        flag = 0
        Resume Next
        Exit Sub
    End Sub
```

Err Object

When an error occurs, the **Err object** contains information about the error. This helps you to determine whether you can attempt to fix the error or ignore the error.

Properties:

Property	**Description**
Number	The error number. If this is zero then no error has occurred..
Description	A short description of the error.
HelpFile	A folder location and filename of the help file.
Source	The project name from the Properties dialog box.
HelpContext	The context ID for a particular error in a help file.

Methods:

Property	**Description**
Clear	It clears the Err Object
Raise	It raises an error.

Example:

```
Private Sub btnrerror_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnresumenext.Click
        On Error GoTo Help
        MsgBox(TextBox1.Text \ TextBox2.Text)
        Exit Sub
Help:
        MsgBox(Err.Number & Err.Description)
        Exit Sub
    End Sub
```

Note: For more Details regarding all the topics in vb.net, you might have to refer the books.