Unit-III

VB.net controls

Textbox controls

- VB.Net provides several mechanisms for gathering input in a program.
- A TextBox control is used to display, or accept as input, a single line of text.
- **VB.Net TextBox Events**
- TextBox Keydown event
- Keydown event occurs when a key is pressed while the control has focus.
- Private Sub TextBox1_KeyDown(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyEventArgs) Handles TextBox1.KeyDown
- If e.KeyCode = Keys.Enter Then
- MessageBox.Show("Enter key pressed")
- ElseIf e.KeyCode = Keys.Escape Then
- MessageBox.Show("Escape key pressed")
- End If
- End Sub

Cont...

- TextChanged Event

  TextChanged Event is raised if the Text property is changed by either through program modification or user input.

- Private Sub TextBox1_TextChanged(ByVal sender As System.Object,ByVal e As System.EventArgs) Handles TextBox1.TextChanged
- Label1.Text = TextBox1.Text
- End Sub

Properties of Textbox

- CharacterCasing :- Gets or sets whether the TextBox control modifies the case of characters as they are typed.
- Font Gets or sets the font of the text displayed by the control.
- FontHeight:- Gets or sets the height of the font of the control.
- ForeColor:- Gets or sets the foreground color of the control.
- Lines Gets or sets the lines of text in a text box control.
- Multiline Gets or sets a value indicating whether this is a multiline TextBox control.
- PasswordChar:- Gets or sets the character used to mask characters of a password in a single-line TextBox control.
- ReadOnly:- Gets or sets a value indicating whether text in the text box is read-only.

Cont...

- ScrollBars Gets or sets which scroll bars should appear in a multiline TextBox control. This property has values:
- None
- Horizontal
- Vertical
- Both
- Text Gets or sets the current text in the TextBox.
- TextAlign Gets or sets how text is aligned in a TextBox control. This property has values:
- Left
- Right
- Center

Listbox Control

- VB.Net provides several mechanisms for gathering input in a program. A Windows Forms ListBox control displays a list of choices which the user can select from.
- You can use the Add or Insert method to add items to a list box.

  ListBox1.Items.Add("Sunday")
- If you want to retrieve a single selected item to a variable , you can code like this
- **Dim var As String var = ListBox1.SelectedItem**
- If you change the selection mode property to multiple select
- **ListBox1.SelectionMode = SelectionMode.MultiSimple**

Listbox Example

- Public Class Form1
- Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
- ListBox1.Items.Add("Sunday")
- ListBox1.Items.Add("Monday")
- ListBox1.Items.Add("Tuesday")
- ListBox1.Items.Add("Wednesday")
- ListBox1.Items.Add("Thursday")
- ListBox1.Items.Add("Friday")
- ListBox1.Items.Add("Saturday")
- ListBox1.SelectionMode = SelectionMode.MultiSimple
- End Sub

Cont...

- Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

- Dim obj As Object
- For Each obj In ListBox1.SelectedItems
- MsgBox(obj.ToString)
- Next

- End Sub
- End Class

Checked ListBox Control

- The CheckedListBox control gives you all the capability of a list box and also allows you to display a check mark next to the items in the list box.
- To add objects to the list at run time, assign an array of object references with the AddRange method. The list then displays the default string value for each object.
- **Dim days As String() = {"Sunday", "Monday", "Tuesday"}**
  checkedListBox1.Items.AddRange(days)
- You can add individual items to the list with the Add method.
- **CheckedListBox1.Items.Add("Sunday", CheckState.Checked)**
  CheckedListBox1.Items.Add("Monday", CheckState.Unchecked)
  CheckedListBox1.Items.Add("Tuesday",
  CheckState.Indeterminate)

Example

- Public Class Form1

  Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
- CheckedListBox1.Items.Add("Sunday", CheckState.Checked)
- CheckedListBox1.Items.Add("Monday", CheckState.Unchecked)
- CheckedListBox1.Items.Add("Tuesday", CheckState.Indeterminate)
- CheckedListBox1.Items.Add("Wednesday", CheckState.Checked)
- CheckedListBox1.Items.Add("Thursday", CheckState.Unchecked)
- CheckedListBox1.Items.Add("Friday", CheckState.Indeterminate)
- CheckedListBox1.Items.Add("Saturday", CheckState.Indeterminate)
- End Sub

- End Class

Combo box Control

- The ComboBox control , which lets the user choose one of several choices.
- The user can type a value in the text field or click the button to display a drop down list.
- In addition to display and selection functionality, the ComboBox also provides features that enable you to efficiently add items to the ComboBox.
- **Add item to combobox**
- ComboBox1.Items.Add("Sunday")

Cont...

- **How to set the selected item in a comboBox**
- comboBox1.Items.Add("test1")
- comboBox1.Items.Add("test2")
- comboBox1.Items.Add("test3")
- comboBox1.SelectedItem = "test3"
- or
- ComboBox1.SelectedItem = ComboBox1.Items(1)
- or
- comboBox1.SelectedIndex = comboBox1.FindStringExact("test3")

Example

- Public Class Form1
- Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
- ComboBox1.Items.Add("weekdays")
- ComboBox1.Items.Add("year")
- End Sub
- Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ComboBox1.SelectedIndexChanged
- ComboBox2.Items.Clear()
- If ComboBox1.SelectedItem = "weekdays" Then

Cont...

```
ComboBox2.Items.Add("Sunday")
            ComboBox2.Items.Add("Monday")
            ComboBox2.Items.Add("Tuesday")
        ElseIf ComboBox1.SelectedItem = "year" Then
            ComboBox2.Items.Add("2012")
            ComboBox2.Items.Add("2013")
            ComboBox2.Items.Add("2014")
        End If
    End Sub
End Class
```

DateTimePicker Control

- The DateTimePicker control allows you to display and collect date and time from the user with a specified format.
- The DateTimePicker control prompts the user for a date or time using a graphical calendar with scroll arrows.
- The most important property of the DateTimePicker is the Value property, which holds the selected date and time.
- **DateTimePicker1.Value = "12/31/2010“**
- The Value property is set to the current date by default. You can use the Text property or the appropriate

Example

```
Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e
    As System.EventArgs) Handles MyBase.Load
        DateTimePicker1.Format = DateTimePickerFormat.Short
    End Sub
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal
    e As System.EventArgs) Handles Button1.Click
        Dim idate As String
        idate = DateTimePicker1.Value
        MsgBox("Selected date is  : " & idate)
    End Sub
End Class
```

GroupBox

- GroupBox control is used to group other controls of VB.NET.
- GroupBox control having a frame to indicate boundary and a text to indicate header or title.
- Generally GroupBox control is used as a container for Radio Button.
- When Radio Buttons are grouped using GroupBox, user can select one RadioButton from each GroupBox.

Properties of GroupBox Control

- BackColor
- BackgroundImage
- BackgroundImageLayout
- Font
- ForeColor
- Enabled
- Visible
- Text

ScrollBars Control

- The ScrollBar controls display vertical and horizontal scroll bars on the form. This is used for navigating through large amount of information.
- There are two types of scroll bar controls: **HScrollBar** for horizontal scroll bars and **VScrollBar** for vertical scroll bars.
- These are used independently from each other.

Properties of the ScrollBar Control

- 1 AutoSize Gets or sets a value indicating whether the ScrollBar is automatically resized to fit its contents.
- 2 BackColor Gets or sets the background color for the control.
- 3 ForeColor Gets or sets the foreground color of the scroll bar control.
- 4 ImeMode Gets or sets the Input Method Editor (IME) mode supported by this control.
- 5 LargeChange Gets or sets a value to be added to or subtracted from the Value property when the scroll box is moved a large distance.

Cont...

- 6 Maximum Gets or sets the upper limit of values of the scrollable range.
- 7 Minimum Gets or sets the lower limit of values of the scrollable range.
- 8 SmallChange Gets or sets the value to be added to or subtracted from the Value property when the scroll box is moved a small distance.
- 9 Value Gets or sets a numeric value that represents the current position of the scroll box on the scroll bar control.

Example

- The Scroll event fires when the user changes the control's value interactively:
- Private Sub HScrollBar1_Scroll(sender As Object, e As ScrollEventArgs) _
- Handles HScrollBar1.Scroll
- YearLabel.Text = HScrollBar1.Value.ToString()
- End Sub

Set any value for maximum and minimum property

- HScrollBar1.Maximum = 2050
- HScrollBar1.Minimum = 1960

Example of Hscrollbar and Vscrollbar

```
Public Class Form1
    Private Sub Form1_Load(ByVal sender As Object, ByVal e As EventArgs) _
      Handles MyBase.Load
       HScrollBar1.Minimum = 20
       HScrollBar1.Maximum = 200
       VScrollBar1.Minimum = 20
       VScrollBar1.Maximum = 200
       Label1.Text = "(" & VScrollBar1.Value & "," & HScrollBar1.Value & ")"
       Label1.Location = New Point(HScrollBar1.Value, VScrollBar1.Value)
    End Sub
    Private Sub VScrollBar1_Scroll(ByVal sender As System.Object, ByVal e As
     System.Windows.Forms.ScrollEventArgs) Handles VScrollBar1.Scroll
       Label1.Text = "(" & VScrollBar1.Value & "," & HScrollBar1.Value & ")"
       Label1.Location = New Point(HScrollBar1.Value, VScrollBar1.Value)
    End Sub
```

Cont...

```
    Private Sub HScrollBar1_Scroll_1(ByVal sender As
    System.Object, ByVal e As
    System.Windows.Forms.ScrollEventArgs) Handles
    HScrollBar1.Scroll
        Label1.Text = "(" & VScrollBar1.Value & "," &
    HScrollBar1.Value & ")"
        Label1.Location = New Point(HScrollBar1.Value,
    VScrollBar1.Value)
    End Sub
End Class
```

Trackbar

- The TrackBar control allows you to **drag a pointer** along a bar to select a **numeric** value.
- The control's **Value, Minimum, Maximum, and TickFrequency** properties are **integer** values, so the TrackBar control is **not ideal** for letting the user select a non-integral value such as 5.34.
- Its Scroll event fires when the user changes the control's value interactively.
- Private Sub TrackBar1_Scroll(sender As Object, e As EventArgs) _
- Handles TrackBar1.Scroll
- VolumeLabel.Text = String.Format("Volume: {0}", TrackBar1.Value)
- End Sub

Example

- Public Class Form2
- Private Sub TrackBar1_Scroll(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles TrackBar1.Scroll
- Label1.Text = String.Format("Volume: {0}", TrackBar1.Value)
- End Sub
- Private Sub TrackBar1_ValueChanged(ByVal sender As Object, ByVal e As System.EventArgs) Handles TrackBar1.ValueChanged
- If TrackBar1.Value = 50 Then
- msgbox("The Volume Has Been Maxed Out!")
- Else
- msgbox("The volume has been set to: " & TrackBar1.Value)
- End If
- End Sub
- End Class

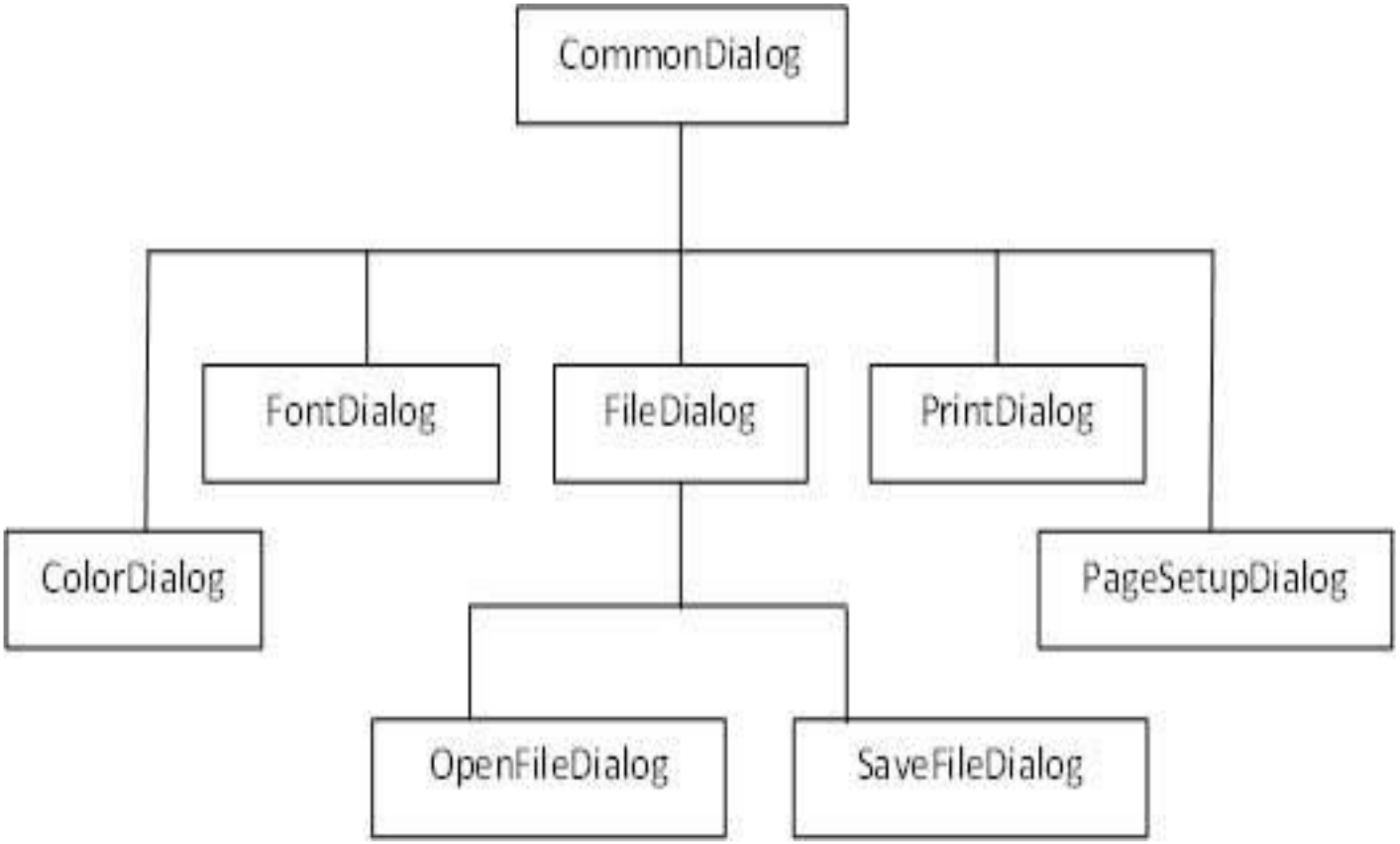
Common Dialog Control

- There are many built-in dialog boxes to be used in Windows forms for various tasks like opening and saving files, printing a page, providing choices for colors, fonts, page setup, etc., to the user of an application.
- These built-in dialog boxes reduce the developer's time and workload.
- All of these dialog box control classes inherit from the **CommonDialog** class .
- The **ShowDialog** method is used to display all the dialog box controls at run-time.

It returns a value of the type of **DialogResult** enumeration. The values of DialogResult enumeration are:

- **Abort** - returns DialogResult.Abort value, when user clicks an Abort button.
- **Cancel**- returns DialogResult.Cancel, when user clicks a Cancel button.
- **Ignore** - returns DialogResult.Ignore, when user clicks an Ignore button.
- **No** - returns DialogResult.No, when user clicks a No button.
- **None** - returns nothing and the dialog box continues running.
- **OK** - returns DialogResult.OK, when user clicks an OK button
- **Retry** - returns DialogResult.Retry , when user clicks an Retry button
- **Yes** - returns DialogResult.Yes, when user clicks an Yes button

The following diagram shows the common dialog class inheritance



VB.Net - ColorDialog Control

- The ColorDialog control class represents a common dialog box that displays available colors along with controls that enable the user to define custom colors. It lets the user select a color.
- The main property of the ColorDialog control is *Color*, which returns a **Color** object.
- Example:-

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
    Button1.Click
    If ColorDialog1.ShowDialog <> Windows.Forms.DialogResult.Cancel Then
    Label1.ForeColor = ColorDialog1.Color
    End If
End Sub
```

OpenFile Dialog Box

- The **OpenFileDialog** control prompts the user to open a file and allows the user to select a file to open.
- The user can check if the file exists and then open it.
- The OpenFileDialog control class inherits from the abstract class **FileDialog**.

Example:-

```
Private Sub Button1_Click(ByVal sender As Object, ByVal e As EventArgs) Handles Button1.Click
    If OpenFileDialog1.ShowDialog <> Windows.Forms.DialogResult.Cancel Then
        PictureBox1.Image = Image.FromFile(OpenFileDialog1.FileName)
        Dim fileName As String
        fileName = OpenFileDialog1.FileName
        MsgBox(fileName)
    End If
End Sub
```

VB.Net - SaveFileDialog Control

- The **SaveFileDialog** control prompts the user to select a location for saving a file and allows the user to specify the name of the file to save data.
- The SaveFileDialog control class inherits from the abstract class FileDialog.

Example:-

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
    SaveFileDialog1.Filter = "TXT Files (*.txt*)|*.txt"
    If SaveFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK _
  Then
        My.Computer.FileSystem.WriteAllText _
        (SaveFileDialog1.FileName, RichTextBox1.Text, True)
    End If
  End Sub
```

PrintDialog Control

- The PrintDialog control lets the user to print documents by selecting a printer and choosing which sections of the document to print from a Windows Forms application.
- There are various other controls related to printing of documents.

  These other controls are:
- **PrintDocument** control
- **PrinterSettings**
- **PageSetUpDialog**
- **PrintPreviewControl**
- **PrintPreviewDialog**

Example

- Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
- PrintDialog1.Document = PrintDocument1
- PrintDialog1.PrinterSettings = PrintDocument1.PrinterSettings
- PrintDialog1.AllowSomePages = True
- If PrintDialog1.ShowDialog = DialogResult.OK Then
- PrintDocument1.PrinterSettings = PrintDialog1.PrinterSettings
- PrintDocument1.Print()
- End If
- End Sub

FontDialog Control

- It prompts the user to choose a font from among those installed on the local computer and lets the user select the font, font size, and color. It returns the Font and Color objects.
- By default, the Color ComboBox is not shown on the Font dialog box. You should set the **ShowColor** property of the FontDialog control to be **True**.

Example:-

- Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
- If FontDialog1.ShowDialog <> Windows.Forms.DialogResult.Cancel Then
-
- RichTextBox1.Font = FontDialog1.Font
- End If
- End Sub

The RichTextBox Control

- The **RichTextBox** control is the core of a full-blown word processor. It provides all the functionality of a TextBox control.
- It can handle multiple typefaces, sizes, and attributes, and offers precise control over the margins of the text.
- You can even place images in your text on a RichTextBox control.
- The fundamental property of the **RichTextBox** control is its Rtf property.
- RTF, which stands for Rich Text Format, is a standard for storing formatting information along with the text.

Listview

- The ListView control is used to display a list of items.
- The Item property of the ListView control allows you to add and remove items from it.
- The *SelectedItem* property contains a collection of the selected items.
- The *MultiSelect* property allows you to set select more than one item in the list view.
- The *CheckBoxes* property allows you to set check boxes next to the items.

Example:-

- Private Sub Form4_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
- ListView1.Columns.Add("Emp Name", 100, HorizontalAlignment.Left)
- ListView1.Columns.Add("Emp Address", 150, HorizontalAlignment.Left)
- ListView1.Columns.Add("Title", 60, HorizontalAlignment.Left)
- ListView1.Columns.Add("Salary", 50, HorizontalAlignment.Left)
- ListView1.Columns.Add("Department", 60, HorizontalAlignment.Left)
- ListView1.View = View.Details
- End Sub

Cont...

- Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
- Dim str(5) As String
- Dim itm As ListViewItem
- str(0) = TextBox1.Text
- str(1) = TextBox2.Text
- str(2) = TextBox3.Text
- str(3) = TextBox4.Text
- str(4) = TextBox5.Text
- itm = New ListViewItem(str)
- ListView1.Items.Add(itm)
- End Sub

TreeView control

- TreeView control is used to display hierarchical tree like information such as a directory hierarchy.
- The top level in a tree view are root nodes that can be expanded or collapsed if the nodes have child nodes.
- The user can expand the TreeNode by clicking the plus sign (+) button.
- When a parent node is expanded, its child nodes are visible.
- The fullpath method of treeview control provides the path from root node to the selected node.

  TreeView1.SelectedNode.FullPath

Example:-

- Private Sub Form5_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
- Dim tNode As TreeNode
- tNode = TreeView1.Nodes.Add("Department of Computer science")

- TreeView1.Nodes(0).Nodes.Add("PGDCA")
- TreeView1.Nodes(0).Nodes(0).Nodes.Add("VB.net")

- TreeView1.Nodes(0).Nodes.Add("MscIT")
- TreeView1.Nodes(0).Nodes(1).Nodes.Add("JAVA")
- TreeView1.Nodes(0).Nodes(1).Nodes.Add("PHP")

Cont...

- TreeView1.Nodes(0).Nodes.Add("CA&IT")
- TreeView1.Nodes(0).Nodes(2).Nodes.Add("JAVA")
- TreeView1.Nodes(0).Nodes(2).Nodes(0).Nodes.Add("CORE java")
- TreeView1.Nodes(0).Nodes(2).Nodes(0).Nodes.Add("ADV Java")
- End Sub

- Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
- MsgBox(TreeView1.SelectedNode.FullPath)
- End Sub