Working with other controls of toolbox:

DateTime Picker

- If you want to **enable users to select a date and time**, and **to display that date and time in the specified format**, use the DateTimePicker control.
- If you click the arrow in the DateTimePicker control, it displays a month calendar, like a combo box control.
- The user can make selection by clicking the required date. The new selected value appears in the text box part of the control. The **MinDate** and the **MaxDate** properties allow you to put limits on the date range.

Properties:

Property Name	Description
MaxDate	It gets or sets the maximum date and time that can be selected in the control.
MinDate	It gets or sets the minimum date and time that can be selected in the control.
Format	It gets or sets the format of the date and time displayed in the control.
Font	It gets or sets the font of the text displayed by the control.
Value	It gets or sets the date/time value assigned to the control.
ShowCheckBox	It gets or sets a value indicating whether a check box is displayed to the left of the selected date.
ShowUpDown	It gets or sets a value indicating whether a spin button control (also known as an up-down control) is used to adjust the date/time value.
Visible	It gets or sets a value indicating whether the control and all its child controls are displayed.
DropDownAlign	It gets or sets the alignment of the drop-down calendar on the DateTimePicker control.

Methods:

Method Name	Description
ToString()	It returns the string representing the control.

Events:

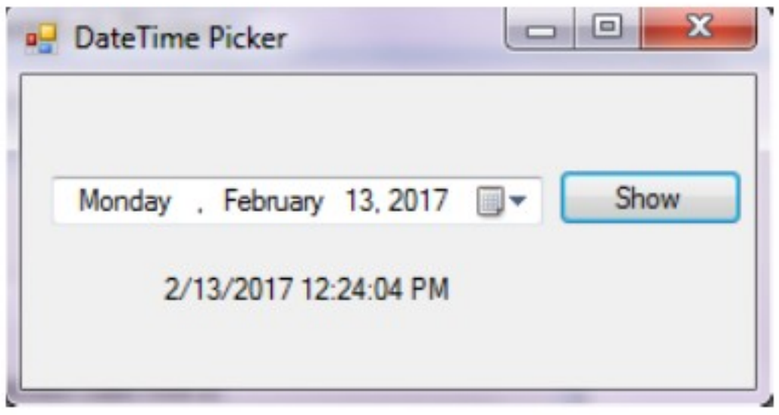
Event Name	Description
CloseUp	It occurs when the drop-down calendar is dismissed and disappears.
FontChanged	It occurs when the Font property value changes.
FormatChanged	It occurs when the Format property value has changed.
ValueChanged	It occurs when the Value property changes.
VisibleChanged	It occurs when the Visible property value changes.
TabIndexChanged	It occurs when the TabIndex property value changes.

Example:

```
Private Sub btnShow_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnShowMessage.Click
    lbldate.Text = slttimepicker.Value
```

End Sub

Output:



List Box

- The ListBox represents a Windows control to display a list of items to a user.
- A user can select an item from the list. It allows the programmer to add items at design time by using the properties window or at the runtime.

Properties:

Property Name	Description
Items	It gets the items of the list box.
MultiColumn	It gets or sets a value indicating whether the list box supports multiple columns.
ScrollAlwaysVisible	It gets or sets a value indicating whether the vertical scroll bar is shown at all times.
SelectedIndex	It gets or sets the zero-based index of the currently selected item in a list box.
SelectedItem	It gets or sets the currently selected item in the list box.
SelectedValue	It gets or sets the value of the member property specified by the ValueMember property.
SelectionMode	It gets or sets the method in which items are selected in the list box. This property has values: None, One, MultiSimple, and MultiExtended
Sorted	It gets or sets a value indicating whether the items in the list box are sorted alphabetically.

Methods:

Method Name	Description
ClearSelected()	Unselects all items in the ListBox.
FindString()	Finds the first item in the ListBox that starts with the string specified as an argument.
ToString()	Returns a string representation of the ListBox.
OnSelectedIndexChanged()	Raises the SelectedIndexChanged event.
OnSelectedValueChanged()	Raises the SelectedValueChanged event.

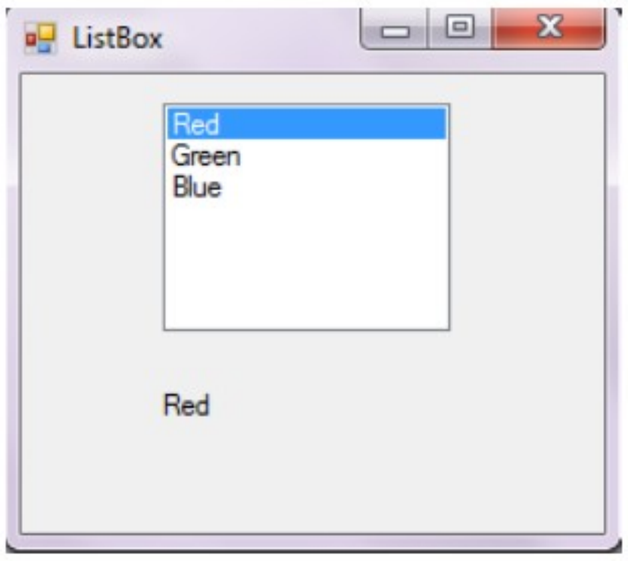
Events:

Event Name	Description
Click	Occurs when a list box is selected.
SelectedIndexChanged	Occurs when the SelectedIndex property of a list box is changed.

Example:

```
Private Sub ListBox1_SelectedIndexChanged(ByVal sender As Object, ByVal e As EventArgs)
Handles ListBox1.SelectedIndexChanged
        Label1.Text = ListBox1.SelectedItem.ToString()
End Sub
```

Output:



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.
- To add items to a ComboBox, it will opens the String Collection Editor Dialog box, where you can enter the values.

Properties:

Property Name	Description
DataSource	Gets or sets the data source for this ComboBox.
Items	Gets an object representing the collection of the items contained in this ComboBox.
SelectedIndex	Gets or sets the index specifying the currently selected item.
SelectedItem	Gets or sets currently selected item in the ComboBox.
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.
Text	Gets or sets the text associated with this control.

Methods:

Method Name	Description
FindString()	Finds the first item in the combo box that starts with the string specified as an argument.
FindStringExact()	Finds the first item in the combo box that exactly matches the specified string.
SelectAll ()	Selects all the text in the editable area of the combo box.

Events:

Event Name	Description
DropDown	Occurs when the drop-down portion of a combo box is displayed.
DropDownClosed	Occurs when the drop-down portion of a combo box is no longer visible.
DropDownStyleChanged	Occurs when the DropDownStyle property of the ComboBox has changed.
SelectedIndexChanged	Occurs when the SelectedIndex property of a ComboBox control has changed.
SelectionChangeCommitted	Occurs when the selected item has changed and the change appears in the combo box.

Example:

```
Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As Object, ByVal e As EventArgs)
Handles ComboBox1.SelectedIndexChanged
        Label1.Text = ComboBox1.SelectedItem.ToString()
End Sub
```

Output:

Picture Box

- The PictureBox control is **used for displaying images on the form**.
- The **Image property** of the control allows you to set an image **either at design time or at run time**.

Properties:

Property Name	Description
AllowDrop	Specifies whether the picture box accepts data that a user drags on it.
ErrorImage	It gets or specifies an image to be displayed when an error occurs during the image-loading process or if the image load is cancelled.
Image	It gets or sets the image that is displayed in the control.
ImageLocation	It gets or sets the path or the URL for the image 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, StrechImage, AutoSize, CenterImage, and Zoom**.

Methods:

Method Name	Description
CancelAsync()	Cancels an asynchronous image load.
Load()	Displays an image in the picture box.
LoadAsync()	Loads image asynchronously.

ToString()	Returns the string that represents the current picture box.

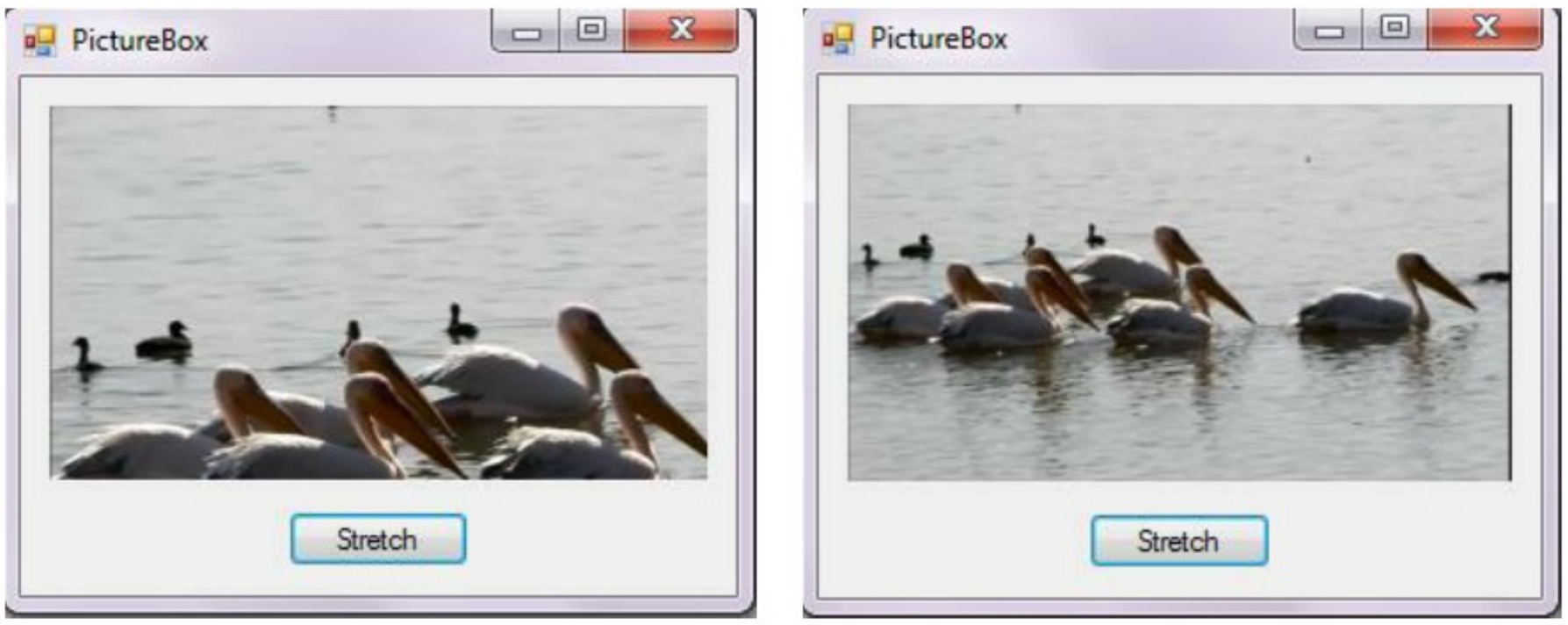
Events:

Event Name	Description
Click	Occurs when the control is clicked.
Enter	Overrides the **Control.Enter** property.
KeyPress	Occurs when a key is pressed when the control has focus.
KeyDown	Occurs when a key is pressed when the control has focus.
Resize	Occurs when the control is resized.
SizeModeChanged	Occurs when SizeMode changes.

Example:

```
Private Sub btnStretch_Click((ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnStretch.Click
        PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
End Sub
```

Output:



(a). Normal Image **(b). Stretch Image**

Rich Text Box

- The RichTextBox control **enables you to display or edit flow content including paragraphs, images, tables, and more.**
- 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.

Properties:

Property Name	Description
AcceptsTab	It gets or sets a value that indicates how the text editing control responds when the user presses the TAB key.
BackColor	It gets or sets the background color of the control.
ForeColor	It gets or sets the foreground color of the control.
MaxLength	It gets or sets the maximum number of characters the user can type or paste into

	the rich text box control.
Multiline	It gets or sets a value indicating whether this is a multiline RichTextBox control.
ScrollBars	It gets or sets the type of scroll bars to display in the RichTextBox control.
ReadOnly	It gets or sets a value indicating whether text in the text box is read-only.

Methods:

Method Name	Description
Clear()	Clears all text from the text box control.
Copy()	Copies the current selection in the text box to the **Clipboard**
Cut()	Moves the current selection in the text box to the **Clipboard**.
Select()	Activates the control.
Undo()	Undoes the last edit operation in the text box.

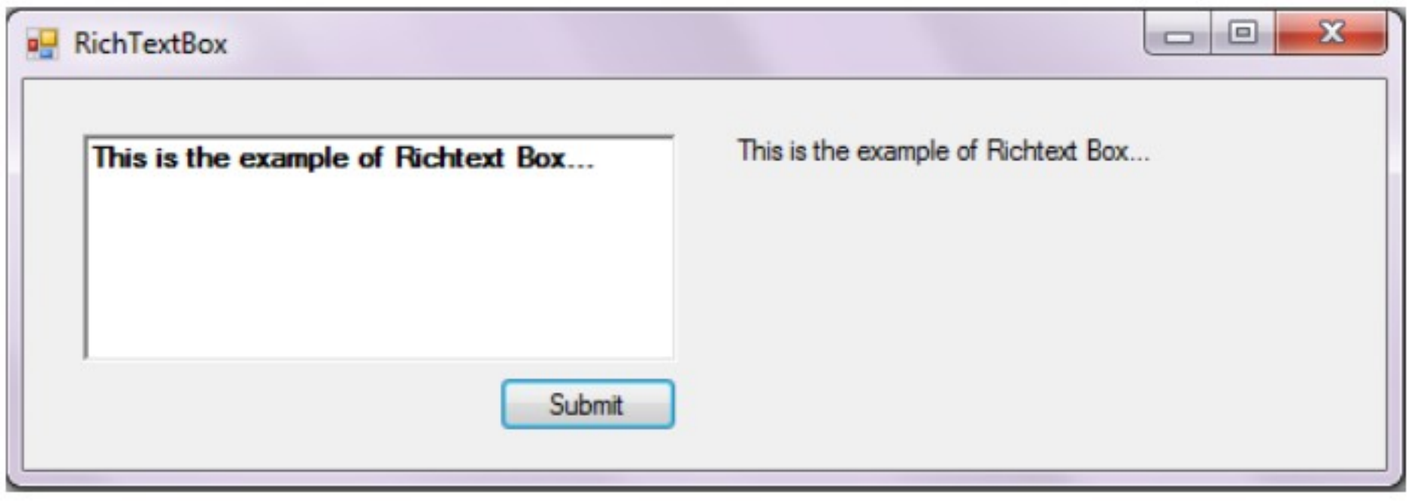
Events:

Event Name	Description
Click	Occurs when the control is clicked.
VScroll	Occurs when content scroll vertically by mouse in the text element.
HScroll	Occurs when content scroll horizontally by mouse in the text element.
TextChanged	Occurs when content changes in the text element.
LinkClicked	Occurs when the user clicks on the link within the text.
SelectionChanged	Occurs when the value of the Selection property is changed.

Example:

```
Private Sub btn_save_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btn_save.Click
        Label1.Text = RichTextBox1.Text
End Sub
```

Output:



Progress bar

- The ProgressBar control visually indicates the progress of a lengthy **operation such as calculating a complex result, downloading a large file from the Web,** printing documents, **etc.**
- It shows a bar that fills in from **left to right** as the operation progresses.
- The main properties of a progress bar are **Value, Maximum and Minimum**.
- The **Value** property specifies the **current position of the progress bar.**

- The **Minimum and Maximum properties** are used to set the minimum and maximum values that the **progress bar can display**.

Properties:

Property Name	Description
Value	It gets or sets the current position of the progress bar.
Step	It gets or sets the amount by which a call to the PerformStep method increases the current position of the progress bar.
Style	It gets or sets the manner in which progress should be indicated on the progress bar.
Maximum	It gets or sets the maximum value of the range of the control.
Minimum	It gets or sets the minimum value of the range of the control.

Methods:

Method Name	Description
Increment()	Increments the current position of the ProgressBar control by specified amount.
PerformStep()	Increments the value by the specified step.
ResetText()	Resets the Text property to its default value.
ToString()	Returns a string that represents the progress bar control.

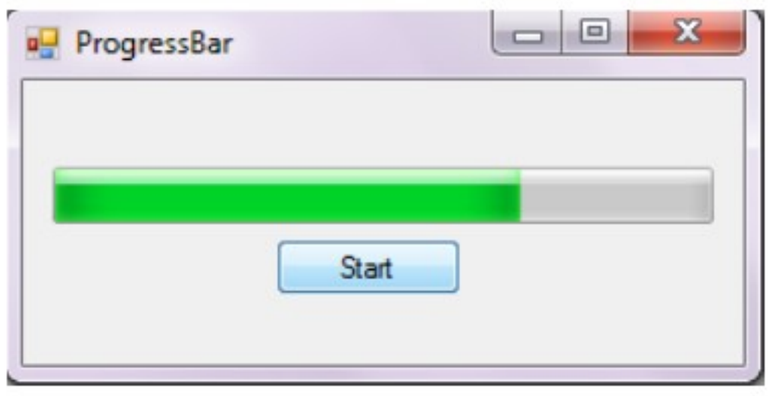
Events:

Event Name	Description
Click	Occurs when the control is clicked.
Enter	Occurs when focus enters the control.
KeyDown	Occurs when the user presses a key while the control has focus.
KeyUp	Occurs when the user releases a key while the control has focus.
TextChanged	Occurs when the Text property changes.

Example:

```
Private Sub btn_start_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btn_start.Click
    Dim i As Integer
    For i = 0 To 100 Step 1
        ProgressBar1.Value = i
        Threading.Thread.Sleep(100)
    Next
End Sub
```

Output:



Masked Text Box

- It provides a mask that **helps the user in entering a value in a particular format.**
- The mask determines which characters are allowed at different positions in the text, displaying placeholder characters to help prompt the user and underscores where the user can enter characters.

Properties:

Property Name	Description
Size	It specifies the size of the control.
Name	It specifies a unique name of the control.
Text	It specifies a current text of the control.
Mask	It represents the format of the input can be accepted by a control.
ReadOnly	It represent that if you want the text box only readable mode, so no one can perform editing on that textbox.
PromptChar	It gets or sets the character used to represent the absence of user input in MaskedTextBox.

Methods:

Method Name	Description
Clear()	Clears all text from the text box control.
Hide()	Conceals the control from the user.
Select()	Activates the control.
ToString()	Returns a string that represents the current masked text box.

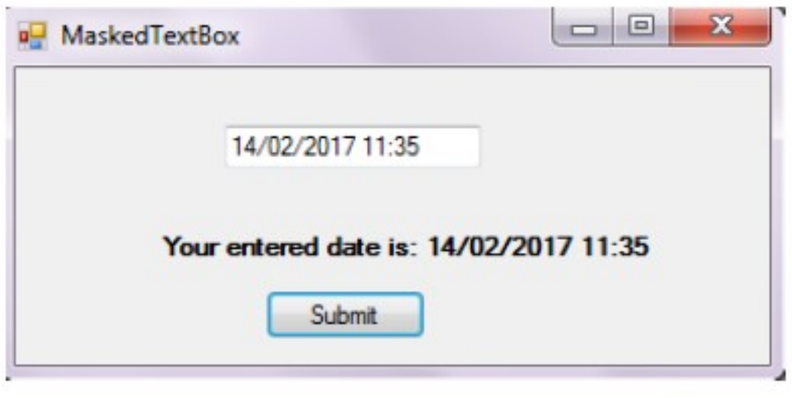
Events:

Event Name	Description
Click	Occurs when the text box is clicked.
MaskChanged	Occurs after the input mask is changed.
Resize	Occurs when the control is resized.
Enter	Occurs when the control is entered.

Example:

```
Private Sub btn_click_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btn_click.Click
        Label1.Text = "Your entered date is: "
        Label1.Text &= MaskedTextBox1.Text
End Sub
```

Output:



Link Label

- A LinkLabel control is a **label control that can display a hyperlink**.
- A LinkLabel control is **inherited from the Label class** so it has all the functionality provided by the Windows Forms Label control.
- LinkLabel control does not participate in user input or capture mouse or keyboard events.

Properties:

Property Name	Description
Links	It gets the collection of links contained within the LinkLabel.
LinkColor	It gets or sets the color used when displaying a normal link.
LinkArea	It gets or sets the range in the text to treat as a link.
LinkVisited	It gets or sets a value indicating whether a link should be displayed as though it were visited.
LinkBehavior	It gets or sets a value that represents the behavior of a link.
Image	It gets or sets the image that is displayed on a Label.
ActiveLinkColor	It gets or sets the color used to display an active link.
VisitedLinkColor	It gets or sets the color used when displaying a link that that has been previously visited.

Methods:

Method Name	Description
ToString()	Returns a string that represents the current Label
ResetText()	Resets the Text property to its default value.
Focus()	Sets input focus to the control.

Events:

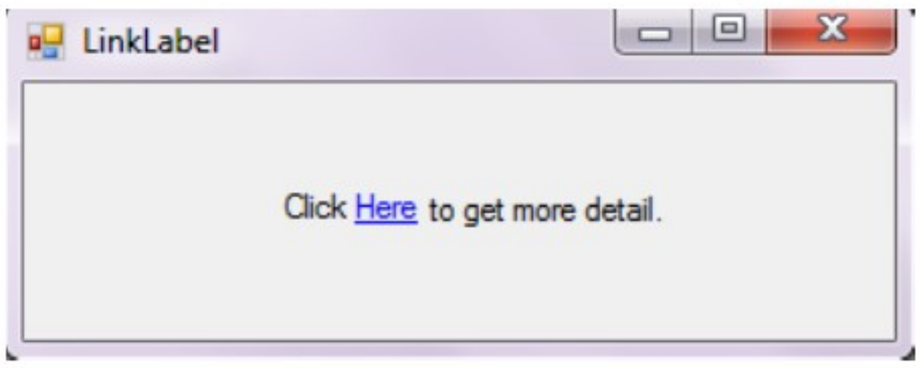
Event Name	Description
LinkClicked	Occurs when a link is clicked within the control.
TextChanged	Occurs when the Text property value changes.
Click	Occurs when the control is clicked.

Example:

```
Private Sub LinkLabel1_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles LinkLabel1.LinkClicked
        System.Diagnostics.Process.Start("http://www.yahoo.com")
End Sub
```

Output:

:



Checked List box

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

Properties:

Property Name	Description
CheckOnClick	It gets or sets a value indicating whether the check box should be toggled when an item is selected.
ColumnWidth	It gets or sets the width of columns in a multicolumn ListBox.
Items	It gets the collection of items in this CheckedListBox.
MultiColumn	It gets or sets a value indicating whether the ListBox supports multiple columns.
ScrollAlwaysVisible	It gets or sets a value indicating whether the vertical scroll bar is shown at all times.
SelectedIndex	It gets or sets the zero-based index of the currently selected item in a ListBox.
SelectedItem	It gets or sets the currently selected item in the ListBox.
Sorted	Displays the list of the content in alphabetic order.

Methods:

Method Name	Description
ClearSelected()	Method used to unselect the items in the checked list box.
FindString()	Method used to find the string in the first item of the control.
GetItemText()	Method used to get the text of an item.

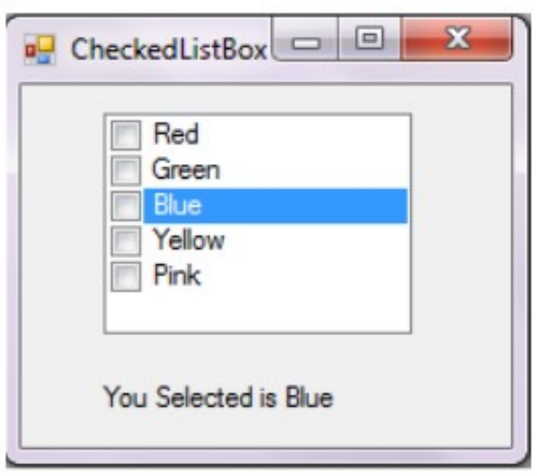
Events:

Event Name	Description
ItemCheck	Occurs when the checked state of an item changes.
SelectedIndexChanged	Triggered when the SelectedIndex property is changed

Example:

```
Private Sub CheckedListBox1_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CheckedListBox1.SelectedIndexChanged
        Label1.Text = "You Selected is " & CheckedListBox1.SelectedItem.ToString()
End Sub
```

Output:



Scroll Bars

- 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:

Property Name	Description
LargeChange	It gets or sets a value to be added to or subtracted from the Value property when the scroll box is moved a large distance.
Maximum	It gets or sets the upper limit of values of the scrollable range.
Minimum	It gets or sets the lower limit of values of the scrollable range.
SmallChange	It gets or sets the value to be added to or subtracted from the Value property when the scroll box is moved a small distance.
Value	It gets or sets a numeric value that represents the current position of the scroll box on the scroll bar control.

Methods:

Method Name	Description
OnClick()	Generates the Click event.
Select()	Activates the control.

Events:

Event Name	Description
Click	Occurs when the control is clicked.
Scroll	Occurs when the control is moved.
ValueChanged	Occurs when the Value property changes, either by handling the Scroll event or programmatically.

Example:

```
Private Sub HScrollBar1_Scroll(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.ScrollEventArgs) Handles HScrollBar1.Scroll
        Label2.Text = HScrollBar1.Value.ToString()
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.ToString()
End Sub
```

Output:

Timer

- Timer Control is **used to set time intervals**, this control is visible only at design time and not in the runtime.
- You can change the properties of Timer as needed, right click on it and click on Properties. If you set Enabled to **True**, Timer will start ticking as soon as the form loads. **Default is False**.
- Interval is the **frequency of Elapsed events in milliseconds (1000 = 1 second).**

Properties:

Property Name	Description
Enabled	Property used to Get or set whether the timer is running.
Interval	Gets or sets the interval, expressed in milliseconds.

Methods:

Method Name	Description
Start()	Method used to start timer.
Stop()	Method used to stop timer.

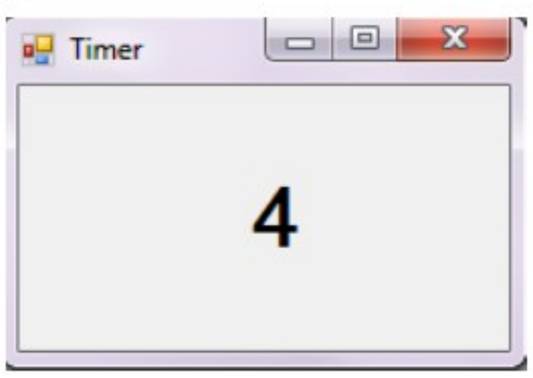
Events:

Event Name	Description
Tick	Triggered when the time interval has elapsed.

Example:

```
Dim i As Integer = 0
Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Timer1.Tick
        Label1.Text = i.ToString()
        i = i + 1
End Sub
```

Output:



Working with Menus

- A menu contains a **list of related commands**.
- A menu will require in application so that you can perform actions such as **opening child forms, copying and pasting data, and arranging windows, etc.**
- The **MenuStrip control** represents the container for the menu structure (see Fig 3.1).
- The MenuStrip control works as the **top-level container for the menu structure**.
- The **ToolStripMenuItem** class and the **ToolStripDropDownMenu** class provide the functionalities to create menu items, sub menus and drop-down menus.

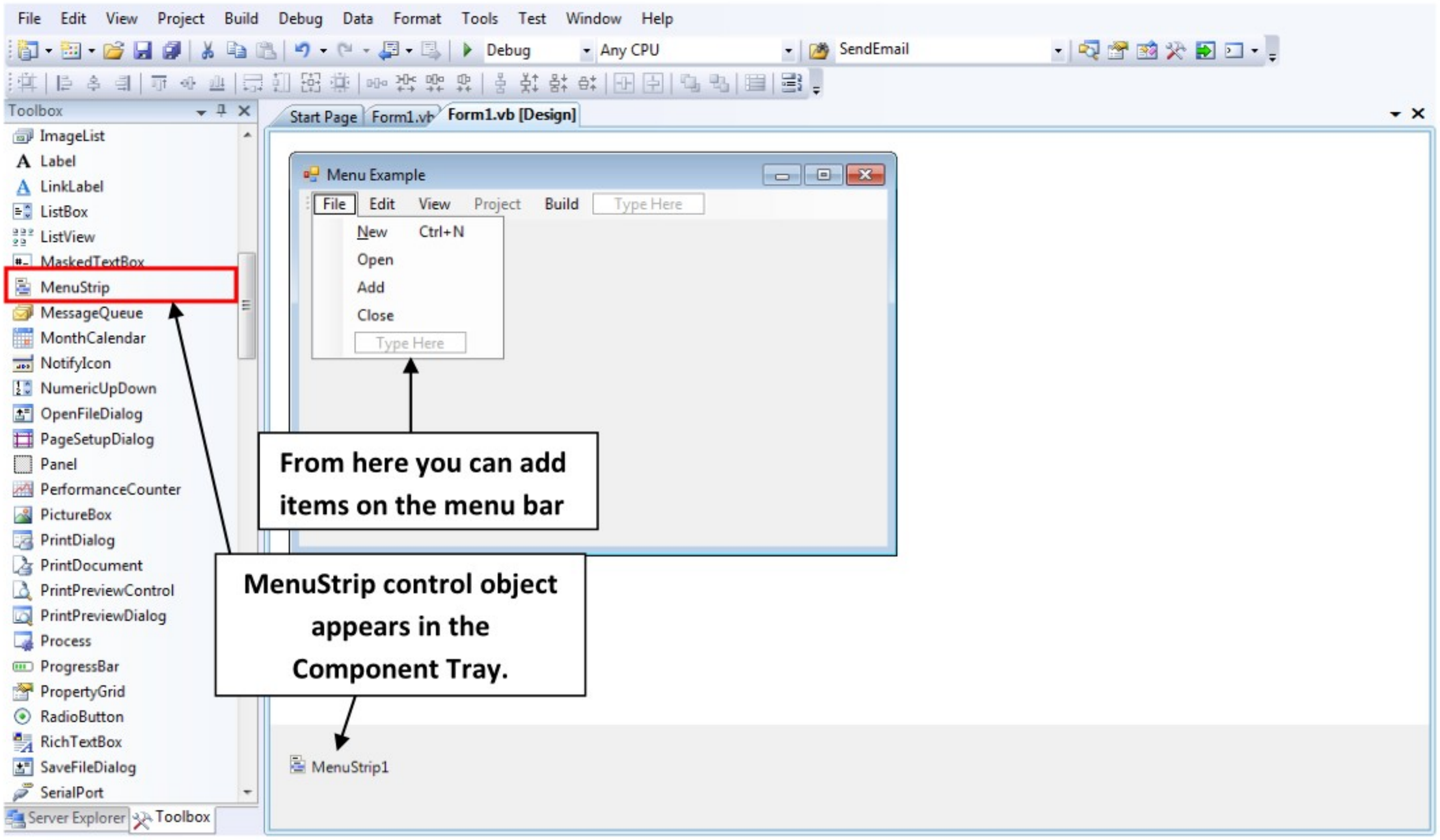


Fig 3.1 MenuStrip control on the form

Properties:

Property Name	Description
Items	It gets all the items that belong to a ToolStrip.
GripStyle	It gets or sets the visibility of the grip used to reposition the control.
ShowItemToolTips	It gets or sets a value indicating whether ToolTips are shown for the MenuStrip.

Events:

Event Name	Description
Click	Occurs when the control is clicked.
ItemClicked	Occurs when the ToolStripItem is clicked.
MenuActivate	Occurs when the user accesses the menu with the keyboard or mouse.
MenuDeactivate	Occurs when the MenuStrip is deactivated.

Add menu and sub-menu items:

- Following are the steps to add menu and sub menu in your application:
  - **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 Fig 3.2.
  - Add a sub menu **Close** under the **File menu**.

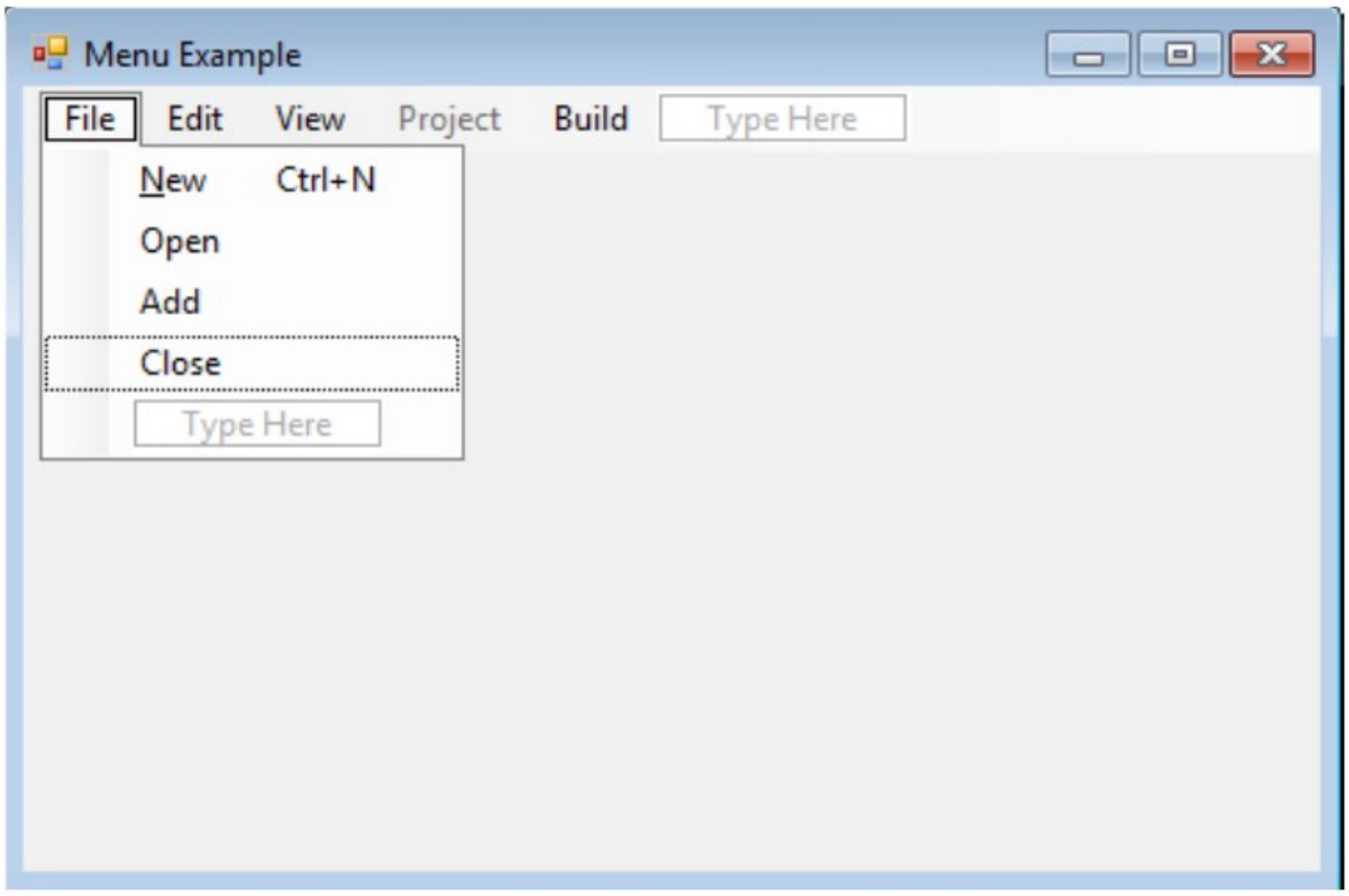


Fig 3.2 Menu Example

  - **Double-Click on the Close menu which we created** and add the following code to the Click event of **CloseToolStripMenuItem.**

Example :

```
Private Sub CloseToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CloseToolStripMenuItem.Click
        End
End Sub
```

Delete menu and sub-menu items

- Following are the steps to delete menu and sub menu from your application:
  - To delete a menu and sub menu from application, just **Right click** on the **Menuitem** which you want to delete.
  - It will open a one dialog box which is shown in Fig 3.3.
  - Click on the **Delete** option from the opened menu to delete a particular **Menuitem** from the menu.

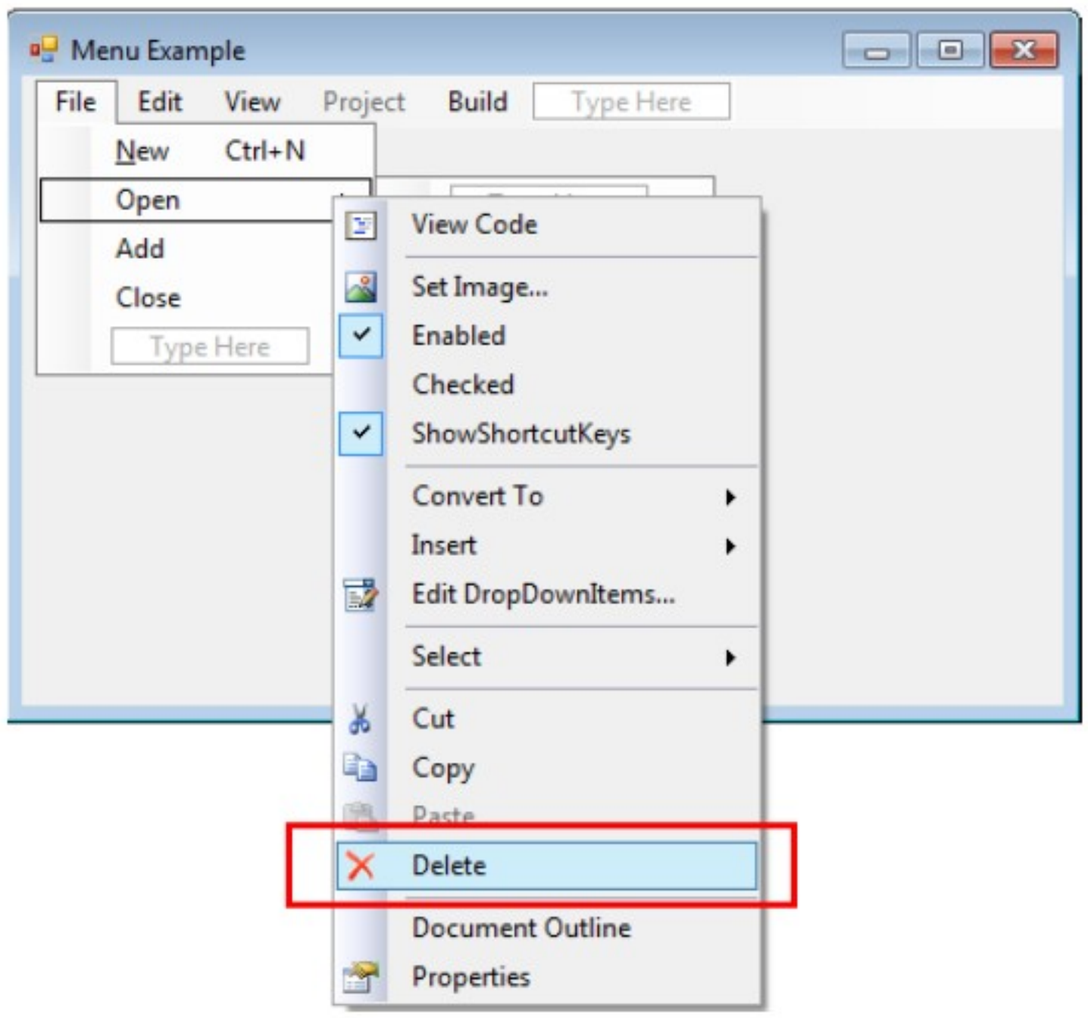


Fig 3.3 Delete Menuitem from Menu

Assigning shortcut keys

- A key combination shortcut is one that appears at the end of a menu item (Ctrl + X, for example). You can easily add this option to your own programs. Following are the steps to assign shortcut key to menuitem:
  - In Design (See Fig 3.2), select the **Close** item on your created menu and look at the **properties box** on the right side.
  - Locate the **ShortcutKeys** item on the properties box (See Fig 3.4).
  - The Modifier is the key you press with your shortcut. For example, the CTRL key then the "X" key on your keyboard. Place a check inside the Ctrl box. Then select the letter "X" from the Key dropdown list.

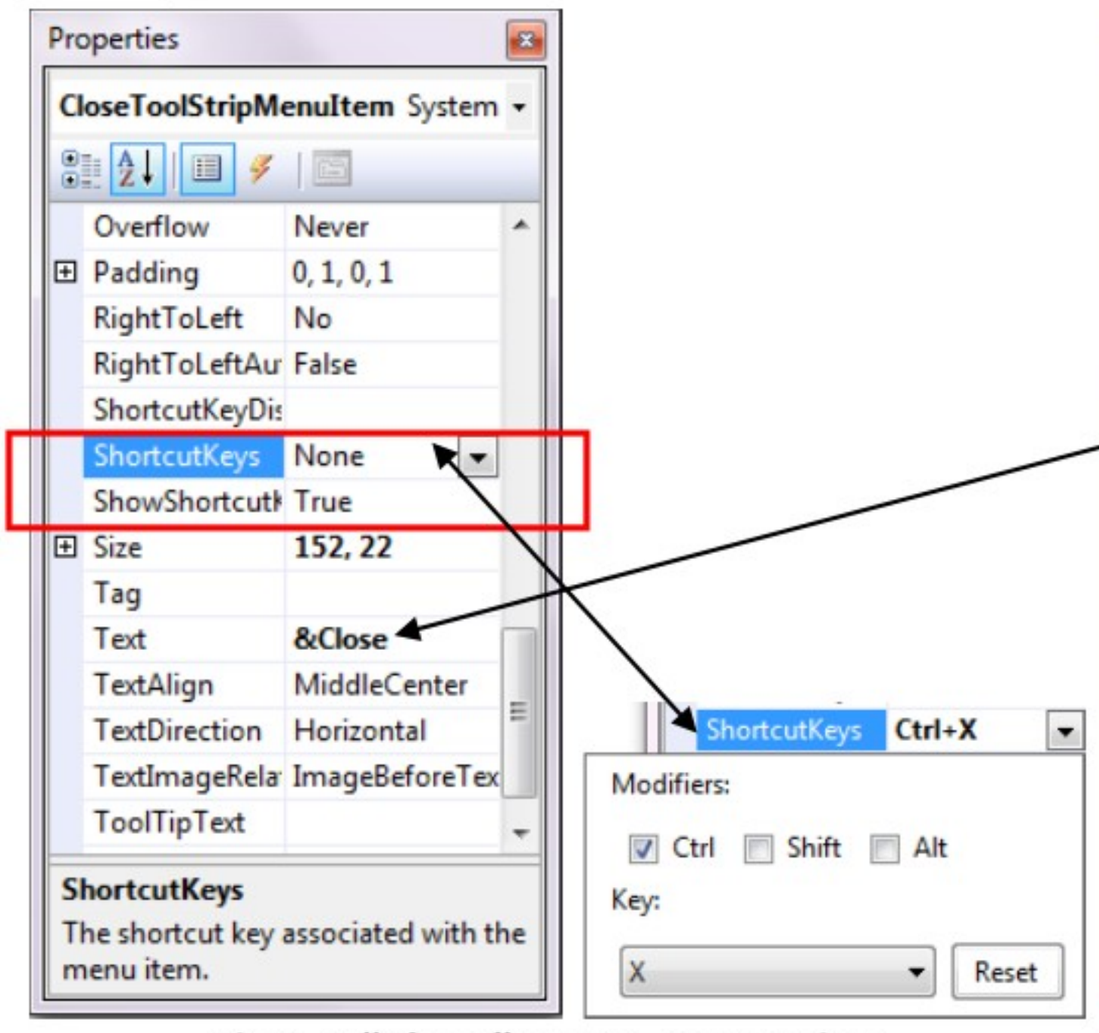


Fig 3.4 "Close" menu properties

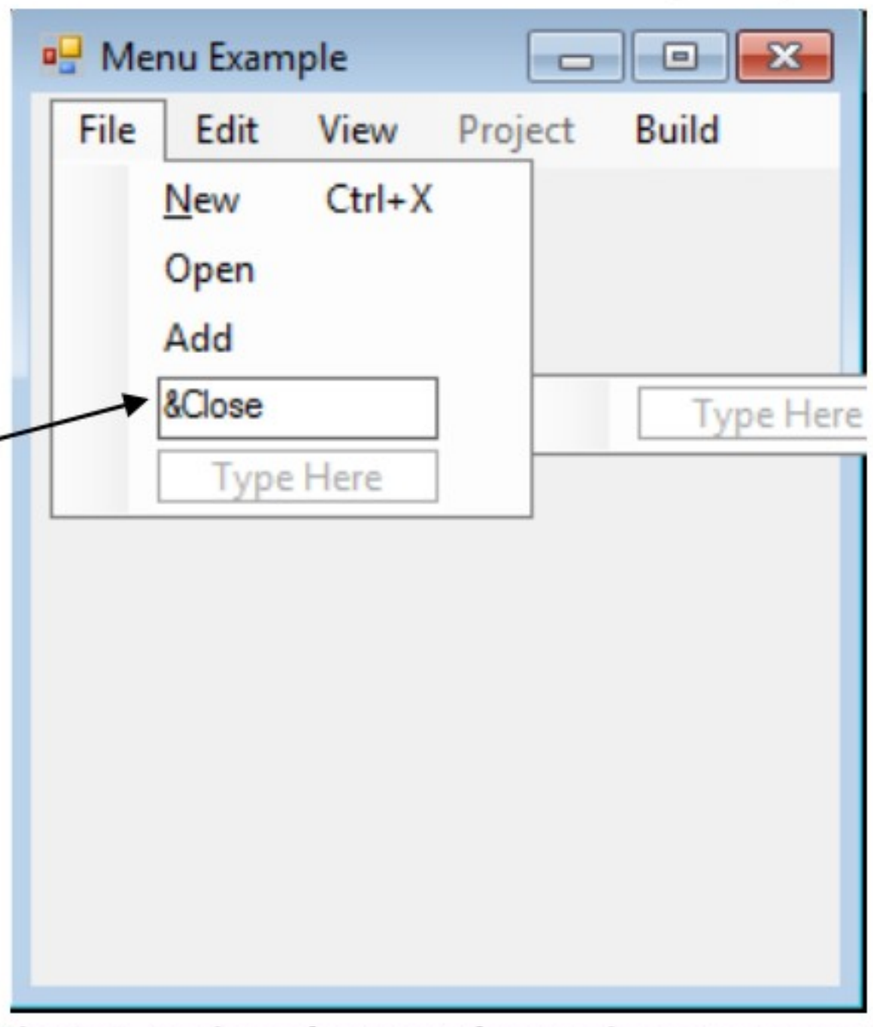


Fig 3.5 Assign shortcut key using ampersand

- **Another way to add shortcut to menuitem**:
  - To add a new menu or edit existing one, just click the menu entry and type.
  - You can set a **shortcut** by placing an **ampersand** in front of the character you want to use as a keyboard accelerator e.g. **&Close (See Fig 3.5)**. Now if type **Alt+C** at runtime, the program closes itself.

Popup Menu:

- A **PopUp** menu is a menu that is **displayed when user right click either on a form or on a control**. It is also **known as Context Menu or Floating Menu.**
- A **PopUp** menu is useful to **represents list of commonly used commands for form or control**.
- In order to create a **PopUp** menu in your application, The **ContextMenuStrip** control which represents a shortcut menu that pops up over controls, usually when you right click them. They appear in context of some specific controls, so are called **context menus.** For example, **Cut, Copy or Paste options.**
- To create a Content/Popup menu with the menu options **Cut, Copy and Paste.** Do the following steps:
  - **Drag and drop or double click** on a **ControlMenuStrip** control to add it to the form.
  - **Add** the menu items, Cut, Copy and Paste to it.
  - **Add** a RichTextBox control on the form.
  - **Set** the **ContextMenuStrip** property of the **RichTextBox** to **ContextMenuStrip1** using the properties window.
  - Double click on the menu items and add following codes in the **Click** event of these menus:

```
Private Sub CutToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CutToolStripMenuItem1.Click
        RichTextBox1.Cut()
End Sub
Private Sub CopyToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CopyToolStripMenuItem1.Click
        RichTextBox1.Copy()
End Sub
Private Sub PasteToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles PasteToolStripMenuItem1.Click
        RichTextBox1.Paste()
End Sub
```

Output: