. Chapter 24

Adding Menus and Toolbar

- ❖ Learning how to Create Menus and Menu Items
- ❖ Learning how to Create Toolbar Items

Menus and toolbars remain as the standard features of all windows applications despite the development of more sophisticated GUI. The menu bar contains menus, which contain groups of menu items that the user can used to execute certain commands to perform certain tasks like opening a file, saving a file, printing a page, formatting a page and more. On the other hand, a standard toolbar displays icons that can be used to open a file, save a file, viewing a document, printing a document and more.

In this chapter, we will show you how to add Menus and icons to the toolbar of your applications. We will use the text editor from the chapter 19 but now we shall execute the commands using the menus and the toolbar icons. We shall also make this program more powerful by enabling it to format the text as well as to print out the text from the text file.

In this project, we will add MenuStrip1, ToolStrip1, SaveFileDialog1, OpenFileDialog1,PrintDialog1 and FontDialog1 controls to the form.

24.1 Adding Menus

Open the text editor file from the chapter 19, but now we will clear the buttons and add menus instead. First, drag the Menu Strip and position it at the top part of the form. Add the first top-level menu by typing it in the textbox that appears with a blurred text “Type Here”. The first menu you will add is File, but you type it with the ampersand sign in front, like this, &File. The reason is the ampersand sign will underline the letter F, File at runtime so that the user can use the keyboard short-cut keys to execute a command. The second top-level menu that we shall add is Format, which we type it as &Format.

The next step is to add menu items to the File and the Format Menu. The three menu items that we are going to add to the File menu are Open, Save, Print and Exit, type them as &Open, &Save, &Print and E&xit. The menu items that we will add to the Format menu are Font (type it as Fo&nt), Font Color (type it as Font &Color) and Background Color (type it as &Background Color). The menu items can be moved upward or downward easily by dragging them. They can be deleted easily by pressing the right mouse button and then click deleted in the pop-up dialog.

When we run the finished design, we shall see a window application that comprises menus and menu items, as shown in Figure 241. Notice the underlined characters of the menu items.

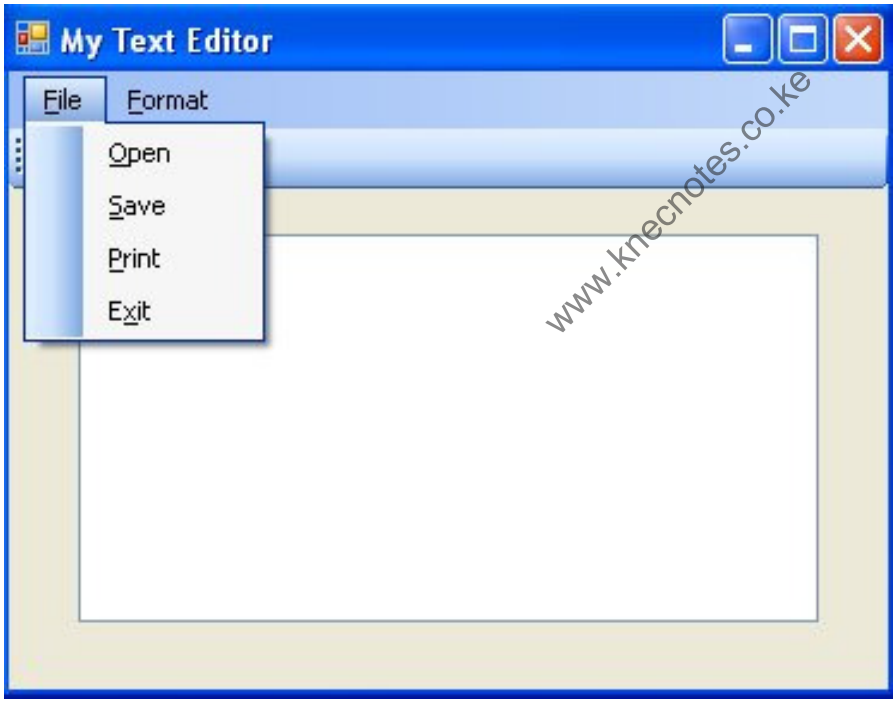

Figure 241

24.1.1 Writing Code for the Menu Items

The application in the preceding section is not able to do anything yet until we write code for the menu items.

The menu item **Open** should execute a command that will allow the user to choose a file from a storage source and open it via a pop-up dialog. The code is the same as the

code to read text file in the previous chapter. It involves the use of the OpenFileDialog control. Now, double click on the Open menu item and enter the code as follows:

```
Private Sub OpenToolStripMenuItem_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles OpenToolStripMenuItem.Click
    Dim FileReader As StreamReader
    Dim results As DialogResult
    results = OpenFileDialog1.ShowDialog
    If results = DialogResult.OK Then
       FileReader = New StreamReader(OpenFileDialog1.FileName)
       TxtEditor.Text = FileReader.ReadToEnd( )
       FileReader.Close( )
    End If
  End Sub
```


Remember place the statement Imports System.IO before Public Class Form1 so that the program is able to read the file. The open dialog is shown in Figure 24.2

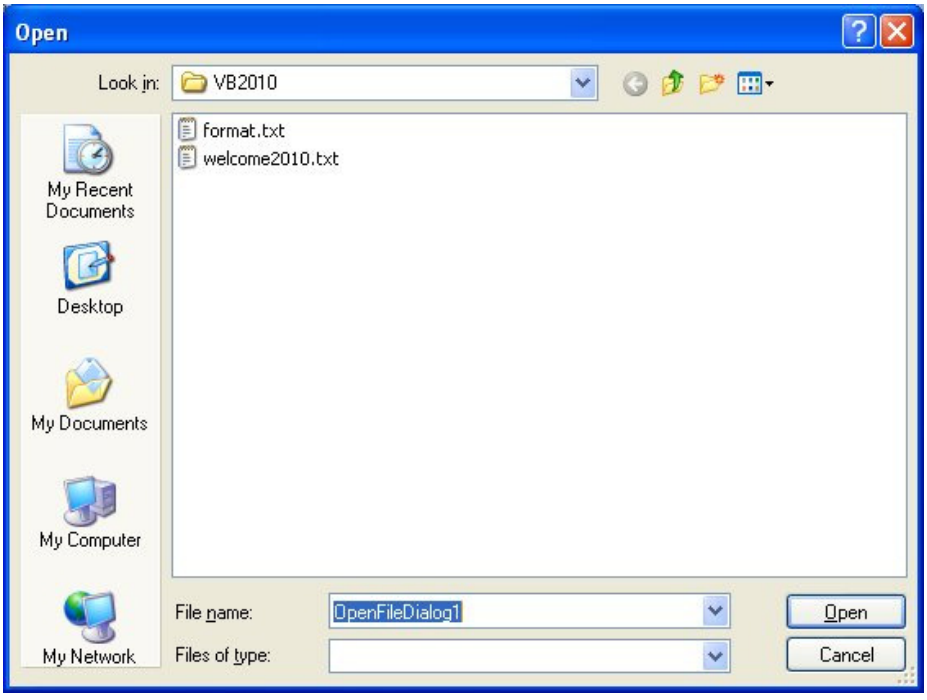

Figure 24.2

Menu item **Save** executes command that writes file to the computer storage unit. The code is the same as the code for writing file in the previous chapter. Click on the Save menu item and enter the following code:

```
Private Sub SaveToolStripMenuItem_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles SaveToolStripMenuItem.Click
    Dim FileWriter As StreamWriter
    Dim results As DialogResult
    results = SaveFileDialog1.ShowDialog
    If results = DialogResult.OK Then
        FileWriter = New StreamWriter(SaveFileDialog1.FileName, False)
        FileWriter.Write(TxtEditor.Text)
        FileWriter.Close()
    End If
End Sub
```


Writing code for the Print command requires the use of the PrintDialog control. It comprises two parts, the first part is to presents a print dialog for the user to set the options to print and second part is to print the document. Click on the print menu item and enter the following code:

i) The code to presents a print dialog

```
Private Sub PrintToolStripMenuItem_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles PrintToolStripMenuItem.Click
    'Let the user to choose the page range to print.
    PrintDialog1.AllowSomePages = True
    'Display the help button.
    PrintDialog1.ShowHelp = True
    PrintDialog1.Document = docToPrint
```

```
    Dim result As DialogResult = PrintDialog1.ShowDialog()
    If (result = DialogResult.OK) Then
        docToPrint.Print()
    End If

End Sub
```

ii) The code to print the document

```
Private Sub document_PrintPage(ByVal sender As Object, _
  ByVal e As System.Drawing.Printing.PrintPageEventArgs) _
    Handles docToPrint.PrintPage
    Dim mytext As String
    mytext = TxtEditor.Text
    Dim printFont As New System.Drawing.Font _
      ("Arial", 12, System.Drawing.FontStyle.Regular)

    ' Format and print the text
    e.Graphics.DrawString(mytext, printFont, _
       System.Drawing.Brushes.Black, 10, 10)
End Sub
```


24.2 Adding Toolbar Icons

Still using the same file, we shall now add some toolbar items in form of icons. You can lookup for some free icons sites in Google to download the icons you intend to place on your toolbar. In our example, we need six icons namely the Open icon, the Save icon, the Print icon, the Font Style and Formatting icon, the Font Color icon and the Background Color icon.

To add items to the toolbar, click on the small icon on the leftmost corner of the toolbar and choose button from the dropdown list, as shown in Figure 25.3

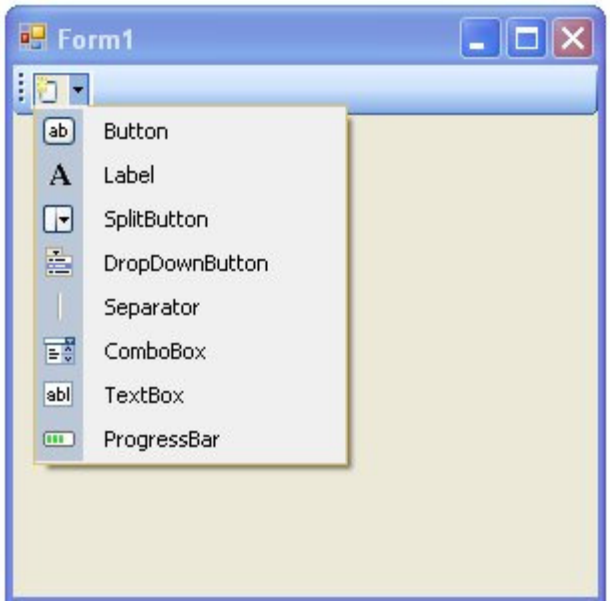

Figure 24.3

Right click on the button and choose properties window from the dropdown list, then proceed to change the default image by clicking the three-dot button on the right of the image property. Choose an icon or image file from your hard drive that you wish to load, as shown in Figure 24.4 and Figure 24.5

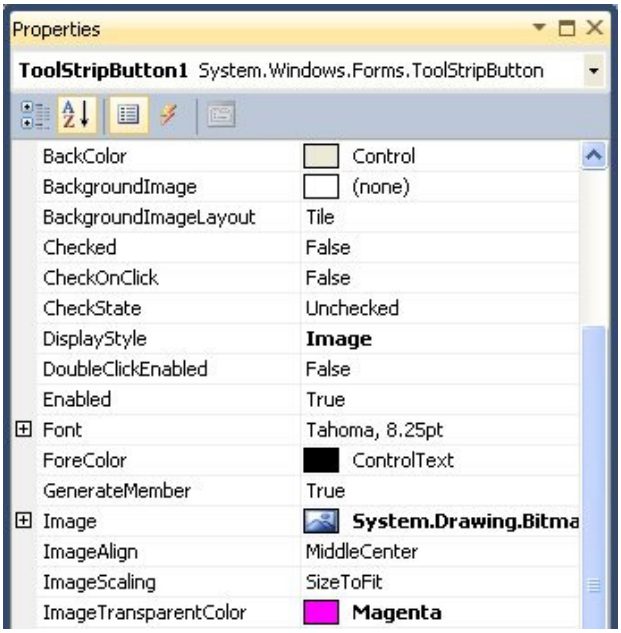

Figure 24.4: Properties window of the ToolStrp Button

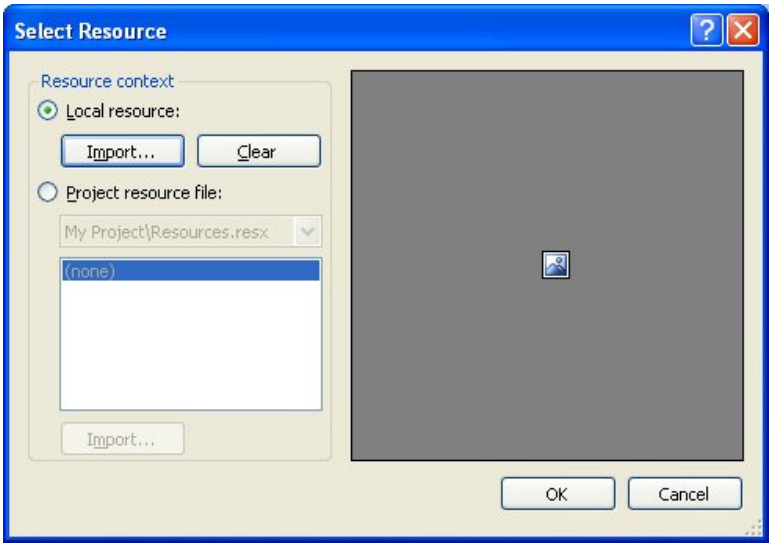

Figure 24.5: Dialog to select image file

Using the aforementioned method, we have added the following toolbar items and set their properties as shown in Table 24.1. The ToolTipText is to display text when the user places his or her mouse over the toolbar icon. The purpose is to provide information about the action that can be executed by clicking the icon.

Toolbar Item	Name	ToolTipText
	ToolOpen	Open
	ToolSave	Save
	ToolPrint	Print
	ToolFontStyle	Font Style and Formatting
	ToolFontColor	Font Color
	ToolBkColor	Background Color

Table 24.1

The finished interface is shown in Figure 24.6

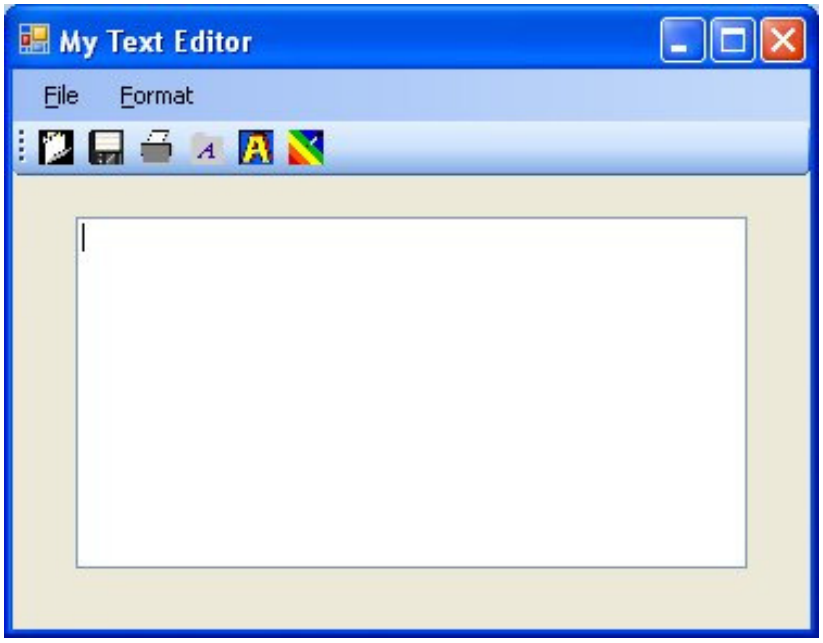

Figure 24.6

Next, we shall write code for every item on the tool bar. The codes are the same as the codes we programmed for the menu items.

Open Folder	

The Code:

```
Private Sub ToolOpen_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToolOpen.Click
    Dim FileReader As StreamReader
    Dim results As DialogResult
    results = OpenFileDialog1.ShowDialog
    If results = DialogResult.OK Then
       FileReader = New StreamReader(OpenFileDialog1.FileName)
       TxtEditor.Text = FileReader.ReadToEnd()
       FileReader.Close()
    End If
  End Sub
```

Save File	

The Code:

```
Private Sub ToolSave_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToolSave.Click
    Dim FileWriter As StreamWriter
    Dim results As DialogResult
    results = SaveFileDialog1.ShowDialog
    If results = DialogResult.OK Then
        FileWriter = New StreamWriter(SaveFileDialog1.FileName, False)
        FileWriter.Write(TxtEditor.Text)
        FileWriter.Close()
    End If
  End Sub
```


Print	

The Code

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

    PrintDialog1.AllowSomePages = True

    PrintDialog1.ShowHelp = True
    PrintDialog1.Document = docToPrint
    Dim result As DialogResult = PrintDialog1.ShowDialog()

    If (result = DialogResult.OK) Then
       docToPrint.Print()
    End If


  End Sub
```

Format Font Style	

The Code

```
Private Sub ToolFontStyle_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToolFontStyle.Click
    FontDialog1.ShowColor = True
    FontDialog1.Font = TxtEditor.Font
    FontDialog1.Color = TxtEditor.ForeColor
    If FontDialog1.ShowDialog() <> DialogResult.Cancel Then
        TxtEditor.Font = FontDialog1.Font
        TxtEditor.ForeColor = FontDialog1.Color
    End If
  End Sub
```


Font Color	

The Code

```
 Private Sub ToolFontColor_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToolFontColor.Click
    Dim MyDialog As New ColorDialog()
    MyDialog.AllowFullOpen = False
    MyDialog.ShowHelp = True
    MyDialog.Color = TxtEditor.ForeColor
    If (MyDialog.ShowDialog() = Windows.Forms.DialogResult.OK) Then
        TxtEditor.ForeColor = MyDialog.Color
    End If

 End Sub
```

Background Color	

The Code

```
Private Sub ToolBkColor_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToolBkColor.Click
    Dim MyDialog As New ColorDialog()
    MyDialog.AllowFullOpen = False
    MyDialog.ShowHelp = True
    MyDialog.Color = TxtEditor.BackColor
     If (MyDialog.ShowDialog() = Windows.Forms.DialogResult.OK) Then
       TxtEditor.BackColor = MyDialog.Color
    End If
  End Sub
```


To test the program, press F5 to run it. Enter the Text “Welcome to Visual Basic 2010 programming” into the text editor, then use the menu items or the toolbar icons to change the font size to 14 ,font color to yellow and the background color to blue. Run the program and you will see the menus and toolbar icons appear on top of the text editor, as shown in Figure 24.7

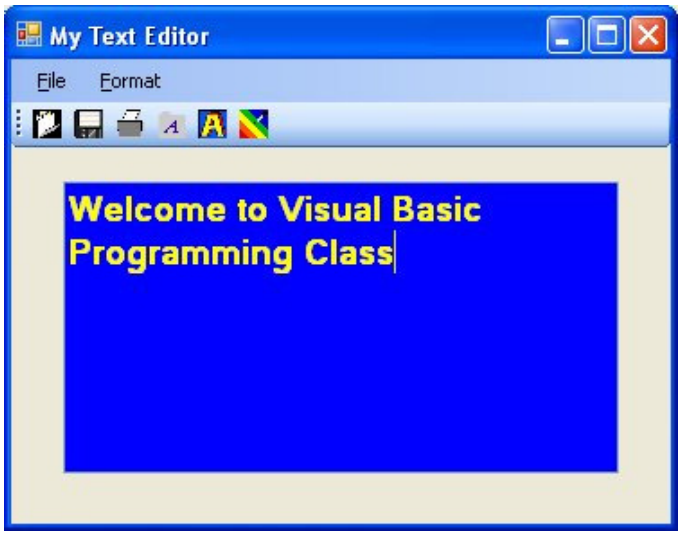

Figure 24.7

Summary

In this chapter, you learned how to add menus and toolbar to your application.

- In section 24.1, you learned how to add menus to your application, a text editor. You also learned how to write code for the menus. Besides, you learned to write code for printing the text.
- In section 24.2, you learned how to add toolbar icons and write code for them.