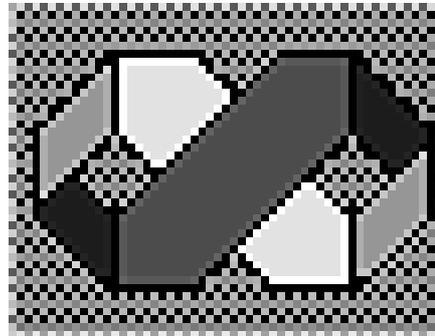


Chapter

10

**Web
Applications**



Introduction

Earlier versions of Visual Basic were excellent for creating applications which ran on a Windows PC, but increasingly there is a great demand for applications which run on the Web within a Web browser. Web pages originally consisted of simple HTML and were static, that is they did not allow you to read information and to link to other pages, but that was all. The development of new technologies such as ASP (Active Server pages) and JSP (Java Server pages) allowed users to interact with Web pages, to produce the applications which are now common on the Web. In this chapter we are going to look at:

- What is ASP.NET?
- How to create Web applications.

In the next chapter we will look in more detail at the Web server controls

What is ASP.NET

When an interactive Web application runs, you are prompted for information, for example by selecting an item from a list or typing in your name or credit card details. This information is sent to the server, which examines the information you have supplied, perhaps references a database and then creates an HTML page which is sent back to you. The server may also send some DHTML (Dynamic HTML) back which contains an application which runs locally on your computer without the need to refer back to the server.

One of the methods of creating dynamic interactive Web pages is to use ASP. Visual Basic .NET uses Microsoft's extended version of ASP called ASP.NET.

ASP.NET has some important advantages over ASP, in particular:

- The Visual Basic .NET framework provides an excellent interactive user interface for rapid application development and debugging.
- ASP uses interpreted languages such as VBScript. ASP.NET uses fully compiled Visual Basic .NET which executes much faster.
- ASP.NET has a far more extensive set of controls, in particular server-side controls.

Before we look at how to develop Web applications we need consider some of the differences compared to Windows based applications.

Windows and Web applications

Many of the techniques used to develop Windows applications can be used for developing Web applications, however there are some important differences:

- You know far less about the environment in which the applications is running when writing a Web application compared to a Windows application..
- The majority of the code for Web applications is on the Web server rather than the computer which is used to view the application.
- The time taken to respond to events will be far longer in Web applications because of the time taken to communicate with the server. This may not be significant, however if a server is being heavily used, it may be sufficient for users to notice that there is a delay. You should bear this in mind when developing Web applications and minimise the interactivity.
- Animations may not be satisfactory because the need to refer to the server, but this can be

overcome to some extent by using DHTML which runs on the local machine.

- Different browsers have different capabilities and this may cause problems particularly with a new product such as this. If you are developing a commercial application you should ensure that your applications run satisfactorily with the most popular browsers.

These problems are not insignificant, however there is one major advantage to developing Web based applications. If you are developing commercial applications, it is easy to update the application – you just change the single copy on the server and it is automatically available to all users. This can be a huge saving in cost if the application has many users since updated versions and bug-fixes do not need to be sent to every customer.

What do I need to develop Web applications?

If you want to develop Web applications you need a Web server supports ASP.NET such as IIS (Internet Information Server), with the required extensions installed. IIS is supplied with Windows 2000 Server and Windows XP Pro (but not the home edition).

Earlier versions of Windows such as 98 and ME support PWS (Personal Web Server), but are not suitable for creating and deploying Web applications.

Creating a Web application

To create a Web project select the **File | New | Project** menu option as shown in figure 10.1.

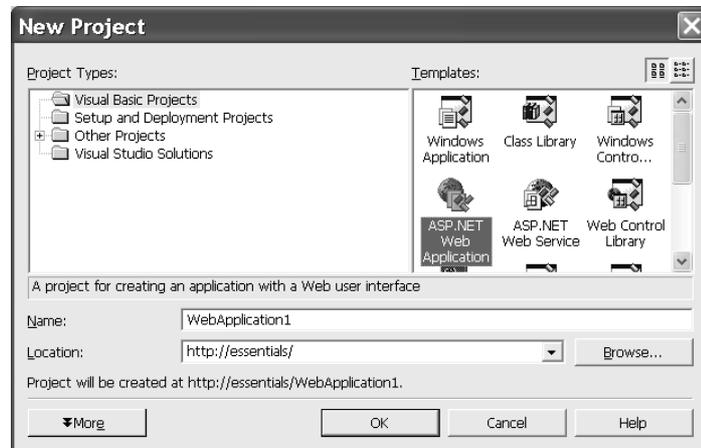


Figure 10.1 Creating a Web application.

You must specify the name of the application and the server it is to be created on, in this case a local server called *essentials*. Click the **OK** button.

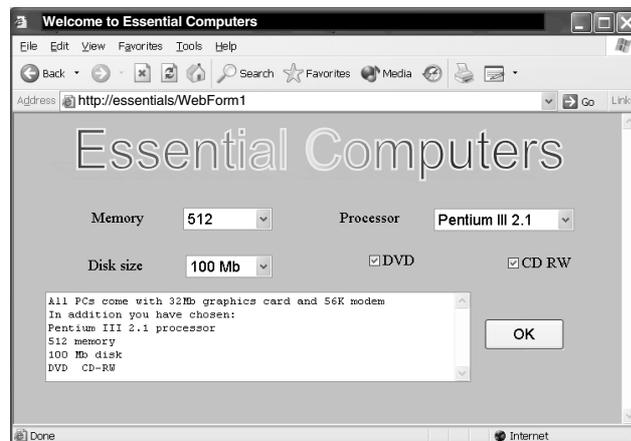


Figure 10.2 The running application.

The running application we are going to develop is shown in figure 10.2.

It has an **Image** control which provides the graphic at the top of the screen and a combination of controls

which allow you to choose from a number of options and so specify the configuration of a computer you may wish to buy. When you click on the *OK* button, a summary of the items you have selected is displayed. Most of the elements of creating a Web application are very similar to those used for Windows controls, however there are some important differences therefore we are going to look in detail at every stage of creating this application.

Creating the application

After creating your Web application, the Solution Explorer should be similar to that shown in figure 10.3. To display the design form double click on the default name of the Web form *WebForm1*.



Figure 10.3 The Solution Explorer.

The controls are added in the same way as for a Windows application, but note that the toolbox provides a different page of controls specifically for Web applications.

- Add an **Image** control to the Form and change its **ImageUrl** property to point to the graphic you wish to display.
- Add the **Label** controls and change their **Text** properties to *Memory*, *Processor* and *Disk size* as appropriate.

- Add the three **DropDownList** controls and change their names to *lstMemory*, *lstprocessor* and *lstDiskSize* as appropriate.
- Add the two **CheckBox** controls and change their **Text** properties to *DVD* and *CD RW* and their names to *chkDVD* and *chkCDRW*.
- Add the **TextBox** control and change its **Text** property to be blank and its name to *txtSummary*. Change the **TextMode** property to **MultiLine**.
- Add the **Button** and change its **Text** property to **OK** and its name to *btnOK*.
- Set the background colour of the form to an attractive colour using the **bgColor** property of the form.

The next is to add the list items to the **ListBox** controls. This is done by selecting each of these controls in turn and clicking on the button at the right of the **Items** property to display the ListItem Collection Editor as shown in figure 10.4.

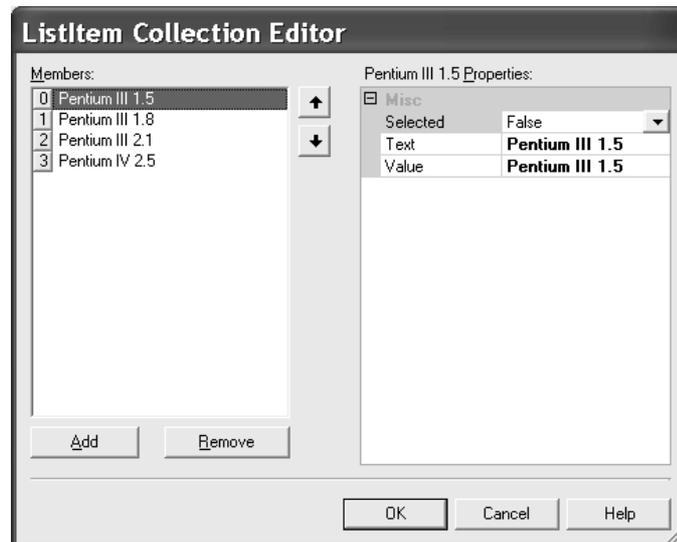


Figure 10.4 The ListItem Collection Editor.

List items are added or deleted using this dialog by clicking the **Add** or **Remove** buttons. Any member can be edited by selecting it, and the position of a selected item can be changed by clicking in the arrowed keys on the right of the **Members** list.

The list items for the *Memory* **ListBox** are *256*, *512* and *1024*. The *Disk size* **ListBox** has items *20Mb*, *40Mb*, *60Mb*, *100Mb*, *150Mb* and the *Processor* **ListBox** has items *Pentium III 1.5*, *Pentium III 1.8*, *Pentium 2.1* and *Pentium IV 2.5*.

It is a good idea to complete the user interface by fine tuning the position of the controls to exactly where you want them to be and Altering the size of the font for these controls to provide an attractive appearance. The font can be changed by using the **Font | Size** property of the controls. Unlike Windows applications which allow you to choose a specific point size for the font you are limited to a range of textual options: **Smaller**, **Larger**, **XX Small**, **X Small**, **Small**, **Medium**, **Large**, **X Large** and **XX Large**. In the application shown here **Medium** was chosen for all controls.

At this stage the completed design form should appear similar to that shown in figure 10.5.

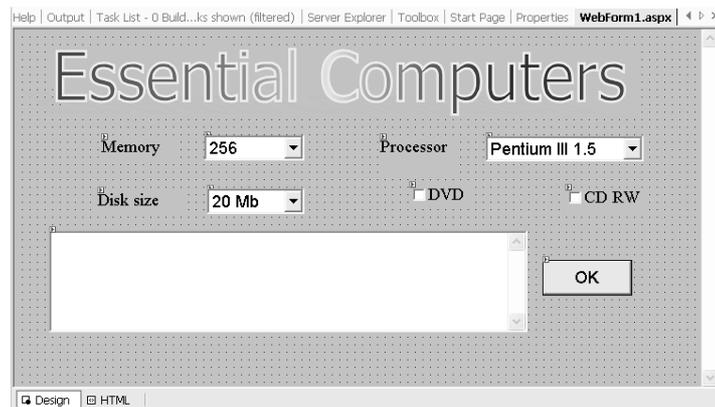


Figure 10.5 The design form.

It is a good idea to run the application at this stage to make sure that the user interface is correct. The lists will be functional, but clicking on the **OK** button will not take any action.

As you add controls to the form and change the properties of controls, Visual Basic .NET converts this to HTML. If you click on the HTML tag at the bottom of the design window you can examine the code which has been generated. It is not advisable to change this unless you are expert at HTML. In practice it is not necessary since you can make all changes indirectly from the design form.

The final stage is to write the code which collects information on the items selected and displays them in the **TextBox** control.

The supporting code

As for Windows applications there is an event handler for dealing with the clicking of the *OK* button. To create an outline event handler double click on the *OK* button. If you need other to create event handlers for any event and control you do so in exactly the same way as for Windows applications as shown in figure 10.6.

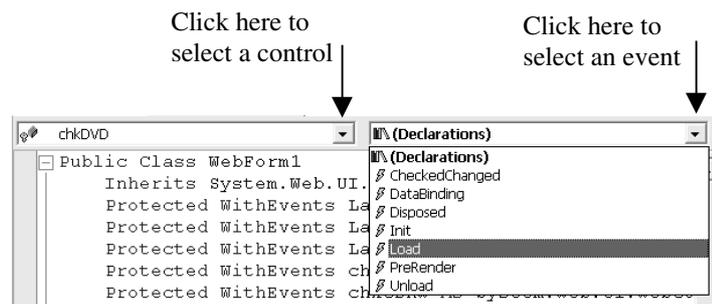


Figure 10.6 Creating event handlers.

The code for the event handler for the *OK* button is shown below:

```
Private Sub btnOK_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles btnOK.Click
    Dim strSummary As String = "All PCs come with 32Mb graphics card"
    strSummary = strSummary & " 56K modem" & vbCrLf
    strSummary = strSummary & "In addition you have chosen: " _
        & vbCrLf
    strSummary = strSummary & lstProcessor.SelectedItem.Text & _
        " processor" & vbCrLf
    strSummary = strSummary & lstMemory.SelectedItem.Text & _
        " memory" & vbCrLf
    strSummary = strSummary & lstDiskSize.SelectedItem.Text & _
        " disk" & vbCrLf
    If chkDVD.Checked = True Then strSummary = strSummary & _
        "DVD "
    If chkCDRW.Checked = True Then strSummary = strSummary & _
        " CD-RW"
    txtSummary.Text = strSummary
End Sub
```

A **String** called *strSummary* is defined and is built up depending on which items have been chosen. This string is assigned to the **Text** property of the **TextBox** for display. The **vbCrLf** constant ensures that following text is placed on the next line. The item chosen from a **ListBox** is given by the **SelectedItem.Text** property. The state of the two **CheckBox** controls is given by examining the **Checked** property which is **True** if the box is checked (that is if the item has been chosen). This completes the application.

In the next chapter we are going to look at some more of the Web controls.