

Lab 3 – Create a Currency Converter

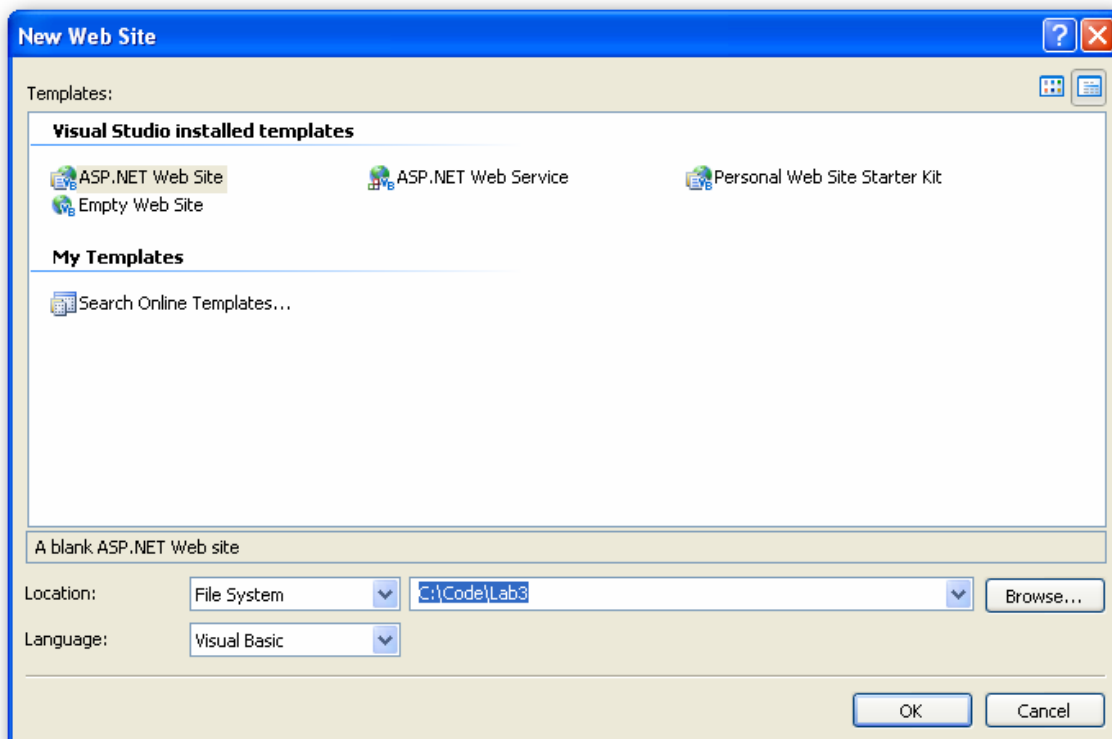
Objective

In this exercise you will create an application similar to the currency converter created in Chapter 5 from your book. This application will convert U.S. dollars to various currencies using HTML server controls. The application will use application settings in the web.config file to make updates easier than recompiling and redeploying the entire application.

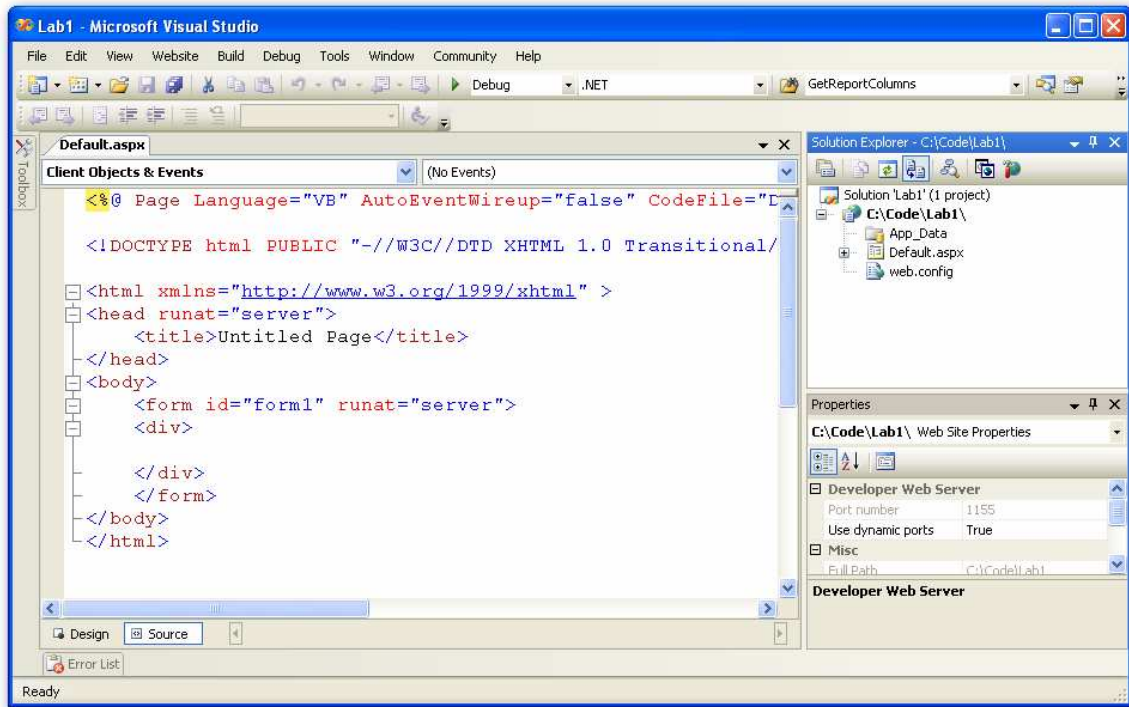
Step-by-Step Instructions

1. Start Visual Studio 2005
2. Create a new web site. File → New → Web Site (Or, File → New Web Site...)
3. Select the following options:

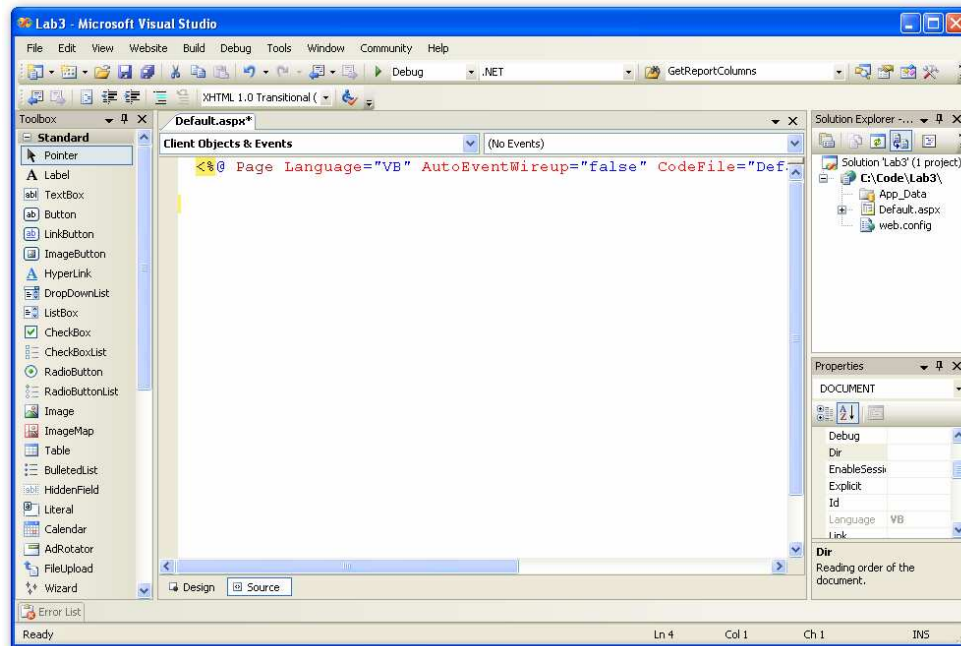
Setting	Value
Visual Studio installed templates	ASP.NET Web Site
Location	File System – C:\Code\Lab3
Language	Visual Basic



- Visual Studio then opens the new web site and displays the Source view of the Default.aspx page. This page is created by default with every new web site.



- Remove all the code in the HTML page except for the page directive at the top of the page.



- Enter the following code into the Default.aspx page. The code below will represent the HTML controls that will be seen by the user. Notice that Visual Studio now uses Intellisense in HTML and within the <code><div> style tag.

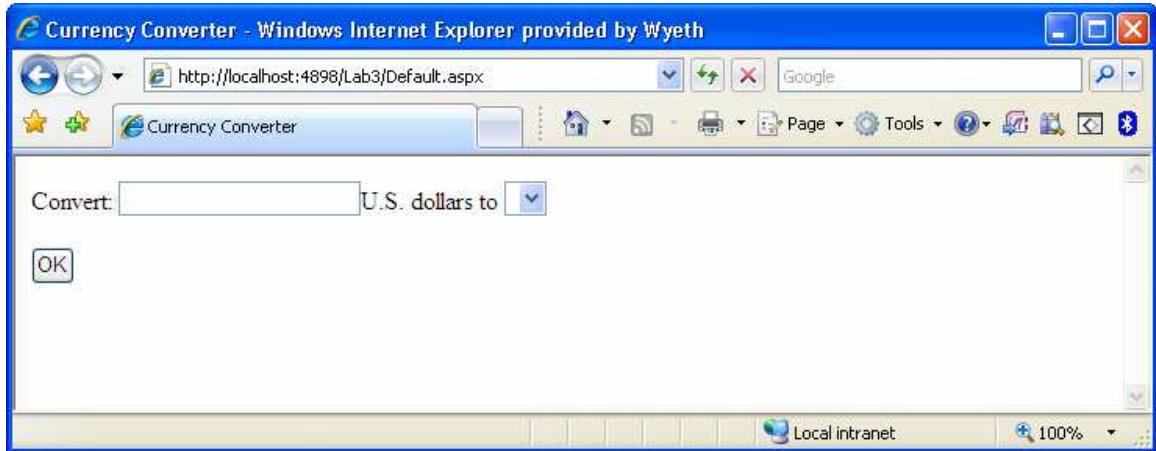
```
<%@ Page Language="VB" AutoEventWireup="false" CodeFile="Default.aspx.vb"
Inherits="_Default" %>

<html>
  <head>
    <title>Currency Converter</title>
  </head>
  <body>
    <form id="Form1" method="post" runat="server">
      Convert:
      <input type="text" id="US" runat="server" />U.S. dollars to
      <select id="Currency" runat="server"></select>
      <br />
      <br />
      <input type="submit" value="OK" id="Convert" runat="server" />
      <br />
      <br />
      <div style="font-weight: bold" id="Result" runat="server"></div>
    </form>
  </body>
</html>
```

- Select **Start Debugging** from the **Debug** menu to view the page.

Notice that the form now has one text box, an empty drop down box, and a submit button. The controls currently have no code behind them so they are essential

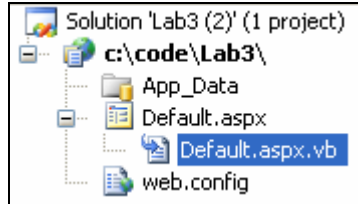
inactive. However, if you click the button, the form will perform a postback to the server.



8. Close the browser to stop debugging the application.

9. Open the Default.aspx.vb page and add the Page_Load event along with the following Visual Basic code to populate the drop down list box.

The Default.aspx.vb file is located under the Default.aspx file as shown below:



This section of code will first check to see if this is the first time the page was loaded or if it was a postback. If the Page_Load event is running because of a post back then there is no need to add the items to the list since they will already be there.

The code then creates three list item objects that contain the name of the items along with a value.

Finally, the three list items are added to the Items collection within the currency control on the form.

```
Partial Class _Default
    Inherits System.Web.UI.Page

    Protected Sub Page_Load(ByVal sender As Object, _
        ByVal e As System.EventArgs) Handles Me.Load

        If Me.IsPostBack = False Then
            Dim euros As New ListItem("Euros", "0.85")
            Dim yen As New ListItem("Japanese Yen", "110.33")
            Dim dollars As New ListItem("Canadian Dollars", "1.2")

            Currency.Items.Add(euros)
            Currency.Items.Add(yen)
            Currency.Items.Add(dollars)
        End If
    End Sub
End Class
```

10. Run the application again to verify that the items were properly added to the list box.



11. Select **Japanese Yen** and then click the **OK** button.

Notice that the postback to the server does work and that the item selected was still Japanese Yen when the page reloaded. This was done automatically by ASP.NET and the use of the view state in the form. In classic ASP, it was required for the developer to write code if they wanted the same behavior.

12. Right-click in the white space of the browser window and click **View Source**. A new window will appear with HTML representation of the form. Notice the **__VIEWSTATE** input tag. This tag now contains hashed data including the item selected in the drop down list box. Your value will be different since view state is hashed differently on each machine.

```
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE" value="/wEPDwUJNjcwMTQxODA2ZGRQDeDths3+Jk07kO2OHT+UHzdYcQ==" />
```

13. Add the following code to the Default.aspx.vb code window below the Page_Load event.

This code is wired to the click event of the Convert button on the form. It will retrieve the value the user entered into the box and then perform the calculation based upon which currency the user selected.

Once the calculation is complete, the result will be shown on the screen. The **Result** <div> tag was added to the form in a previous step. This <div> tag is simply a placeholder for the results.

```

Protected Sub Convert_ServerClick(ByVal sender As Object, _
    ByVal e As System.EventArgs) Handles Convert.ServerClick

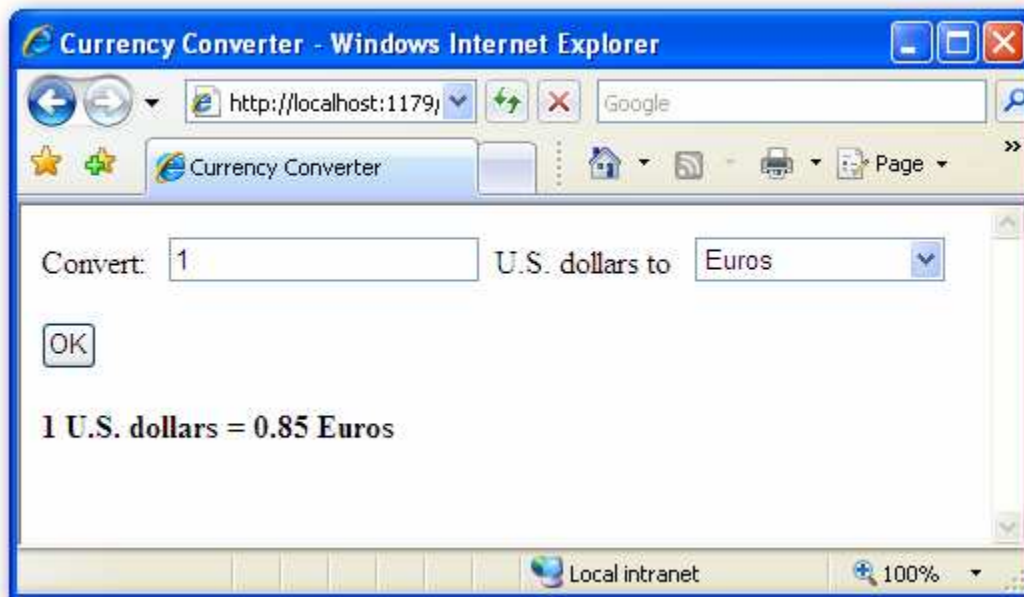
    Dim usValue As Decimal = Val(US.Value)

    Dim item As ListItem = Currency.Items(Currency.SelectedIndex)

    Dim conversionValue As Decimal = usValue * Val(item.Value)
    Result.InnerText = usValue.ToString() & " U.S. dollars = "
    Result.InnerText &= conversionValue.ToString() & " " & item.Text
End Sub

```

14. Run the application again. Experiment with different values and currencies.



15. The limitation of the application right now is that the currencies change frequently. It is always better to avoid recompilation and redeployment of the code. Therefore, we are going to add the currency values to the web.config value of the web site to make the frequent changes easier to manage.

Open the web.config file. Replace `<appSettings />` with the new appSettings section of the file. Then, add the values as shown below:

```

<configuration>
  <appSettings>
    <add key="Euros" value="0.85" />
    <add key="Japanese Yen" value="110.33" />
    <add key="Canadian Dollars" value="1.2" />
  </appSettings>
  <connectionStrings/>

  ...
</configuration>

```

16. Now, update the code in the Default.aspx.vb file.

The first change will be to add the **Imports** line to the first line of the file. This line will allow us to shorten our reference to the WebConfigurationManager later in the code.

Then, update the Convert_ServerClick event to pull the values from the web.config file based upon the currency chosen by the user.

The complete code listing is shown below

```
Imports System.Web.Configuration

Partial Class _Default
    Inherits System.Web.UI.Page

    Protected Sub Page_Load(ByVal sender As Object, _
        ByVal e As System.EventArgs) Handles Me.Load

        If Me.IsPostBack = False Then
            Dim euros As New ListItem("Euros", "0.85")
            Dim yen As New ListItem("Japanese Yen", "110.33")
            Dim dollars As New ListItem("Canadian Dollars", "1.2")

            Currency.Items.Add(euros)
            Currency.Items.Add(yen)
            Currency.Items.Add(dollars)
        End If
    End Sub

    Protected Sub Convert_ServerClick(ByVal sender As Object, _
        ByVal e As System.EventArgs) Handles Convert.ServerClick
        Dim usValue As Decimal = Val(US.Value)

        Dim item As ListItem = Currency.Items(Currency.SelectedIndex)

        Dim conversionRate As Decimal = _
            WebConfigurationManager.AppSettings(item.Text)

        Dim conversionValue As Decimal = usValue * conversionRate
        Result.InnerText = usValue.ToString() & " U.S. dollars = "
        Result.InnerText &= conversionValue.ToString() & " " & _
            item.Text
    End Sub
End Class
```

Optional: The rates placed in the Page_Load event list items can be removed since they are no longer needed. Since the constructor of the ListItem is overloaded you can simply remove the second parameters in the three lines as shown below.

```
Dim euros As New ListItem("Euros")  
Dim yen As New ListItem("Japanes Yen")  
Dim dollars As New ListItem("Canadian Dollars")
```

17. Run the application to test the newest changes.