Lecture 4

Testing and debugging an ASP.NET application

To test an ASP.NET application, you typically start by running it from within Visual Studio so that you can locate and correct any errors you encounter. This initial testing uses the default browser and, if you're working with a file-system web site, the ASP.NET Development Server. Next, you test the application with other web browser to make sure it works with them, and you test a file-system web site under IIS. Finally, you run the application from outside of Visual Studio to be sure it will work correctly in a production environment.

Three ways to run an application with debugging

- Click the *Start Debugging* button in the Standard toolbar
- Press *F5*
- Choose the *Debug* → *Start* command

Three ways to run an application without debugging

- Press *Ctrl+F5*
- Choose *Debug* →*Start* without Debugging
- Right-click a page in the Solution Explorer and choose *View* in Browser

Three ways to stop an application that's run with debugging

- Press *Shift+F5*
- Click the *Stop Debugging* button in the *Debug* toolbar
- Choose *Debug* → *Stop Debugging*

,

The Autos, Locals, and Watch windows to monitor variables

- The *Autos* window displays information about variables, properties, and constants in the current statement and the three statements before and after the current statement
- The *Local* window displays information about the variables and controls within the scope of the current method
- The *Watch* window let you view the values and expressions you specify, called watch expression. You can display up to four Watch windows.
- To add a watch expression, type a variable or expression into the Name column, highlight a variable or expression in the *Code Editor* window and drag it to the *Watch* window or right-click on a variable or highlighted expression in the *Code Editor* window or a data tip and choose *Add Watch.*
- To delete a row from a Watch window, right-click the row and choose *Delete Watch*. To delete all the rows in a *Watch* window, right-click the window and choose *Select All* to select the rows, then right-click and choose *Delete Watch*.

- To display any of these windows, click on its tab if's visible or select the appropriate command from the *Debug → Windows* menu.
- To change the value of a property or variable from any of these windows, double-click on the value in the *Value* column, then type a new value and press the *Enter* key.

The Immediate window

- You can use the *Immediate* window to display and assign values from a program during execution. To display this window, click on the *Immediate* Window tab or use the *Debug → Windows → Immediate* command
- To display a value in the *Immediate* window, enter a question mark followed by the expression whose value you want to display. Then, press the *Enter* key
- To assign a user-defined method from the *Immediate* window, enter its name and any arguments it requires. Then press the *Enter* key. If you want to display the value that's returned by a method, precede the method call with a question mark.
- To reissue a command, use the *Up* and *Down* arrow keys to scroll through the commands until you find the one you want. Then, modify the command if necessary and press the *Enter* key to execute it.
- To remove all commands and output from the *Immediate* window, use the *Clear All* command in the shortcut menu for the window.

Trace feature

The ***Trace feature*** is an ASP.NET feature that displays some useful information that you can't get by using the debugger.

A Page directive that enables tracing for the Cart page

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Cart.aspx.cs" Inherits="Cart"
        Trace="True"%>
```

The ASP.NET *Trace feature* traces the execution of a page and displays trace information and other information at the bottom of that page. To activate the trace feature for a page, you add a *Trace* attribute to the *Page* directive at the top of the *aspx* for the page and set its value to *True* as shown above. The *Trace* information is divided into several tables that provide specific types of trace information. For example, the Trace information table provides information about how the page request was processed, and the *Session State* table provides information about the items currently stored in session state.

Custom trace messages

In some cases, you may want to add your own messages to the trace information that's generated by the trace feature.

Common members of the *TraceContext* class

Property	Description
IsEnabled	True if tracing is enabled for the page
Method	**Description**
Write(message)	Writes a message to the trace output
Write(category, massage)	Writes a message to the trace output with the specified category
Warn(message)	Writes a message in red type to the trace output
Warn(category, massage)	Writes a message in red type to the trace output with the specified category

Code that writes a custom trace message

```
if (Trace.IsEnabled)
    Trace.Write("Page Load", "Binding products drop-down list");
```

A portion of a trace that includes a custom message

Trace Information

Category	Message	From First(s)	From Last(s)
aspx.page	Begin PreInit		
aspx.page	End PreInit	2.93333370582015E-05	0.000029
aspx.page	Begin Init	5.44761973938028E-05	0.000025
aspx.page	End Init	9.07936623230047E-05	0.000036
aspx.page	Begin InitComplete	0.000109231759902446	0.000018
aspx.page	End InitComplete	0.000127111127252207	0.000018
aspx.page	Begin PreLoad	0.000144990494601968	0.000018
aspx.page	End PreLoad	0.000169853989822729	0.000025
aspx.page	Begin Load	0.00018773335717249	0.000018
Page Load	Binding products drop-down list	12.2905950337264	12.290407
aspx.page	End Load	17.3263501366794	5.035755
aspx.page	Begin LoadComplete	17.3264238890697	0.000074
aspx.page	End LoadComplete	17.3264685874881	0.000045
aspx.page	Begin PreRender	17.326507419239	0.000039
aspx.page	End PreRender	17.3271829240867	0.000676
aspx.page	Begin PreRenderComplete	17.3273687018881	0.000186
aspx.page	End PreRenderComplete	17.327412282846	0.000044
aspx.page	Begin SaveState	17.3544140386558	0.027002
aspx.page	End SaveState	17.3555038419687	0.001090
aspx.page	Begin SaveStateComplete	17.3555633467382	0.000060
aspx.page	End SaveStateComplete	17.3556049721403	0.000042
aspx.page	Begin Render	17.3556426864308	0.000038
aspx.page	End Render	17.3601183949357	0.004476

Custom trace message

How to write information directly to the HTTP output stream

A *Page_Load* event handler that writes to the HTTP output stream

```
Protected void Page_Load(object sender, EventArgs e)
{
   this.GetCart();
   if (!IsPostBack)
   {
      this.DisplayCart();
      Response.Write("Items in cart = " + cart.count + "<br />");
    }
}
```

Code that writes HTML output from another class

```
HttpContext.Current.Response.Write("Now updating file.<br />");
```

Working with HTML using Visual Studio

How web server controls are rendered

Before a web page can be displayed in a browser, any web server controls it contains must be rendered to standard HTML.

Some common web server controls and how they're rendered

Name	HTML element	Additional information
Name	**	The ** element is typically used to apply special formatting to the text it enclosed
TextBox	*<input>*	The *type="text"* attribute identifies this element as a text box
Button	*<input>*	The *type="submit"* attribute identifies this element as a button
ImageButton	*<input>*	The *type="image"* attribute identifies this element as an image button
LinkButton	*<a>*	This control is rendered as an *Anchor* element that posts the page back to the server
HyperLink	*<a>*	This control is rendered as an *Anchor* element that links to a URL
ListBox	*<select>*	Each item in the list is defined by an *<option>* element within the *<select>* element
DropDownList	*<select>*	Each item in the list is defined by an *<option>* element within the *<select>* element

Image	*<img>*	Displays the specified image

The *aspx* code with web server controls

```
<asp:Image ID="Image1" runat="server" ImageUrl="~/Images/banner.jpg" /><br /><br />
    Your shopping cart:<br />
<table border="0" cellpadding="0" cellspacing="0" style="width: 500px">
  <tr>
    <td style="width: 286px; height: 153px">
      <asp:ListBox ID="lstCart" runat="server" Height="135px" Width="267px"></asp:ListBox>
    </td>
    <td style="height: 153px">
      <asp:Button ID="btnRemove" runat="server" Text="Remove Item" Width="100px"
                  OnClick="btnRemove_Click" /><br /><br />
      <asp:Button ID="btnEmpty" runat="server" Text="Empty Cart" Width="100px"
                  OnClick="btnEmpty_Click" />
    </td>
  </tr>
</table><br />
<asp:Button ID="btnContinue" runat="server" PostBackUrl="~/Order.aspx" Text="Continue
Shopping" /> 
<asp:Button ID="btnCheckOut" runat="server" Text="Check Out" OnClick="btnCheckOut_Click"
/><br /><br />      <asp:Label ID="lblMessage" runat="server"></asp:Label>
```

How web server controls are rendered in HTML

```
<img id="Image1" src="Images/banner.jpg" style="border-width:0px;" /><br /> <br />
    Your shopping cart:<br />
  <table border="0" cellpadding="0" cellspacing="0" style="width: 500px">
    <tr>
      <td style="width: 286px; height: 153px">
        <select size="4" name="lstCart" id="lstCart" style="height:135px;width:267px;">
            <option value="Austin Powers (1 at $79.99 each)">Austin Powers (1 at $79.99
            each)</option>
        </select>
      </td>
      <td style="height: 153px">
        <input type="submit" name="btnRemove" value="Remove Item" id="btnRemove"
            style="width:100px;"  /><br /><br />
        <input type="submit" name="btnEmpty" value="Empty Cart" id="btnEmpty"
            style="width:100px;" />
      </td>
    </tr>
  </table><br />
```

```
    <input type="submit" name="btnContinue" value="Continue Shopping"
        onclick="javascript:WebForm_DoPostBackWithOptions(new
        WebForm_PostBackOptions(&quot;btnContinue&quot;, &quot;&quot;, false, &quot;&quot;,
        &quot;Order.aspx&quot;, false, false))" id="btnContinue" /> 
    <input type="submit" name="btnCheckOut" value="Check Out" id="btnCheckOut" /><br /><br
/>
    <span id="lblMessage"></span>
```

Coding HTML documents

The HTML generated for a new ASP.NET page

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Cart.aspx.cs" Inherits="Cart" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
            "http://www.w3.org/TR/xhtml1/DTD/xhtml1- transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>Shopping Cart</title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
    <asp:Image ID="Image1" runat="server" ImageUrl="~/Images/banner.jpg" /><br /><br />
       Your shopping cart:<br />
    <table border="0" cellpadding="0" cellspacing="0" style="width: 500px">
      <tr>
        <td style="width: 286px; height: 153px">
          <asp:ListBox ID="lstCart" runat="server" Height="135px" Width="267px"></asp:ListBox>
        </td>
        <td style="height: 153px">
          <asp:Button ID="btnRemove" runat="server" Text="Remove Item" Width="100px"
                    OnClick="btnRemove_Click" /><br /><br />
          <asp:Button ID="btnEmpty" runat="server" Text="Empty Cart" Width="100px"
                    OnClick="btnEmpty_Click" />
        </td>
      </tr>
    </table>
    <br />
    <asp:Button ID="btnContinue" runat="server" PostBackUrl="~/Order.aspx" Text="Continue
    Shopping" /> 
     <asp:Button ID="btnCheckOut" runat="server" Text="Check Out" OnClick="btnCheckOut_Click"
    /><br />
    <br />
    <asp:Label ID="lblMessage" runat="server"></asp:Label>
```

```
    </div>
    </form>
</body>
</html>
```

HTML elements generated for a new page

Element	Start tag	End tag	Description
Page directive	*<%@ page ...>*		Identifies various options applied to the web page.
Doctype declaration	*<!DOCTYPE ...>*		Identifies the type of HTML documents.
Html element	*<html>*	*</html>*	Marks the beginning and end of an HTML document.
Head element	*<head>*	*</head>*	Marks the beginning and end of the document head.
Title element	*<title>*	*</title>*	Provides the title for the page.
Body element	*<body>*	*</body>*	Marks the beginning and end of the document body.
Form element	*<form>*	*</form>*	Marks the beginning and end of the form.
Div element	* *	* *	Marks the beginning and end of a division within the form

A typical *Page* directive

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Cart.aspx.cs" Inherits="Cart" %>
```

Common attributes of the *Page* directive

Attribute	Description
Language	Specifies the language used to implement the processing for the page.
AutoEventWireup	Indicates that the event handlers for the page will be called automatically when the events occur for the page.
CodeFile	Specifies the name of the code-behind file. This file defines the partial class that is compiled with the partial class that ASP.NET generates from the page at runtime.
Inherits	Specifies the name of the compiled class (dll) that the page inherits at runtime. This name is typically the same as the name of the partial class defined in the *CodeFile* attribute.

Trace	Indicates whether tracing is enabled. The default value is *False*.

A typical *Doctype* declaration

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1- transitional.dtd">
```

Description

- The *Page* directive is processed by ASP.NET when the page is compiled. It isn't included in the output that's sent to the browser.
- The *Doctype* declaration specifies that the root element of the document – that is, the first element after the *Doctype* declaration – is an HTML element.
- The *Public* attribute of the *Doctype* declaration and the string that follows it specify the version of HTML that the document compiles with.