



Chapter 20: Web App Development with ASP.NET in C#

Internet & World Wide Web
How to Program, 5/e

Note: This chapter is a copy of Chapter 19 of our book *Visual C# 2010 How to Program*. For that reason, we simply copied the PowerPoint slides for this chapter and *did not* re-number them



OBJECTIVES

In this chapter you'll learn:

- Web application development using ASP.NET.
- To handle the events from a Web Form's controls.
- To use validation controls to ensure that data is in the correct format before it's sent from a client to the server.
- To maintain user-specific information.
- To create a data-driven web application using ASP.NET and LINQ to SQL.



19.1 Introduction

19.2 Web Basics

19.3 Multitier Application Architecture

19.4 Your First Web Application

19.4.1 Building the `WebTime` Application

19.4.2 Examining `WebTime.aspx`'s Code-Behind File

19.5 Standard Web Controls: Designing a Form

19.6 Validation Controls

19.7 Session Tracking

19.7.1 Cookies

19.7.2 Session Tracking with `HttpSessionState`

19.7.3 `Options.aspx`: Selecting a Programming Language

19.7.4 `Recommendations.aspx`: Displaying Recommendations Based on Session Values



19.8 Case Study: Database-Driven ASP.NET Guestbook

19.8.1 Building a Web Form that Displays Data from a Database

19.8.2 Modifying the Code-Behind File for the Guestbook Application

19.9 Online Case Study: ASP.NET AJAX

19.10 Online Case Study: Password-Protected Books Database Application

19.11 Wrap-Up



19.2 Web Basics

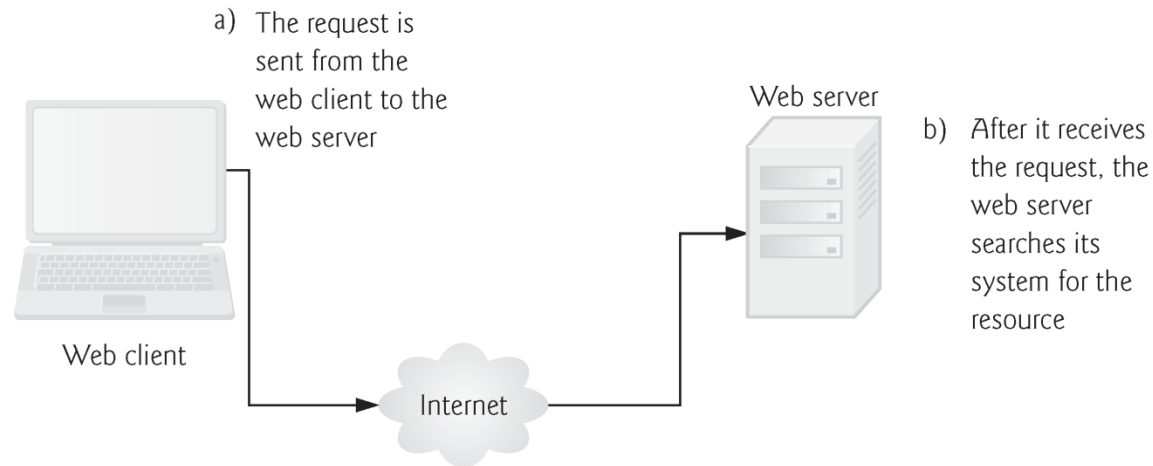


Fig. 19.1 | Client requesting a resource from a web server.

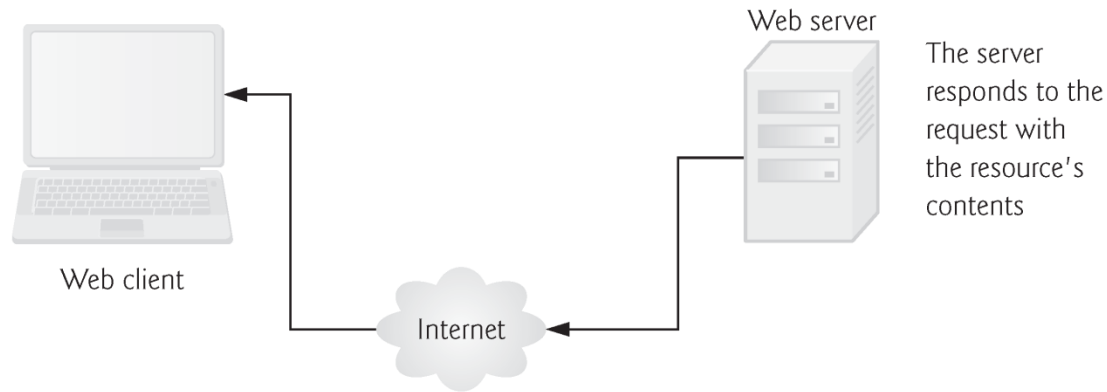


Fig. 19.2 | Client receiving a response from the web server.

19.3 Multitier Application Architecture



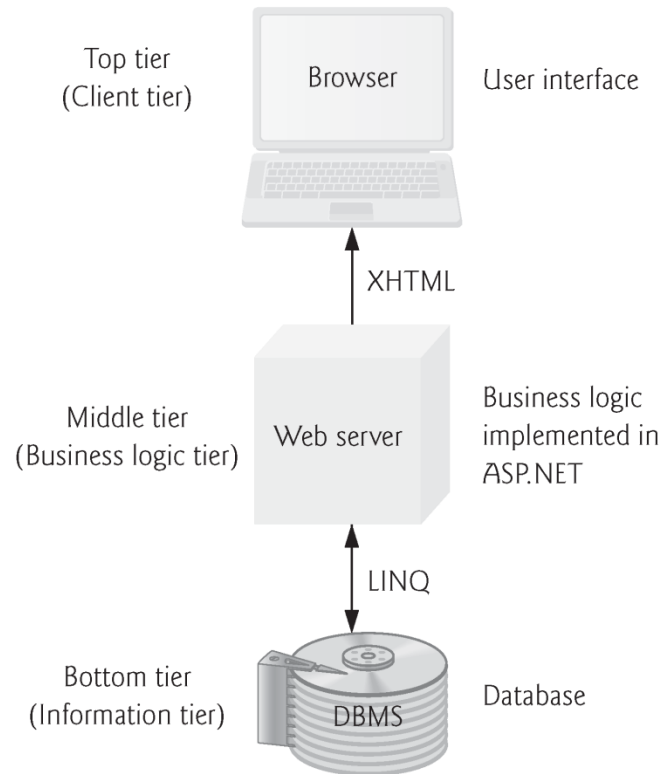


Fig. 19.3 | Three-tier architecture.



19.4 Your First Web Application

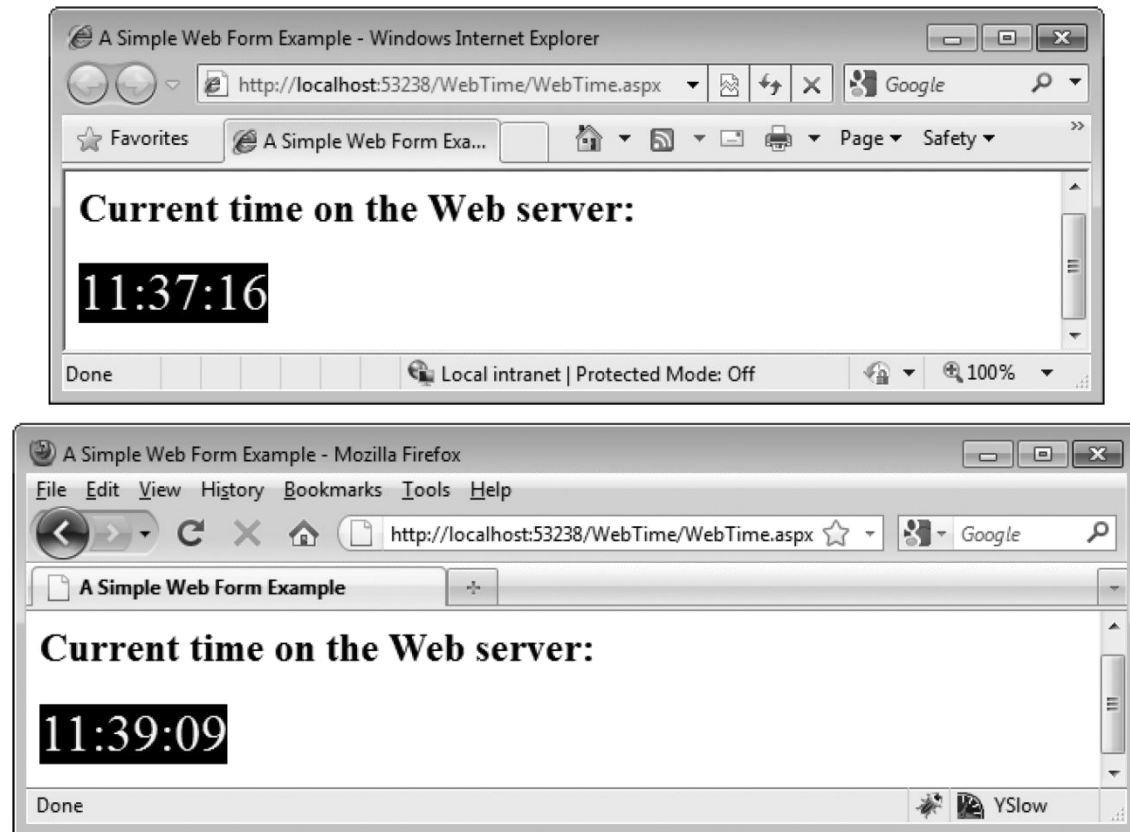


Fig. 19.4 | WebTime web application running in both Internet Explorer and Firefox.

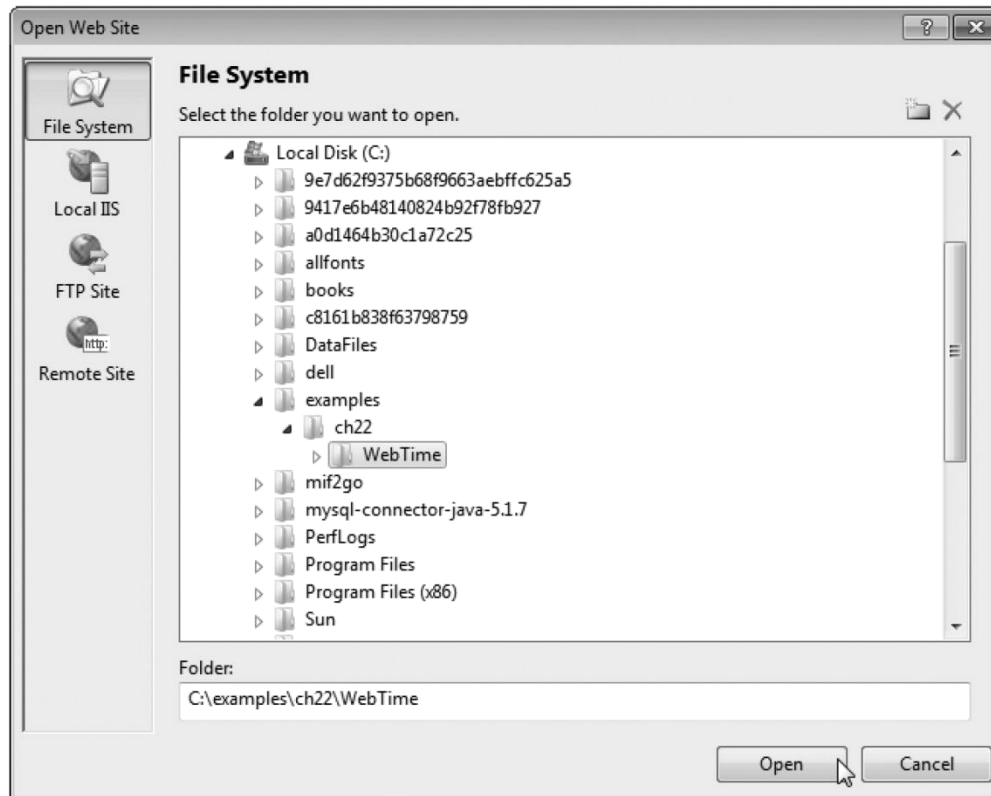


Fig. 19.5 | Open Web Site dialog.

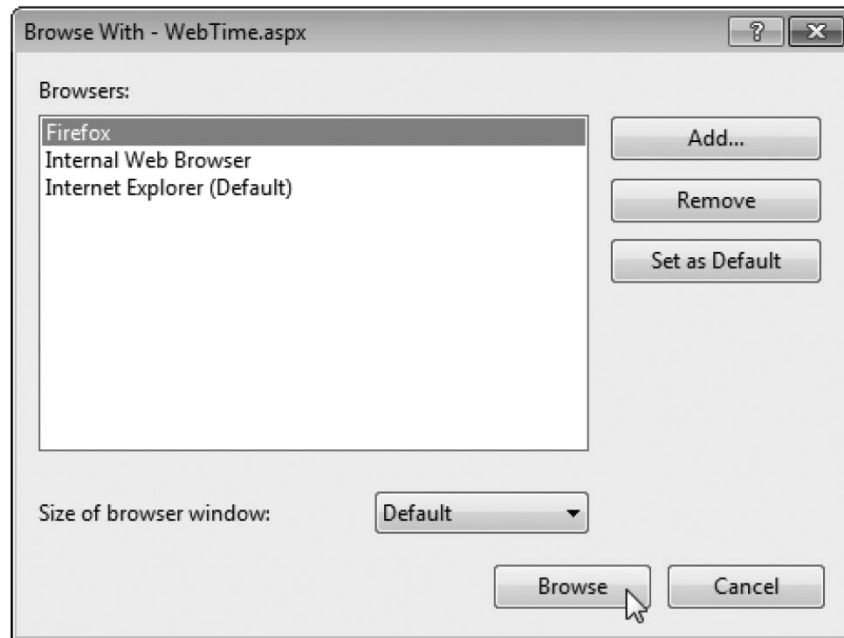


Fig. 19.6 | Selecting another web browser to execute the web application.

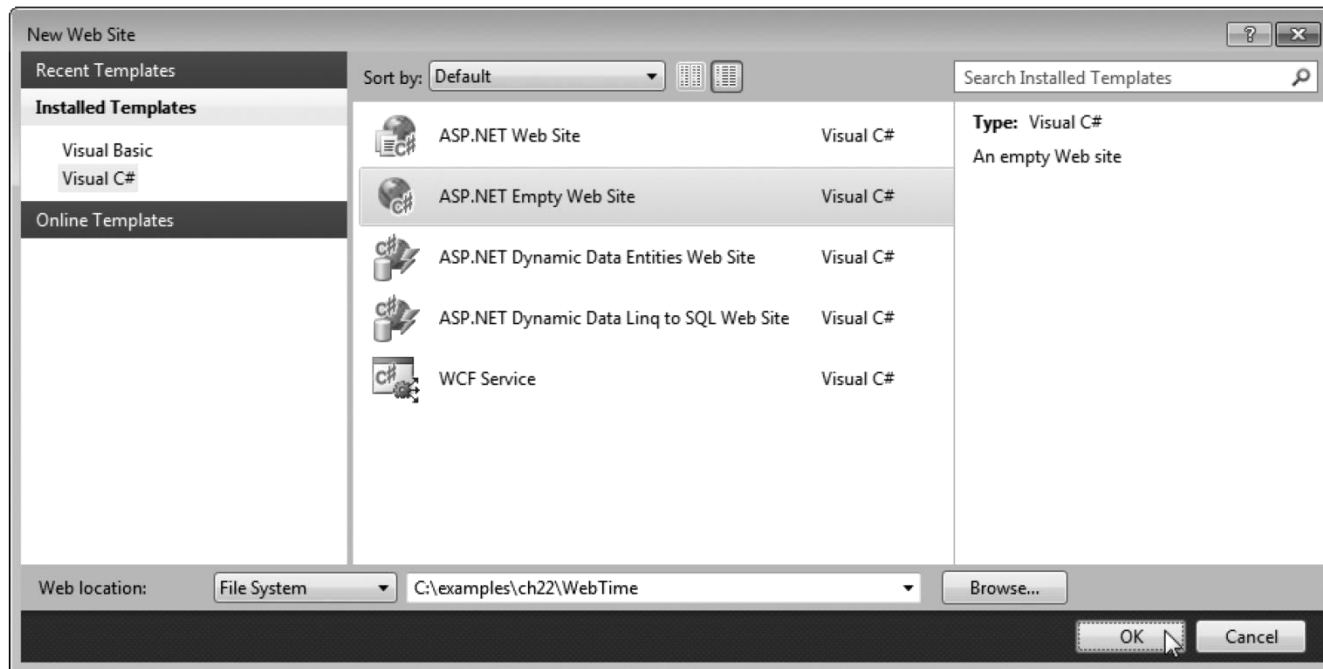


Fig. 19.7 | Creating an ASP.NET Web Site in Visual Web Developer.

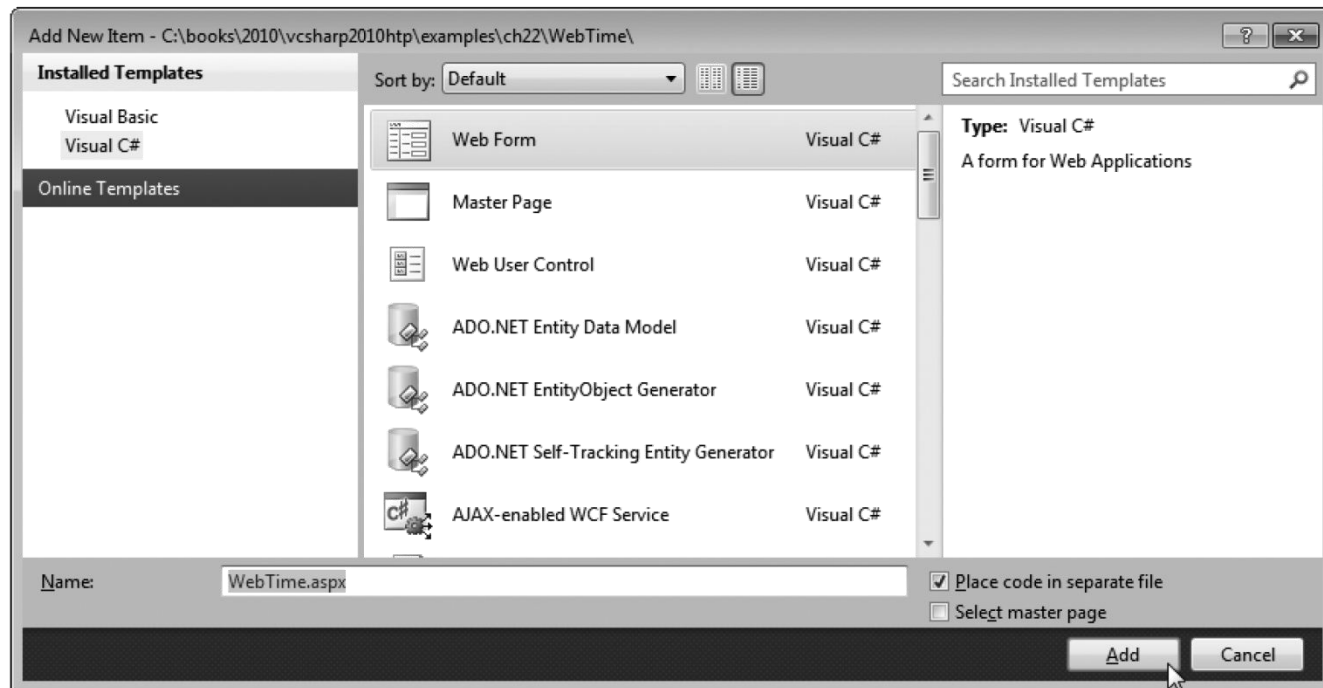


Fig. 19.8 | Adding a new **Web Form** to the website with the **Add New Item** dialog.

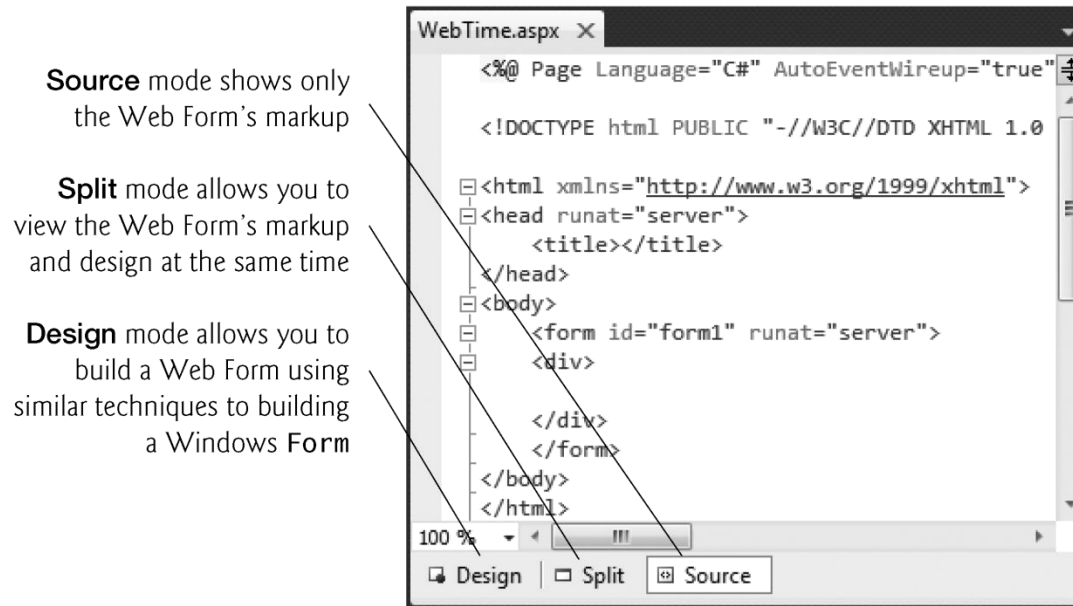


Fig. 19.9 | Web Form in Source view.

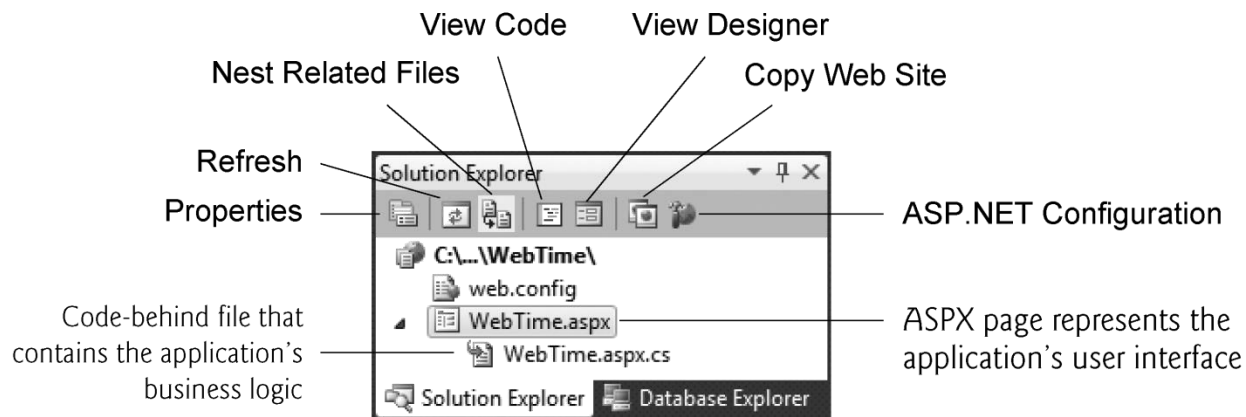


Fig. 19.10 | **Solution Explorer window for an Empty Web Site project after adding the Web Form `WebTime.aspx`.**

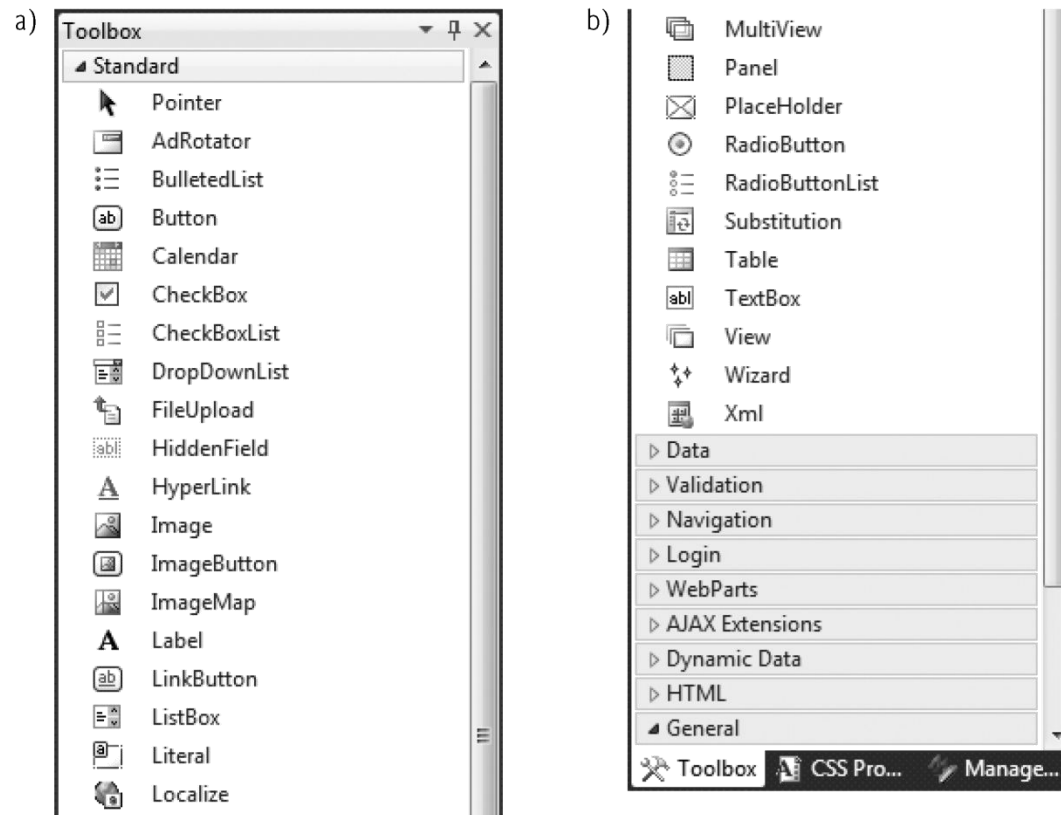


Fig. 19.11 | Toolbox in Visual Web Developer.

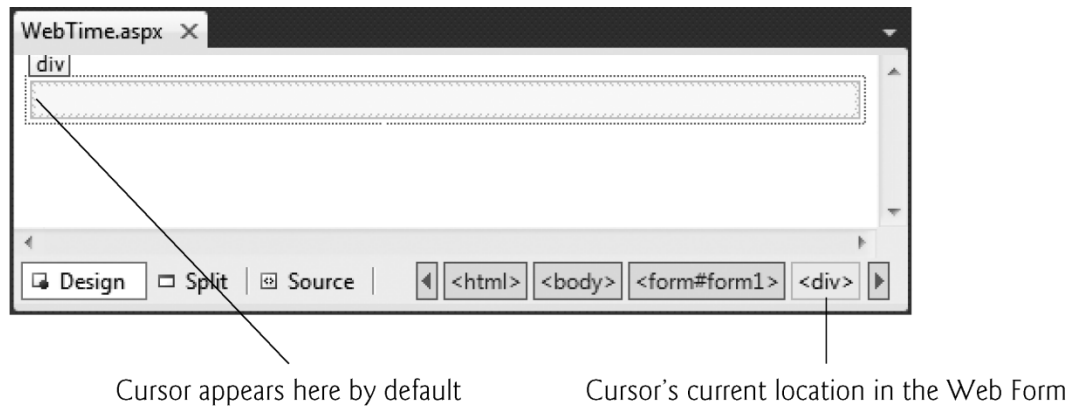


Fig. 19.12 | Design mode of the Web Forms Designer.



Block Format ComboBox

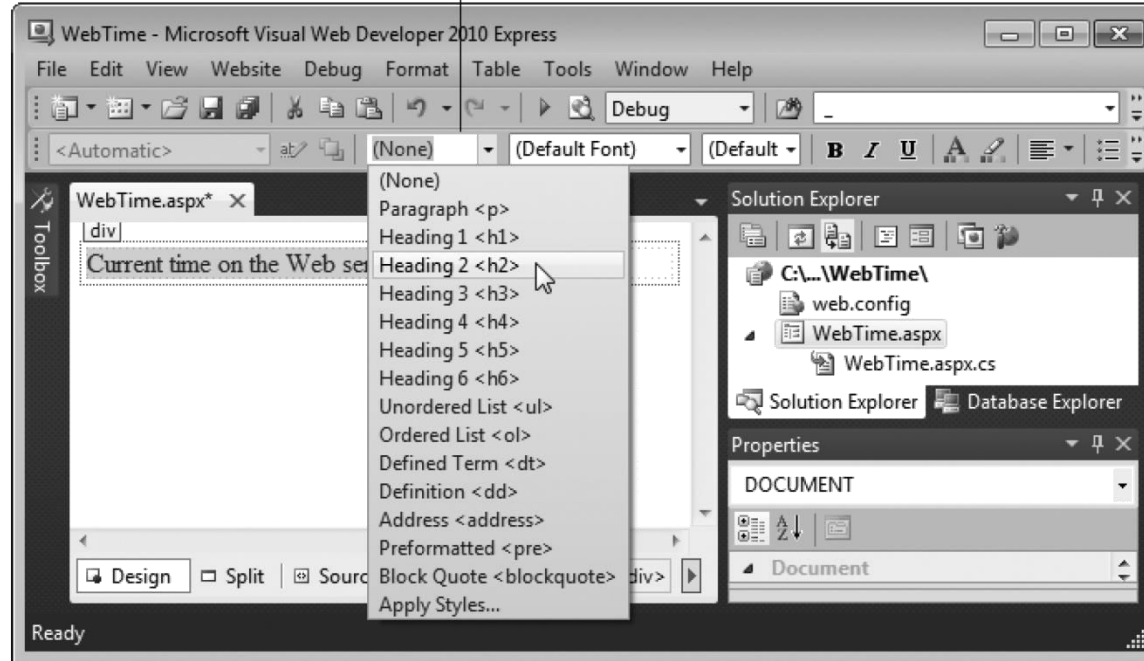


Fig. 19.13 | Changing the text to Heading 2 heading.

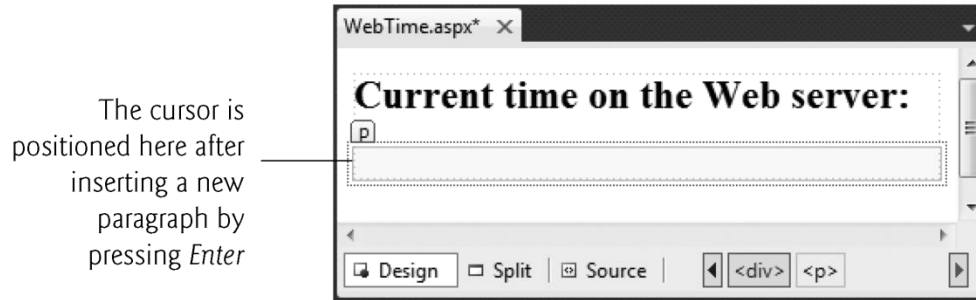


Fig. 19.14 | WebTime.aspx after inserting text and a new paragraph.

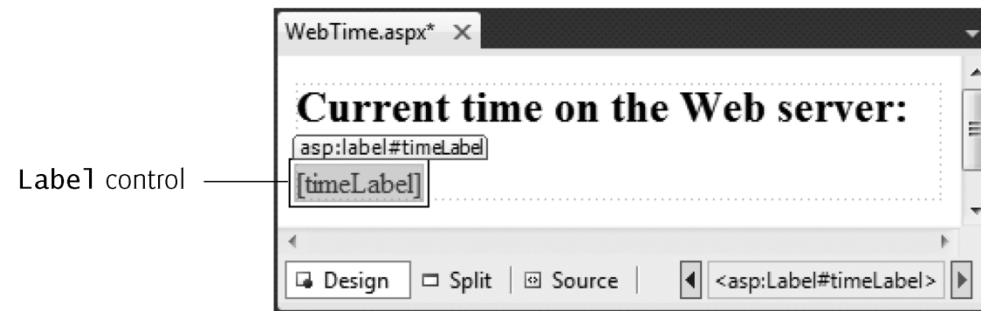


Fig. 19.15 | WebTime.aspx after adding a Label1.

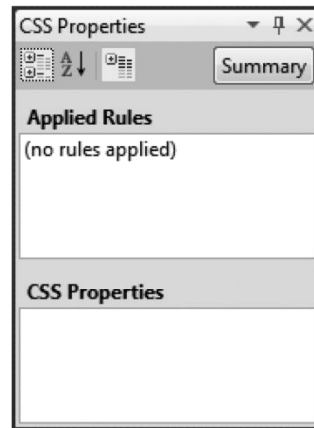


Fig. 19.16 | CSS Properties window.

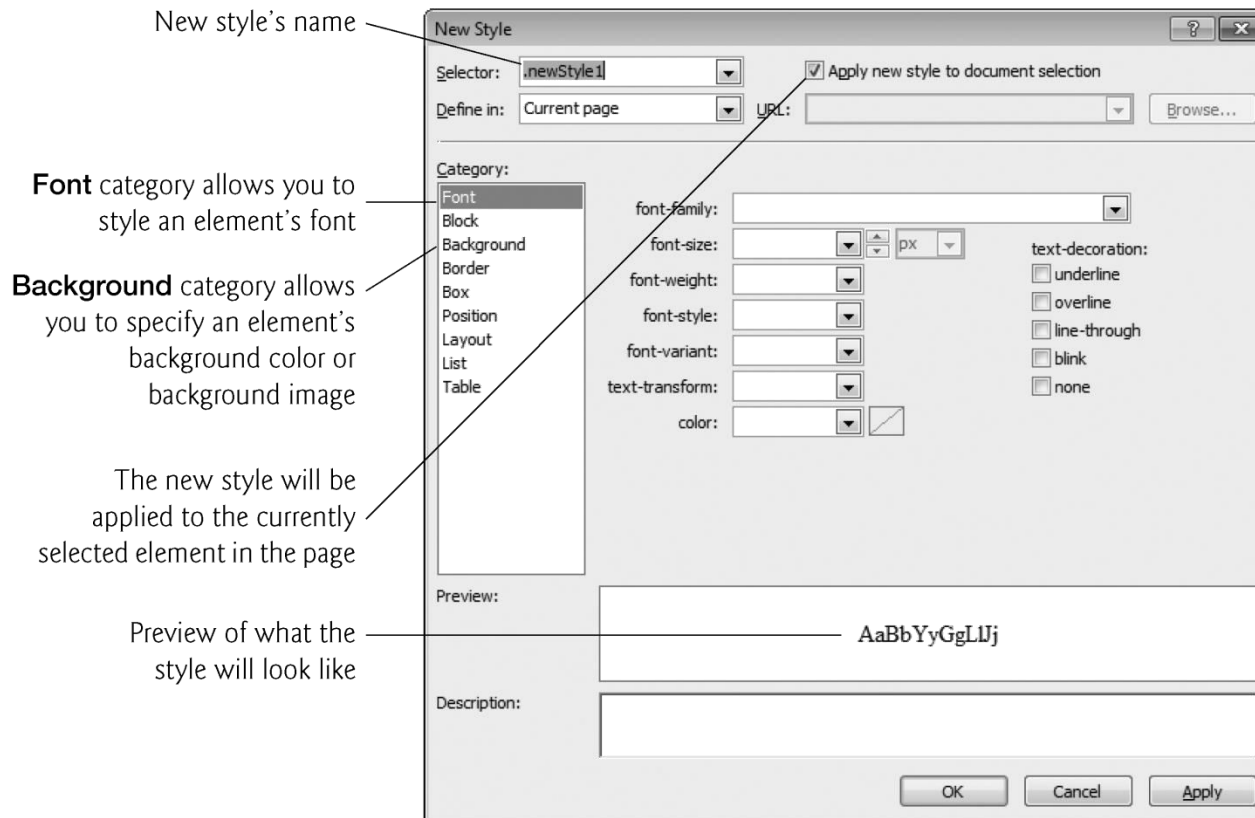


Fig. 19.17 | New Style dialog.



Bold category names indicate the categories in which CSS attribute values have been changed

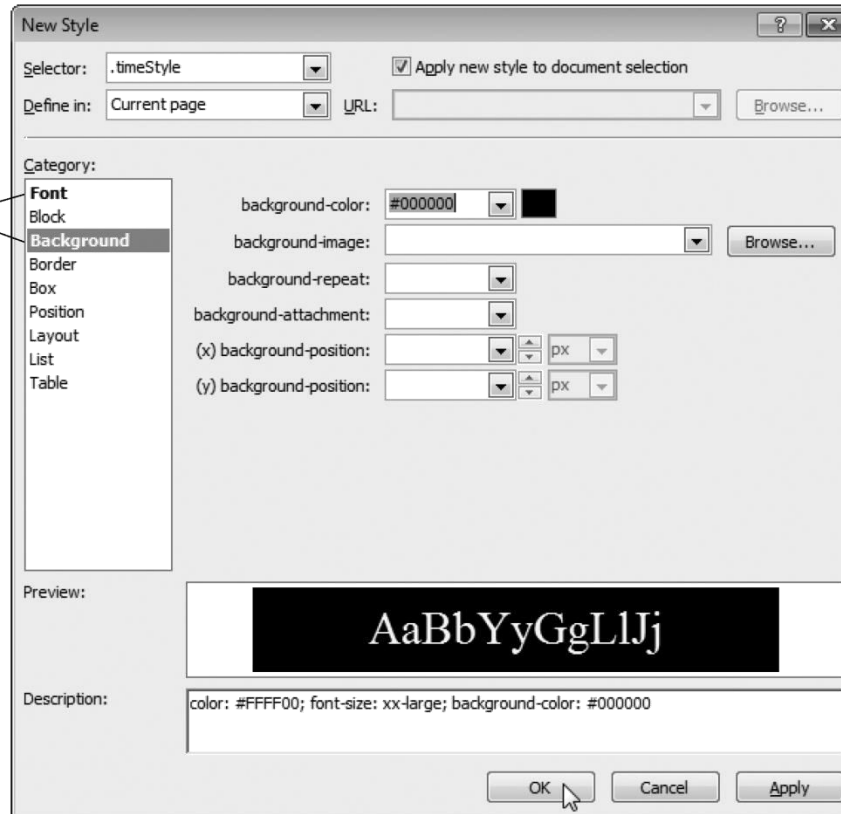


Fig. 19.18 | New Style dialog after changing the Label's style.



Fig. 19.19 | Design view after changing the Label1's style.



```
1 // Fig. 19.20: WebTime.aspx.cs
2 // Code-behind file for a page that displays the web server's time.
3 using System;
4
5 public partial class WebTime : System.Web.UI.Page
6 {
7     // initializes the contents of the page
8     protected void Page_Init( object sender, EventArgs e )
9     {
10         // display the server's current time in timeLabel
11         timeLabel.Text = DateTime.Now.ToString( "hh:mm:ss" );
12     } // end method Page_Init
13 } // end class WebTime
```

Fig. 19.20 | Code-behind file for a page that displays the web server's time.

19.5 Standard Web Controls: Designing a Form





Web control	Description
TextBox	Gathers user input and displays text.
Button	Triggers an event when clicked.
HyperLink	Displays a hyperlink.
DropDownList	Displays a drop-down list of choices from which a user can select an item.
RadioButtonList	Groups radio buttons.
Image	Displays images (for example, PNG, GIF and JPG).

Fig. 19.21 | Commonly used web controls.

Heading 3 paragraph	Registration Form				
Paragraph of plain text	Please fill in all fields and click the Register button.				
Image control					
A table containing four Images and four TextBoxes	<table border="0"> <tr> <td> </td> <td> </td> </tr> <tr> <td> </td> <td> </td> </tr> </table>				
TextBox control					
DropDownList control	Which book would you like information about? 				
HyperLink control	Click here to view more information about our books				
RadioButtonList control	 ☐ Windows 7 ☐ Windows Vista ☐ Windows XP ☐ Mac OS X ☐ Linux ☐ Other				
Button control					

Fig. 19.22 | Web Form that demonstrates web controls.



Insert Table [?] [X]

Size

Rows: Columns:

Layout

Alignment: ☒ Specify width:

Float: ☐ In pixels ☒ In percent

Cell padding: ☐ Specify height:

Cell spacing: ☐ In pixels ☐ In percent

Borders

Size:

Color:

☐ Collapse table border

Background

Color:

☐ Use background picture

Set

☐ Set as default for new tables

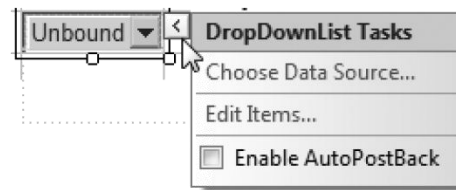
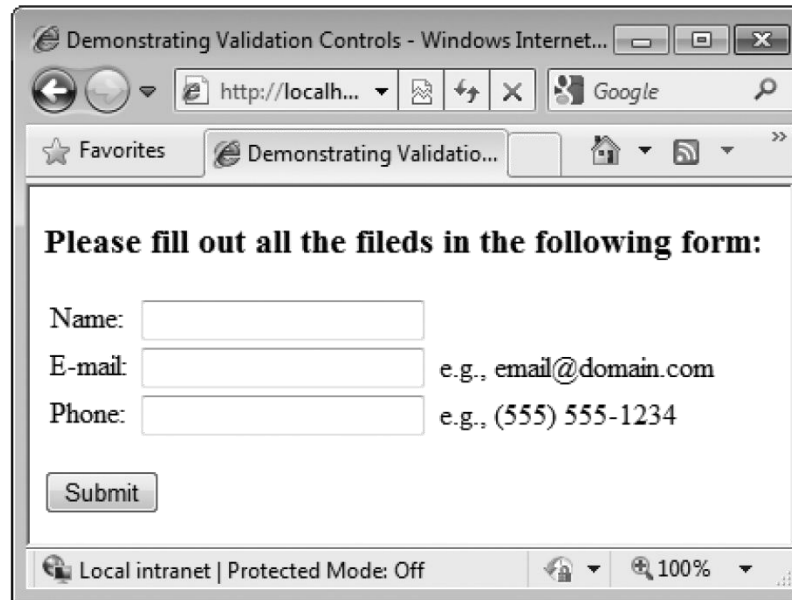


Fig. 19.24 | DropDownList Tasks smart-tag menu.



19.6 Validation Controls

a) Initial Web Form



The screenshot shows a web browser window titled "Demonstrating Validation Controls - Windows Internet...". The address bar displays "http://localh...". The page content includes the instruction "Please fill out all the fields in the following form:" followed by three input fields: "Name:", "E-mail:", and "Phone:". To the right of the "E-mail:" field is the example "e.g., email@domain.com", and to the right of the "Phone:" field is the example "e.g., (555) 555-1234". A "Submit" button is located below the input fields. The browser's status bar at the bottom indicates "Local intranet | Protected Mode: Off" and a zoom level of "100%".

Fig. 19.25 | Validators in a Web Form that retrieves user contact information. (Part I of 4.)



b) Web Form after the user presses the **Submit Button** without having entered any data in the **TextBox**s; each **TextBox** is followed by an error message that was displayed by a validation control

RequiredFieldValidator controls

The screenshot shows a web browser window titled "Demonstrating Validation Controls - Windows Internet...". The address bar shows "http://localh...". The page content includes a heading "Please fill out all the fields in the following form:" followed by three input fields: "Name:", "E-mail:", and "Phone:". Each field is followed by a red error message: "Please enter your name", "Please enter your e-mail address", and "Please enter your phone number". A "Submit" button is at the bottom. The browser status bar at the bottom indicates "Local intranet | Protected Mode: Off" and "100%".

Fig. 19.25 | Validators in a Web Form that retrieves user contact information. (Part 2 of 4.)



c) Web Form after the user enters a name, an invalid e-mail address and an invalid phone number in the **TextBoxes**, then presses the **Submit Button**; the validation controls display error messages in response to the invalid e-mail and phone number values

RegularExpressionValidator
controls

The screenshot shows a web browser window titled "Demonstrating Validation Controls - Windows Internet...". The address bar shows "http://localh...". The page content includes a heading "Please fill out all the fields in the following form:". Below this are three input fields: "Name:" with the value "Bob White", "E-mail:" with the value "bwhite", and "Phone:" with the value "55-1234". To the right of the "E-mail:" field is a hint "e.g., email@domain.com" and an error message "Please enter an e-mail address in a valid format". To the right of the "Phone:" field is a hint "e.g., (555) 555-1234" and an error message "Please enter a phone number in a valid format". At the bottom of the form is a "Submit" button. The browser's status bar at the bottom indicates "Local intranet | Protected Mode: Off" and a zoom level of "100%".

Fig. 19.25 | Validators in a Web Form that retrieves user contact information. (Part 3 of 4.)



d) The Web Form after the user enters valid values for all three **TextBoxes** and presses the **Submit Button**

Please fill out all the fields in the following form:

Name:

E-mail: e.g., email@domain.com

Phone: e.g., (555) 555-1234

Thank you for your submission
We received the following information:
Name: Bob White
E-mail: bwhite@deitel.com
Phone: (555) 555-9876

Fig. 19.25 | Validators in a Web Form that retrieves user contact information. (Part 4 of 4.)



```
1 // Fig. 19.26: Validation.aspx.cs
2 // Code-behind file for the form demonstrating validation controls.
3 using System;
4
5 public partial class Validation : System.Web.UI.Page
6 {
7     // Page_Load event handler executes when the page is loaded
8     protected void Page_Load( object sender, EventArgs e )
9     {
10         // if this is not the first time the page is loading
11         // (i.e., the user has already submitted form data)
12         if ( IsPostBack )
13         {
14             Validate(); // validate the form
15
16             // if the form is valid
17             if ( IsValid )
18             {
19                 // retrieve the values submitted by the user
20                 string name = nameTextBox.Text;
21                 string email = emailTextBox.Text;
22                 string phone = phoneTextBox.Text;
23             }
16 }
```

Fig. 19.26 | Code-behind file for the form demonstrating validation controls. (Part 1 of 2.)



```
24         // show the the submitted values
25         outputLabel.Text = "Thank you for your submission<br/>" +
26             "We received the following information:<br/>";
27         outputLabel.Text +=
28             String.Format( "Name: {0}{1}E-mail:{2}{1}Phone:{3}",
29                 name, "<br/>", email, phone);
30         outputLabel.Visible = true; // display the output message
31     } // end if
32 } // end if
33 } // end method Page_Load
34 } // end class Validation
```

Fig. 19.26 | Code-behind file for the form demonstrating validation controls. (Part 2 of 2.)



19.7 Session Tracking



Portability Tip 19.1

Users may disable cookies in their web browsers to help ensure their privacy. Such users will experience difficulty using web applications that depend on cookies to maintain state information.

a) User selects a language from the **Options.aspx** page, then presses **Submit** to send the selection to the server



Fig. 19.27 | ASPX file that presents a list of programming languages. (Part I of 5.)



d) The Web Form
after the user enters
valid values for all
three `TextBox`s and
presses the **Submit**
Button

Please fill out all the fields in the following form:

Name:

E-mail: e.g., email@domain.com

Phone: e.g., (555) 555-1234

Thank you for your submission
We received the following information:
Name: Bob White
E-mail: bwhite@deitel.com
Phone: (555) 555-9876

Local intranet | Protected Mode: Off 100%

Fig. 19.27 | ASPX file that presents a list of programming languages. (Part 2 of 5.)

c) User selects another language from the **Options.aspx** page, then presses **Submit** to send the selection to the server



Fig. 19.27 | ASPX file that presents a list of programming languages. (Part 3 of 5.)

d) `Options.aspx` page is updated to hide the controls for selecting a language and to display the user's selection; the user clicks the hyperlink to get a list of book recommendations

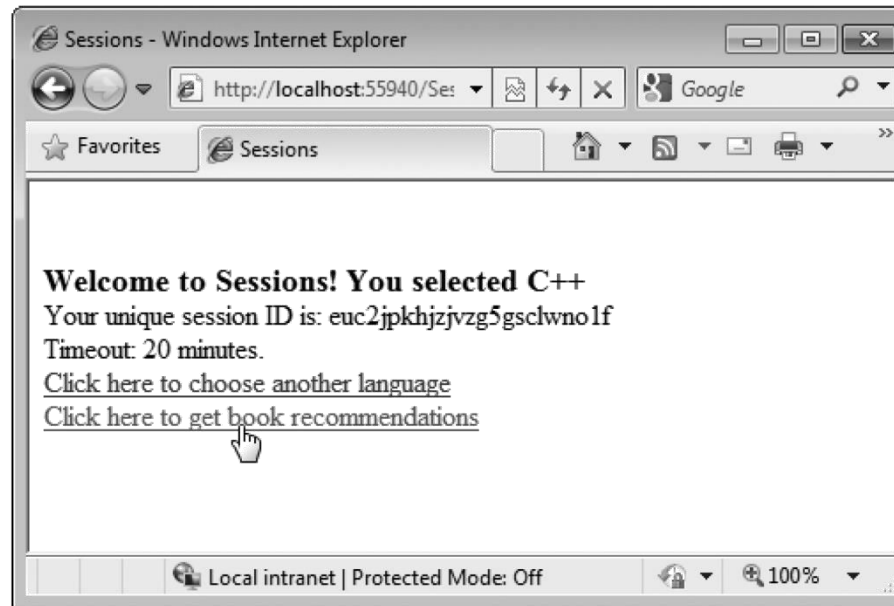


Fig. 19.27 | ASPX file that presents a list of programming languages. (Part 4 of 5.)



e) `Recommendations.aspx`
displays the list of
recommended books based on
the user's selections

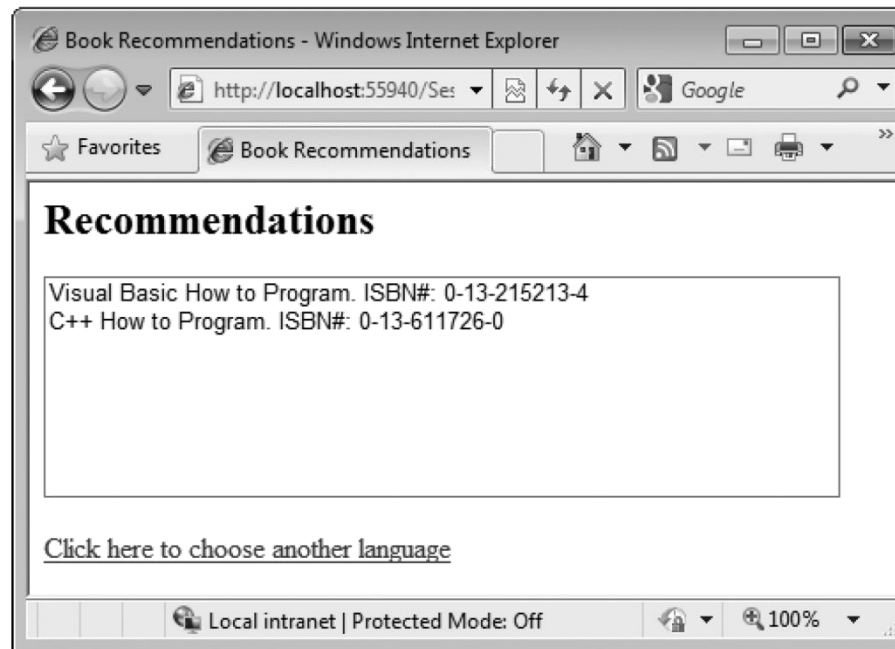


Fig. 19.27 | ASPX file that presents a list of programming languages. (Part 5 of 5.)



```
1 // Fig. 19.28: Options.aspx.cs
2 // Processes user's selection of a programming language by displaying
3 // links and writing information in a Session object.
4 using System;
5
6 public partial class Options : System.Web.UI.Page
7 {
8     // if postback, hide form and display links to make additional
9     // selections or view recommendations
10    protected void Page_Load( object sender, EventArgs e )
11    {
12        if ( IsPostBack )
13        {
14            // user has submitted information, so display message
15            // and appropriate hyperlinks
16            responseLabel.Visible = true;
17            idLabel.Visible = true;
18            timeoutLabel.Visible = true;
19            languageLink.Visible = true;
20            recommendationsLink.Visible = true;
21        }
22    }
23 }
```

Fig. 19.28 | Process user's selection of a programming language by displaying links and writing information in an HttpSessionState object. (Part I of 3.)



```
22 // hide other controls used to make language selection
23 promptLabel.Visible = false;
24 languageList.Visible = false;
25 submitButton.Visible = false;
26
27 // if the user made a selection, display it in responseLabel
28 if ( languageList.SelectedItem != null )
29     responseLabel.Text += " You selected " +
30         languageList.SelectedItem.Text;
31 else
32     responseLabel.Text += " You did not select a language.";
33
34 // display session ID
35 idLabel.Text = "Your unique session ID is: " + Session.SessionID;
36
37 // display the timeout
38 timeoutLabel.Text = "Timeout: " + Session.Timeout + " minutes.";
39 } // end if
40 } // end method Page_Load
41
```

Fig. 19.28 | Process user's selection of a programming language by displaying links and writing information in an HttpSessionState object. (Part 2 of 3.)



```
42 // record the user's selection in the Session
43 protected void submitButton_Click( object sender, EventArgs e )
44 {
45     // if the user made a selection
46     if ( languageList.SelectedItem != null )
47         // add name/value pair to Session
48         Session.Add( languageList.SelectedItem.Text,
49                     languageList.SelectedItem.Value );
50 } // end method submitButton_Click
51 } // end class Options
```

Fig. 19.28 | Process user's selection of a programming language by displaying links and writing information in an HttpSessionState object. (Part 3 of 3.)



Properties	Description
Count	Specifies the number of key/value pairs in the Session object.
IsNewSession	Indicates whether this is a new session (that is, whether the session was created during loading of this page).
Keys	Returns a collection containing the Session object's keys.
SessionID	Returns the session's unique ID.
Timeout	Specifies the maximum number of minutes during which a session can be inactive (that is, no requests are made) before the session expires. By default, this property is set to 20 minutes.

Fig. 19.29 | HttpSessionState properties.



Software Engineering Observation 19.1

A Web Form should not use instance variables to maintain client state information, because each new request or postback is handled by a new instance of the page. Instead, maintain client state information in `HttpSessionState` objects, because such objects are specific to each client.



Software Engineering Observation 19.2

A benefit of using `HttpSessionState` objects (rather than cookies) is that they can store any type of object (not just `Strings`) as attribute values. This provides you with increased flexibility in determining the type of state information to maintain for clients.



```
1 // Fig. 19.30: Recommendations.aspx.cs
2 // Creates book recommendations based on a Session object.
3 using System;
4
5 public partial class Recommendations : System.Web.UI.Page
6 {
7     // read Session items and populate ListBox with recommendations
8     protected void Page_Init( object sender, EventArgs e )
9     {
10         // determine whether Session contains any information
11         if ( Session.Count != 0 )
12         {
13             // display Session's name-value pairs
14             foreach ( string keyName in Session.Keys )
15                 booksListBox.Items.Add( keyName +
16                     " How to Program. ISBN#: " + Session[ keyName ] );
17         } // end if
```

Fig. 19.30 | Session data used to provide book recommendations to the user. (Part 1 of 2.)



```
18     else
19     {
20         // if there are no session items, no language was chosen, so
21         // display appropriate message and clear and hide booksListBox
22         recommendationsLabel.Text = "No Recommendations";
23         booksListBox.Items.Clear();
24         booksListBox.Visible = false;
25
26         // modify languageLink because no language was selected
27         languageLink.Text = "Click here to choose a language";
28     } // end else
29 } // end method Page_Init
30 } // end class Recommendations
```

Fig. 19.30 | Session data used to provide book recommendations to the user. (Part 2 of 2.)

19.8 Case Study: Database-Driven ASP.NET Guestbook



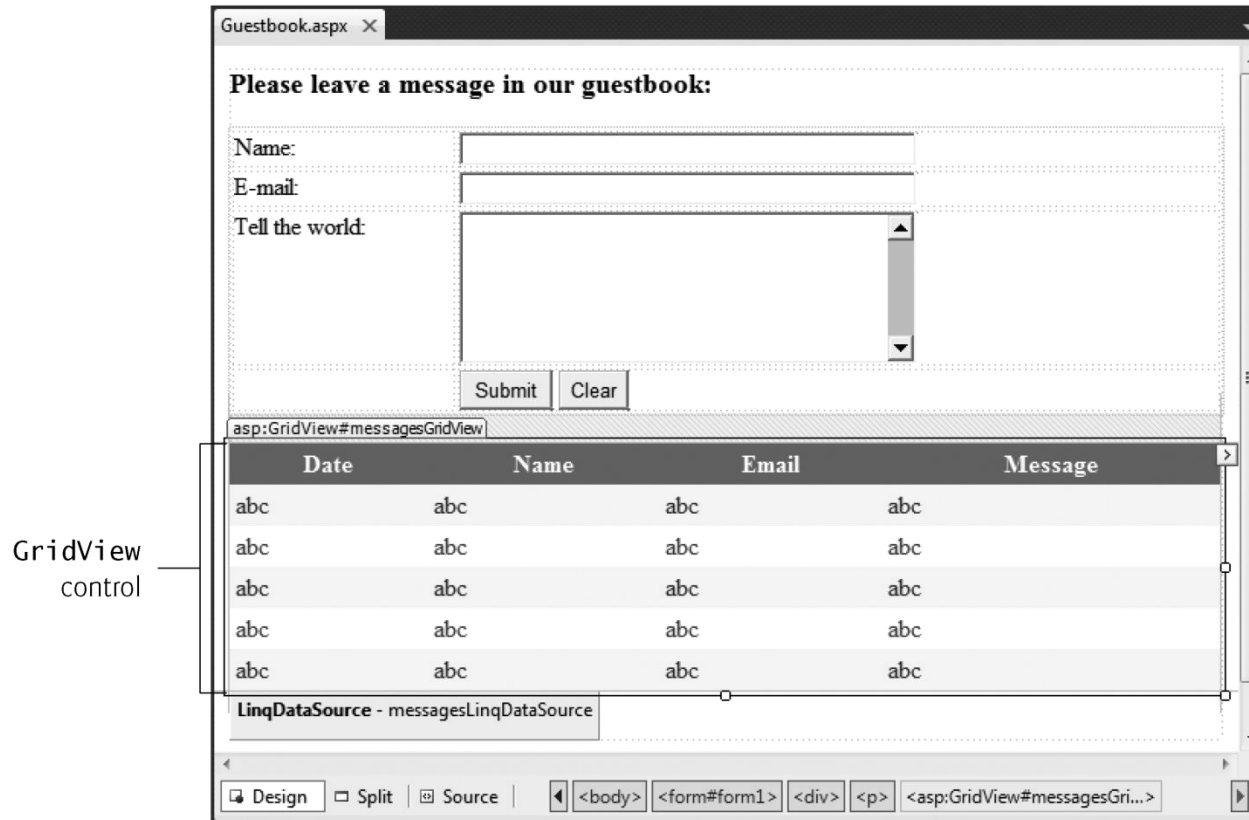


Fig. 19.31 | Guestbook application GUI in Design mode.



a) User enters data for the name, e-mail and message, then presses Submit to send the data to the server

Please leave a message in our guestbook:

Name:

E-mail:

Tell the world:

Date	Name	Email	Message
1/27/2010	Bob Green	bgreen@bug2bug.com	Great site!
1/28/2010	Sue Black	sblack@bug2bug.com	I love the site! Keep up the good work!
1/29/2010	Liz White	lwhite@bug2bug.com	Very useful information. Will visit again soon.

Guestbook.aspx Local intranet | Protected Mode: Off 100%

Fig. 19.32 | Sample execution of the Guestbook application.



b) Server stores the data in the database, then refreshes the GridView with the updated data

Please leave a message in our guestbook:

Name:

E-mail:

Tell the world:

Date	Name	Email	Message
1/27/2010	Bob Green	bgreen@bug2bug.com	Great site!
1/28/2010	Sue Black	sblack@bug2bug.com	I love the site! Keep up the good work!
1/29/2010	Liz White	lwhite@bug2bug.com	Very useful information. Will visit again soon.
7/22/2010	Mike Brown	mbrown@bug2bug.com	Wonderful use of ASP.NET!

Local intranet | Protected Mode: Off 100%

Fig. 19.32 | Sample execution of the Guestbook application.



Configure Data Source - messagesLinqDataSource

Configure Data Selection

Table:
Messages (Table<Message>)

GroupBy:
[None]

Select:

- ☒ *
- ☐ MessageID
- ☐ Date
- ☐ Name
- ☐ Email
- ☐ Message

Where...

OrderBy...

Advanced...

< Previous Next > Finish Cancel

Fig. 19.33 | Configuring the query used by the LinqDataSource to retrieve data.

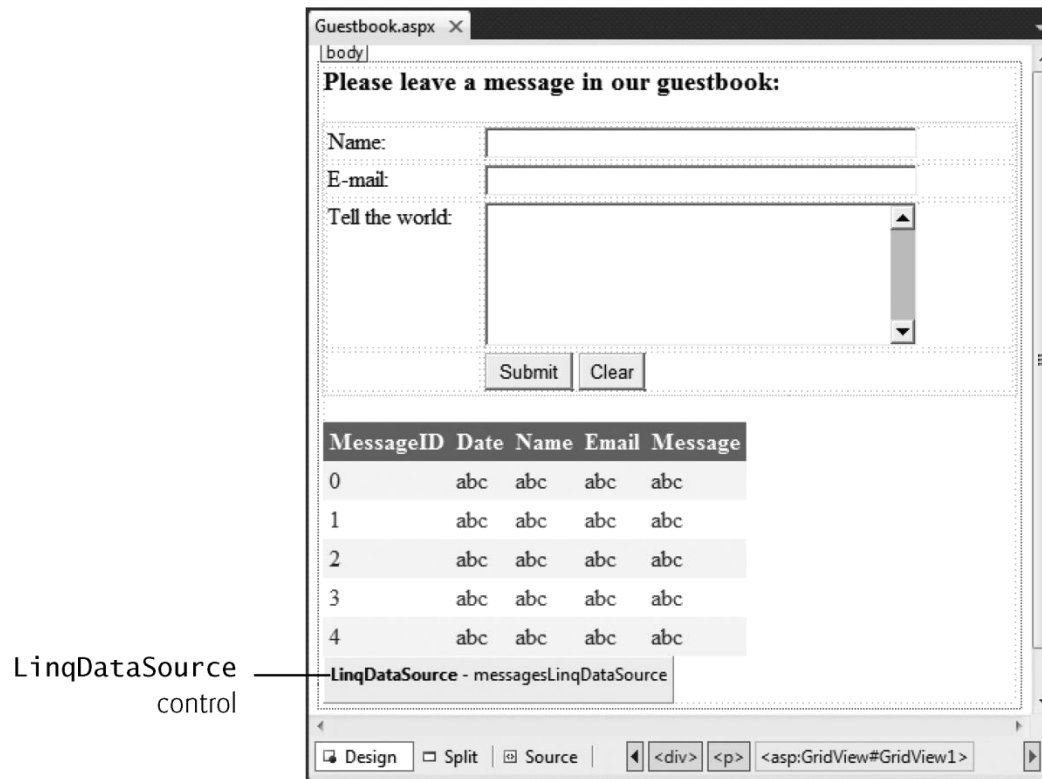


Fig. 19.34 | Design mode displaying LinqDataSource control for a GridView.

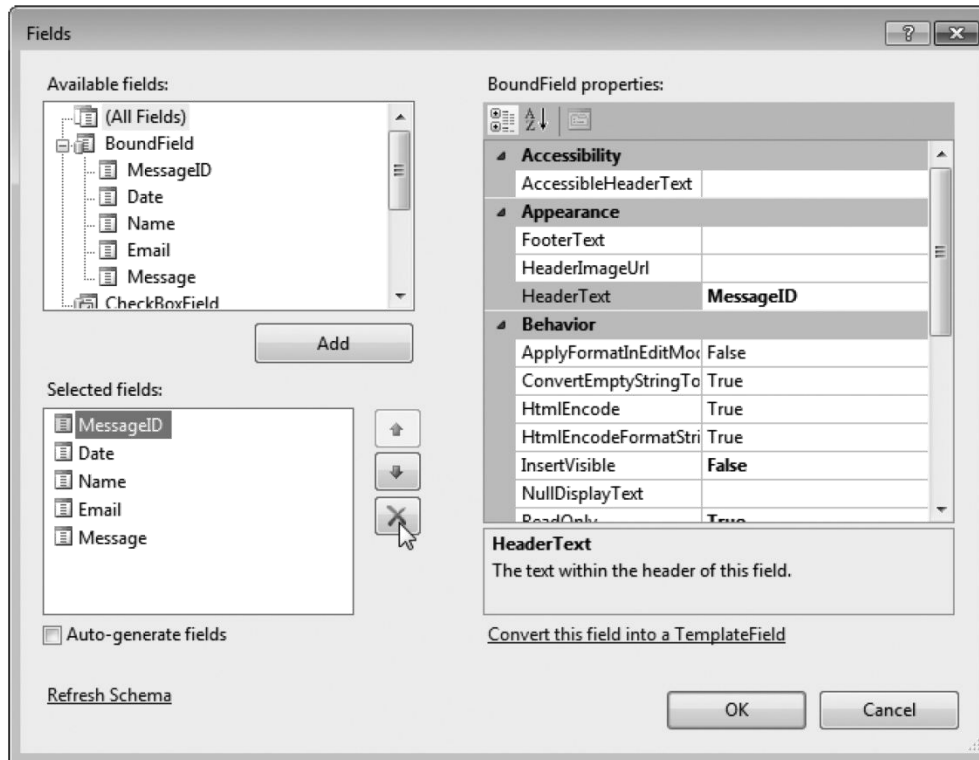


Fig. 19.35 | Removing the MessageID column from the GridView.



```
1 // Fig. 19.36: Guestbook.aspx.cs
2 // Code-behind file that defines event handlers for the guestbook.
3 using System;
4 using System.Collections.Specialized; // for class ListDictionary
5
6 public partial class Guestbook : System.Web.UI.Page
7 {
8     // Submit Button adds a new guestbook entry to the database,
9     // clears the form and displays the updated list of guestbook entries
10    protected void submitButton_Click( object sender, EventArgs e )
11    {
12        // create dictionary of parameters for inserting
13        ListDictionary insertParameters = new ListDictionary();
14
15        // add current date and the user's name, e-mail address
16        // and message to dictionary of insert parameters
17        insertParameters.Add( "Date", DateTime.Now.ToShortDateString() );
18        insertParameters.Add( "Name", nameTextBox.Text );
19        insertParameters.Add( "Email", emailTextBox.Text );
20        insertParameters.Add( "Message1", messageTextBox.Text );
21    }
```

Fig. 19.36 | Code-behind file for the guestbook application. (Part I of 2.)



```
22 // execute an INSERT LINQ statement to add a new entry to the
23 // Messages table in the Guestbook data context that contains the
24 // current date and the user's name, e-mail address and message
25 messagesLinqDataSource.Insert( insertParameters );
26
27 // clear the TextBoxes
28 nameTextBox.Text = String.Empty;
29 emailTextBox.Text = String.Empty;
30 messageTextBox.Text = String.Empty;
31
32 // update the GridView with the new database table contents
33 messagesGridView.DataBind();
34 } // submitButton_Click
35
36 // Clear Button clears the Web Form's TextBoxes
37 protected void clearButton_Click( object sender, EventArgs e )
38 {
39     nameTextBox.Text = String.Empty;
40     emailTextBox.Text = String.Empty;
41     messageTextBox.Text = String.Empty;
42 } // clearButton_Click
43 } // end class Guestbook
```

Fig. 19.36 | Code-behind file for the guestbook application. (Part 2 of 2.)