Chapter 21: Web App Development with ASP.NET in C#, A Deeper Look

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

Note: This chapter is a copy of Chapter 27 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-numb er them

OBJECTIVES

In this chapter you'll learn:

- To use the **Web Site Administration Tool** to modify web application configuration settings.
- To restrict access to pages to authenticated users.
- To create a uniform look-and-feel for a website using master pages.
- To use ASP.NET Ajax to improve the user interactivity of your web applications.

27.2.1 Examining the ASP.NET Web Site Template

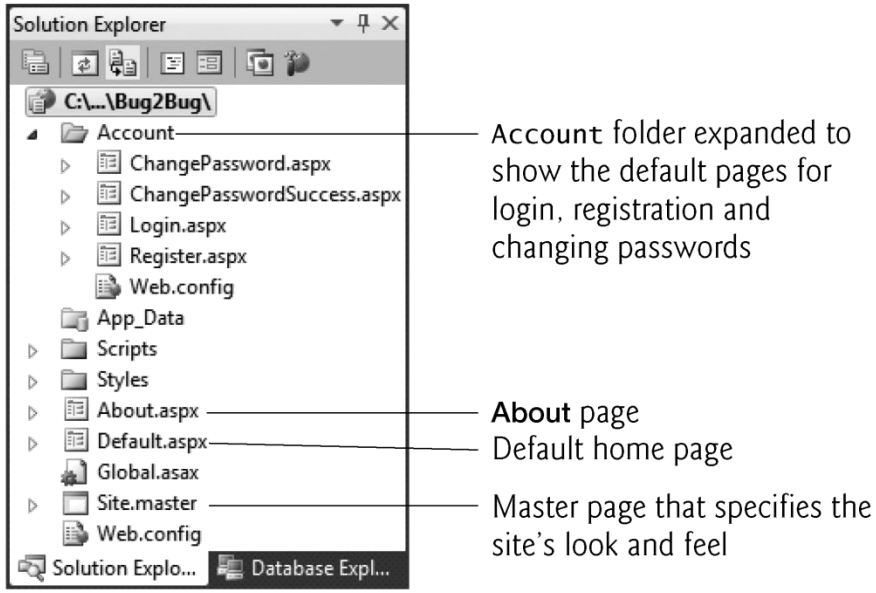

Fig. 27.1 | The default **ASP.NET Web Site** in the **Solution Explorer**.

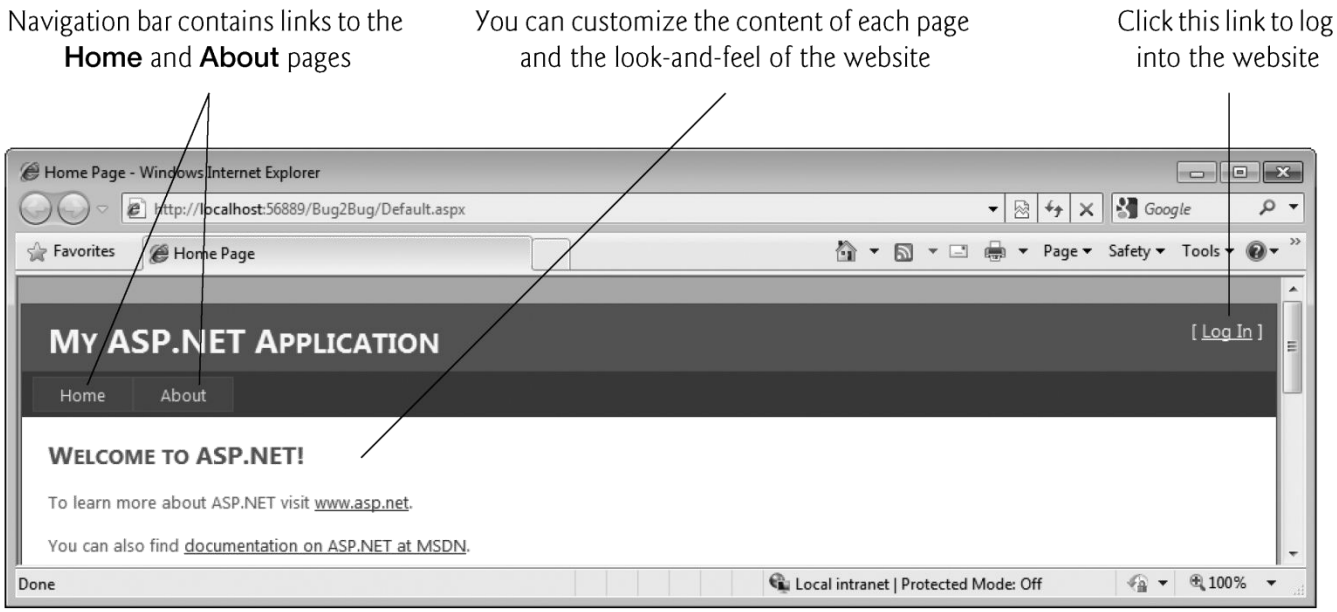

Fig. 27.2 | Default **Home** page of a website created with the **ASP.NET Web Site** template.

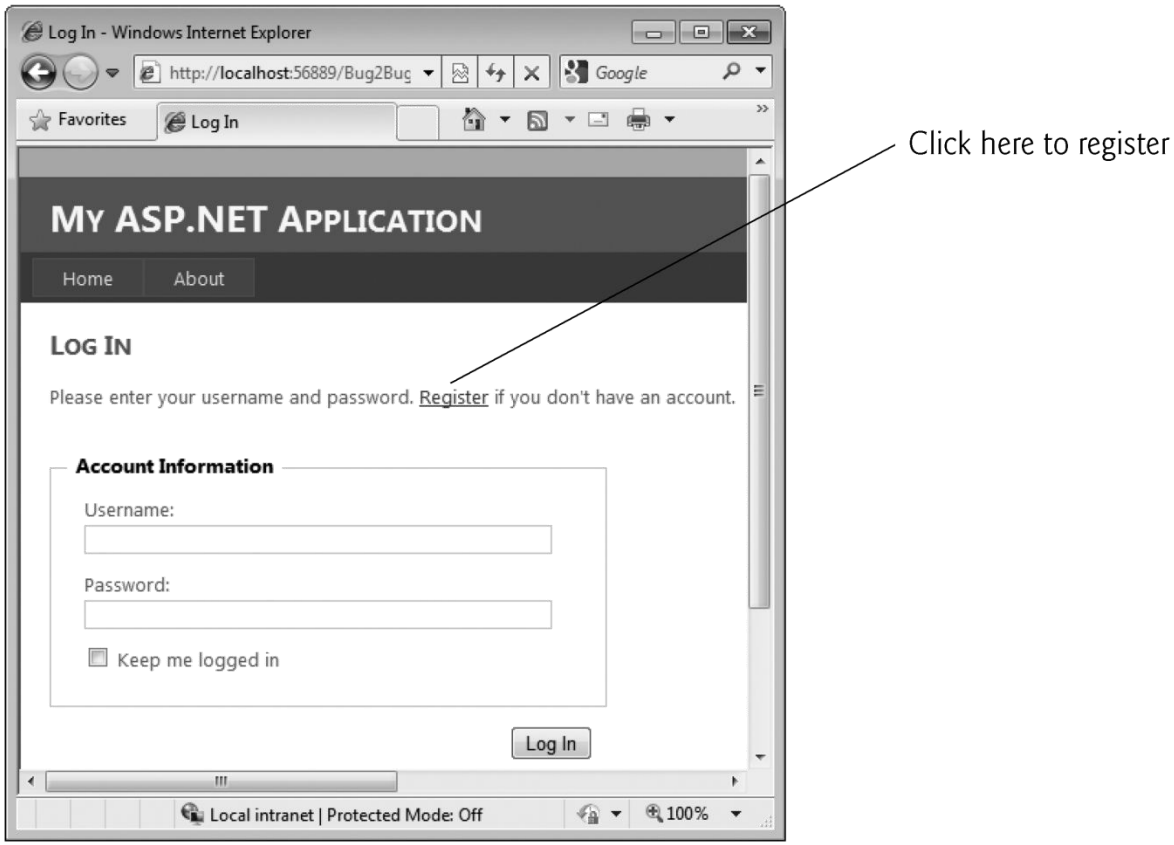

Fig. 27.3 | Login page.

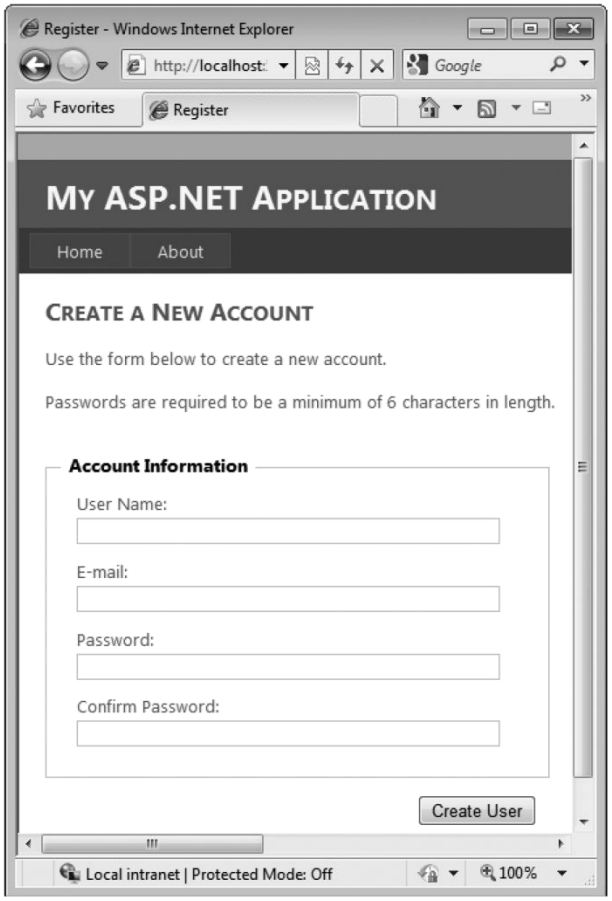

Fig. 27.4 | Register page.

27.2.2 Test-Driving the Completed Application

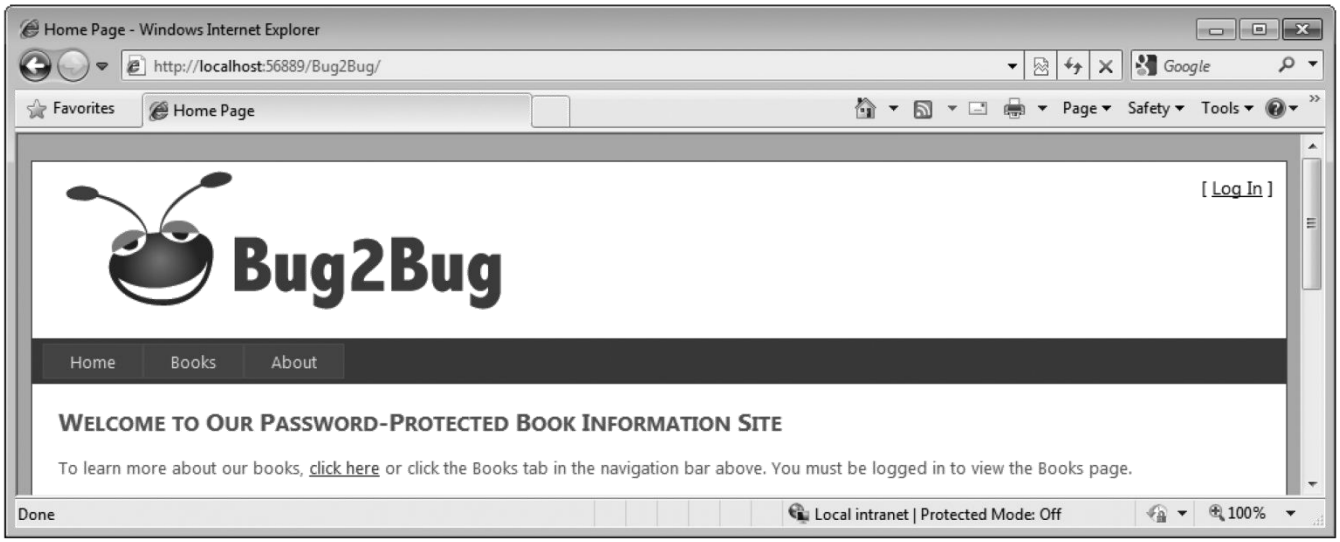

Fig. 27.5 | **Home** page for the completed Bug2Bug website.

Fig. 27.6 | Books.aspx displaying books by Harvey Deitel (by default).

Fig. 27.7 | Books.aspx displaying books by Greg Ayer.

27.2.3 Configuring the Website

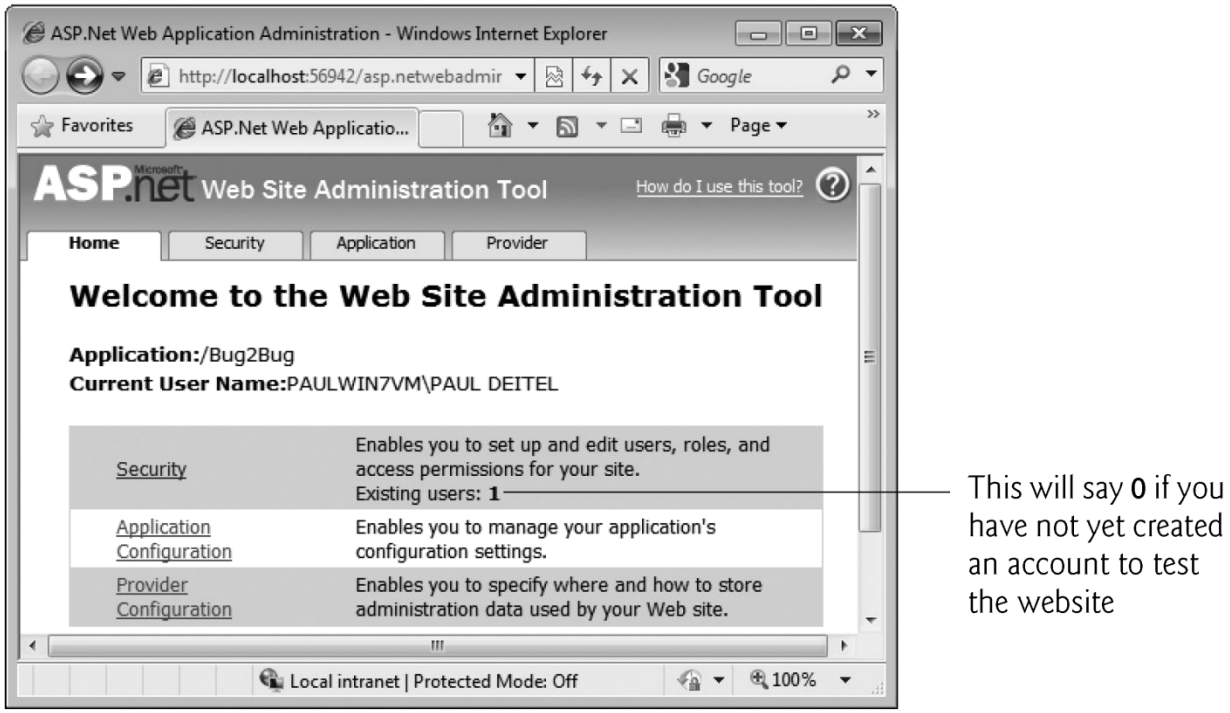

Fig. 27.8 | **Web Site Administration Tool** for configuring a web application.

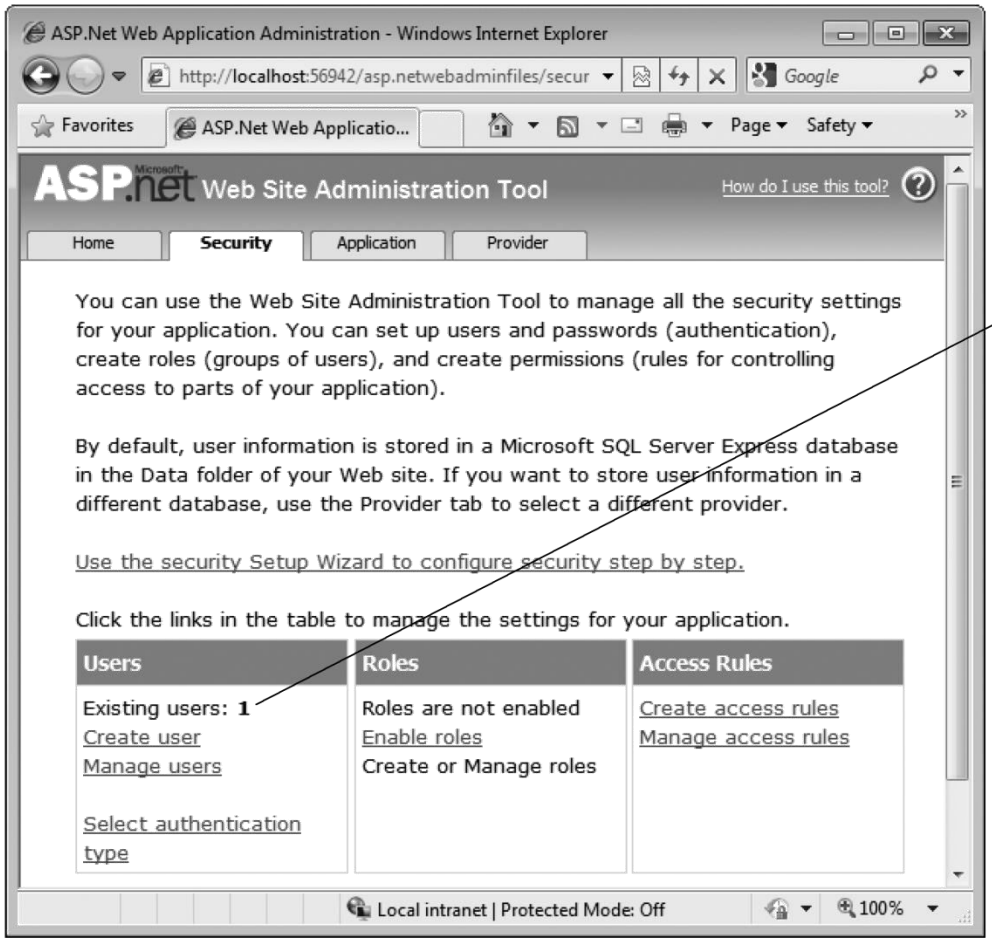

Fig. 27.9 | Security page of the Web Site Administration Tool.

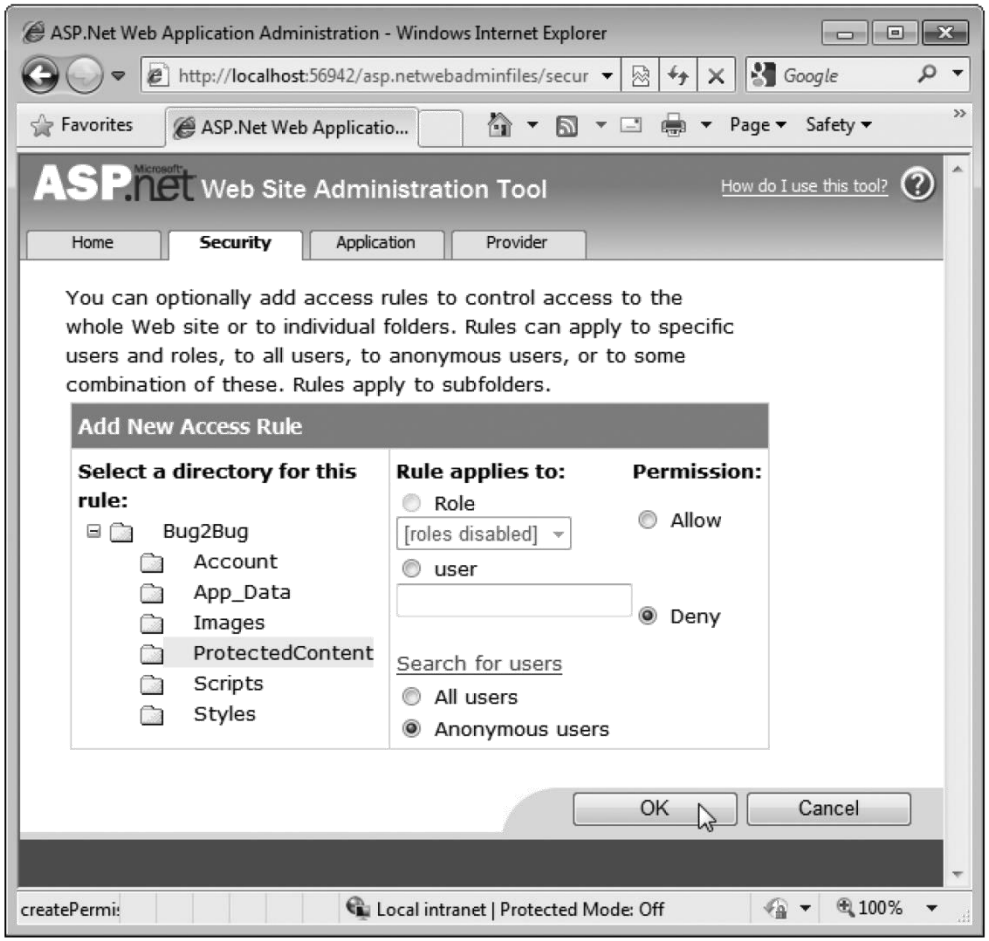

Fig. 27.10 | **Add New Access Rule** page used to configure directory access.

27.2.4 Modifying the Default.aspx and About.aspx Pages

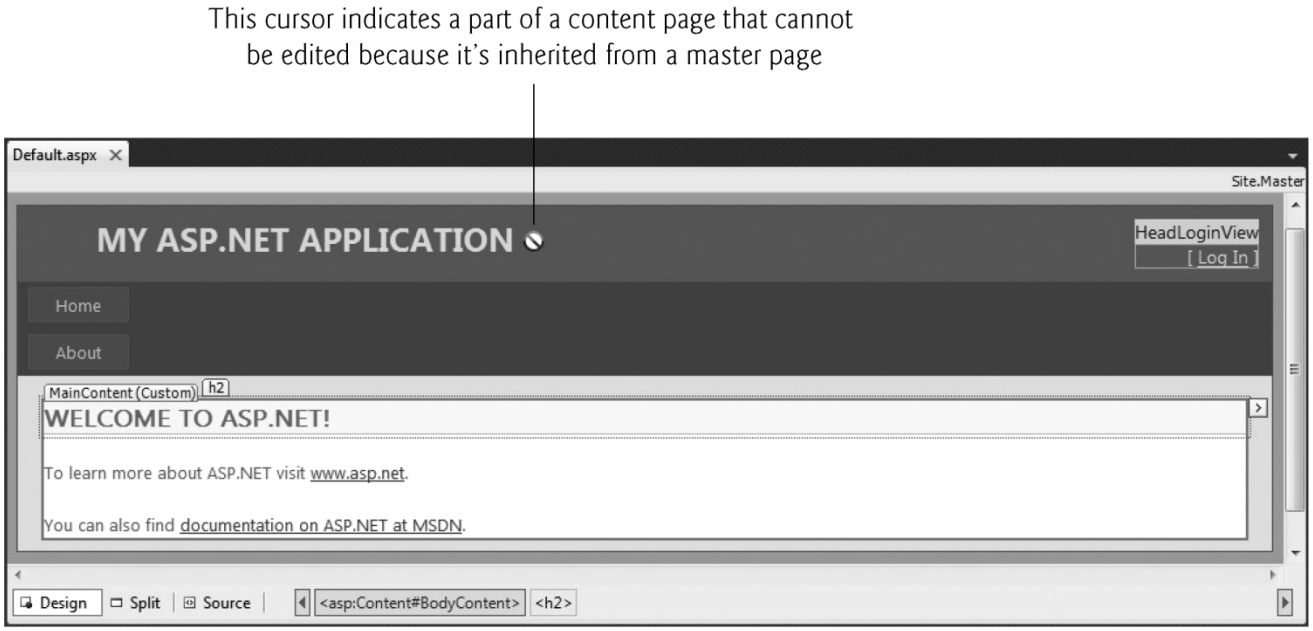

Fig. 27.11 | `Default.aspx` page in **Design** view.

27.2.6 Linking from the Default.aspx Page to the Books.aspx Page

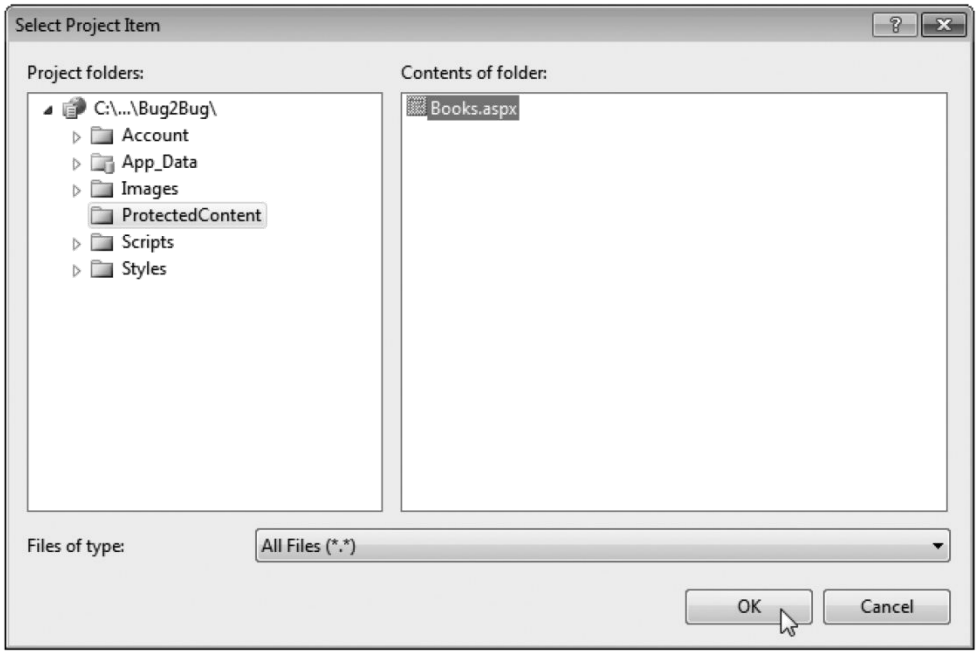

Fig. 27.12 | Selecting the Books.aspx page from the Select Project Item dialog.

27.2.7 Modifying the Master Page (Site.master)

Fig. 27.13 | Buttons for selecting parts of a page in **Design** view.

27.2.8 Customizing the Password-Protected Books.aspx Page

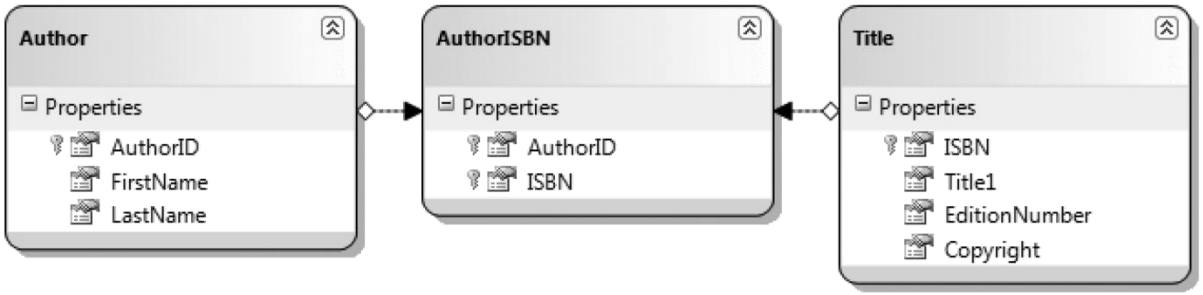

Fig. 27.14 | **Object Relational Designer** for the `Books` database.

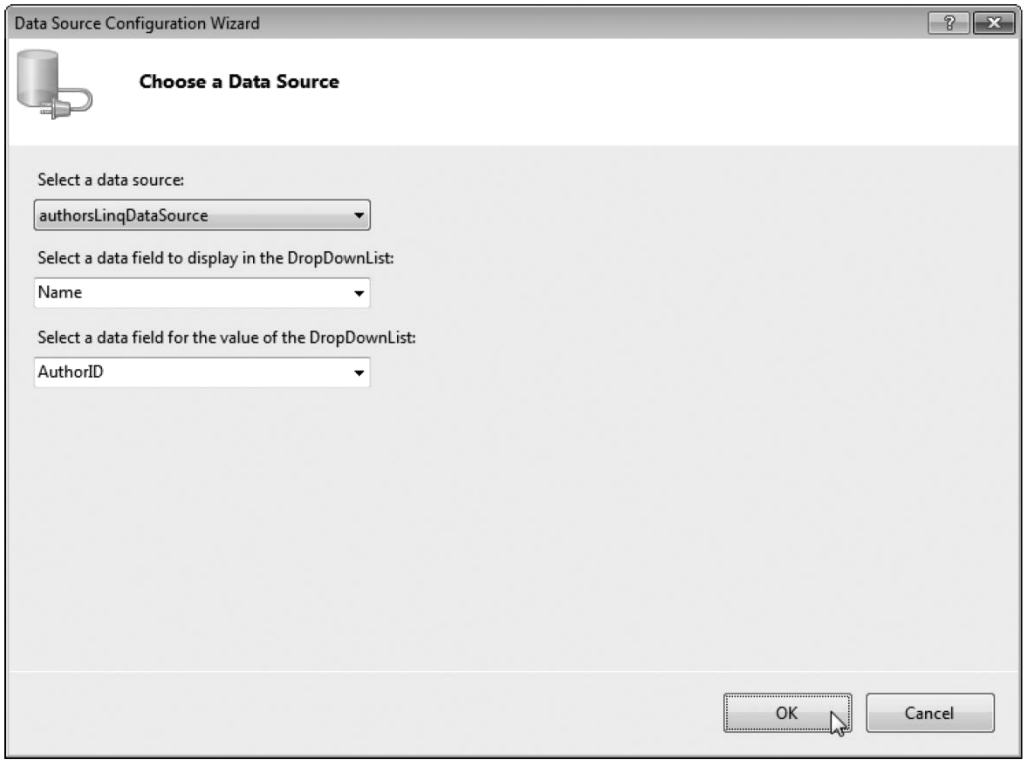

Fig. 27.15 | Choosing a data source for a `DropDownList`.

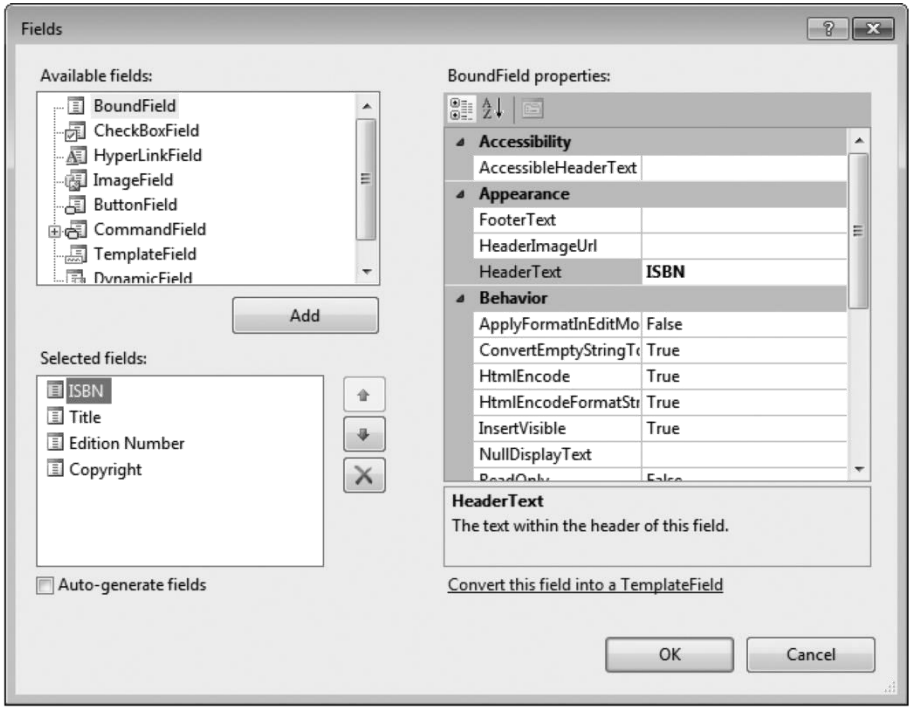

Fig. 27.16 | Creating `GridView` fields in the **Fields** dialog.

```
1  // Fig. 27.17: ProtectedContent_Books.aspx.cs
2  // Code-behind file for the password-protected Books page.
3  using System;
4  using System.Linq;
5  using System.Web.UI.WebControls;
6
7  public partial class ProtectedContent_Books : System.Web.UI.Page
8  {
9     // data context queried by data sources
10    BooksDataContext database = new BooksDataContext();
11
12    // specify the Select query that creates a combined first and last name
13    protected void authorsLinqDataSource_Selecting( object sender,
14       LinqDataSourceSelectEventArgs e )
15    {
16       e.Result =
17          from author in database.Authors
18          select new { Name = author.FirstName + " " + author.LastName,
19             author.AuthorID };
20    } // end method authorsLinqDataSource_Selecting
21
```

Fig. 27.17 | Code-behind file for the password-protected **Books** page. (Part 1 of 2.)

```
22      // specify the Select query that gets the specified author's books
23      protected void titlesLinqDataSource_Selecting( object sender,
24         LinqDataSourceSelectEventArgs e )
25      {
26         e.Result =
27            from book in database.AuthorISBNs
28            where book.AuthorID ==
29               Convert.ToInt32( authorsDropDownList.SelectedValue )
30            select book.Title;
31      } // end method titlesLinqDataSource_Selecting
32
33      // refresh the GridView when a different author is selected
34      protected void authorsDropDownList_SelectedIndexChanged(
35         object sender, EventArgs e )
36      {
37         titlesGridView.DataBind(); // update the GridView
38      } // end method authorsDropDownList_SelectedIndexChanged
39   } // end class ProtectedContent_Books
```

Fig. 27.17 | Code-behind file for the password-protected **Books** page. (Part 2 of 2.)

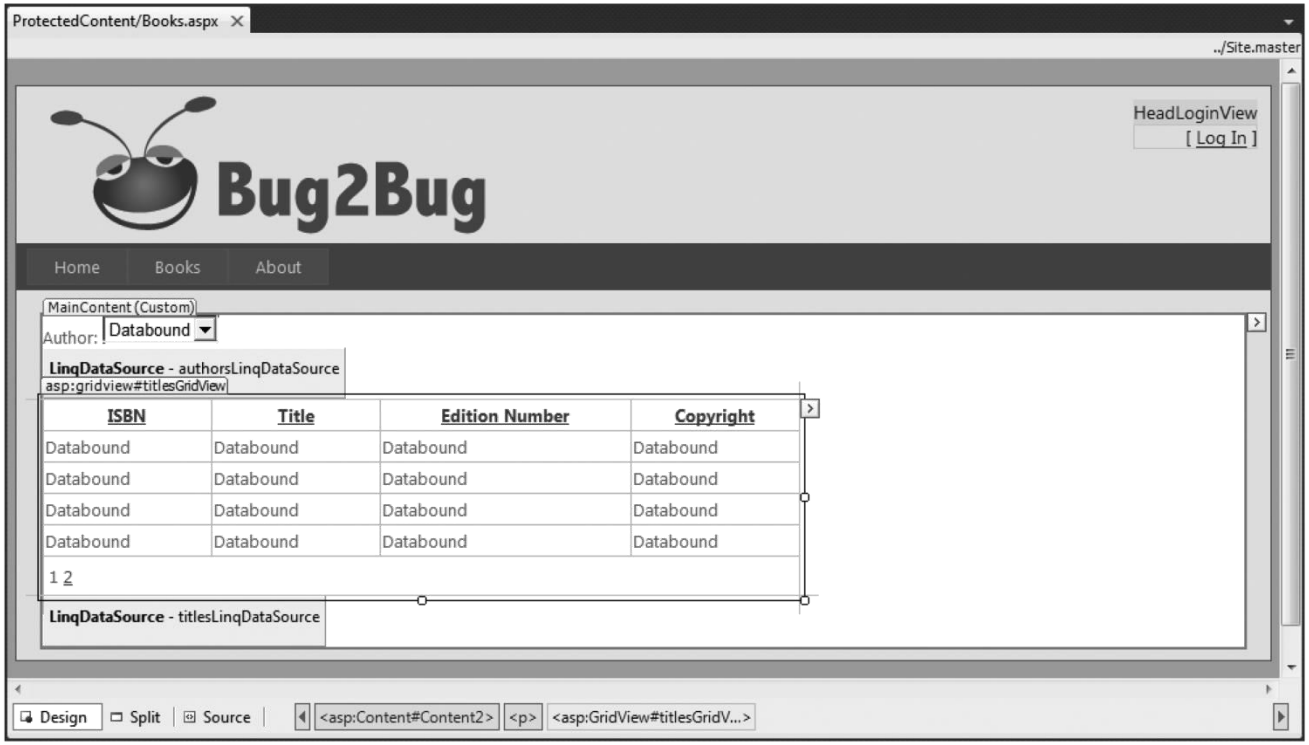

Fig. 27.18 | Completed `Books.aspx` page in `Design` mode.

27.3.1 Traditional Web Applications

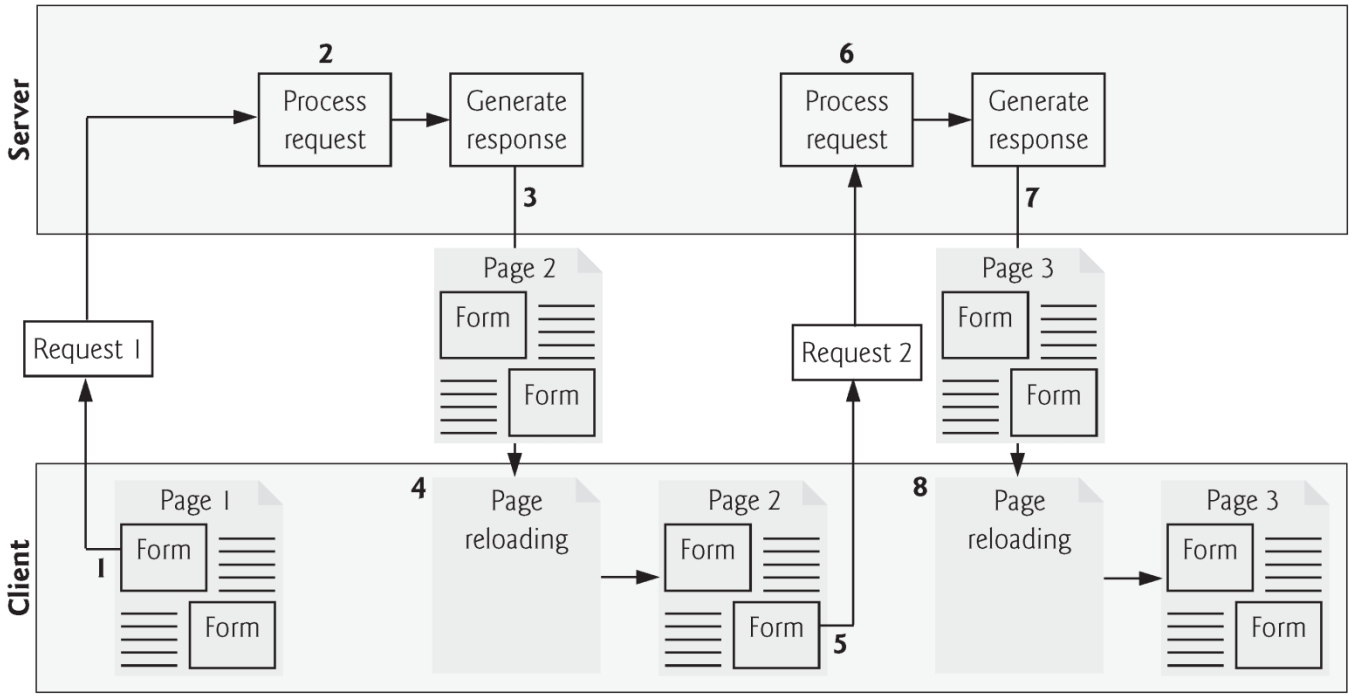

Fig. 27.19 | Traditional web application reloading the page for every user interaction.

27.3.2 Ajax Web Applications

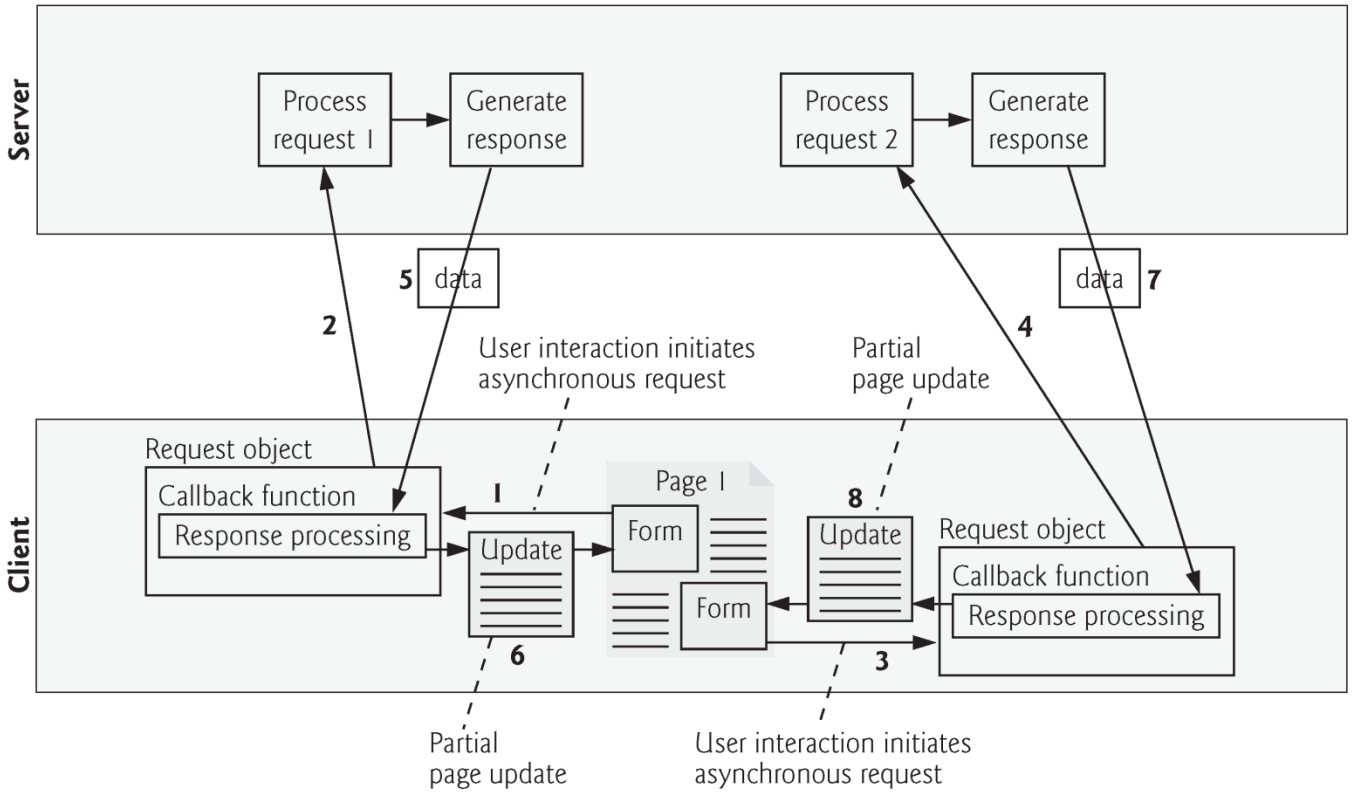

Fig. 27.20 | Ajax-enabled web application interacting with the server asynchronously.

27.3.3 Testing an ASP.NET Ajax Application

a) Entering a name on the **Name** tab then clicking the **Contact** tab

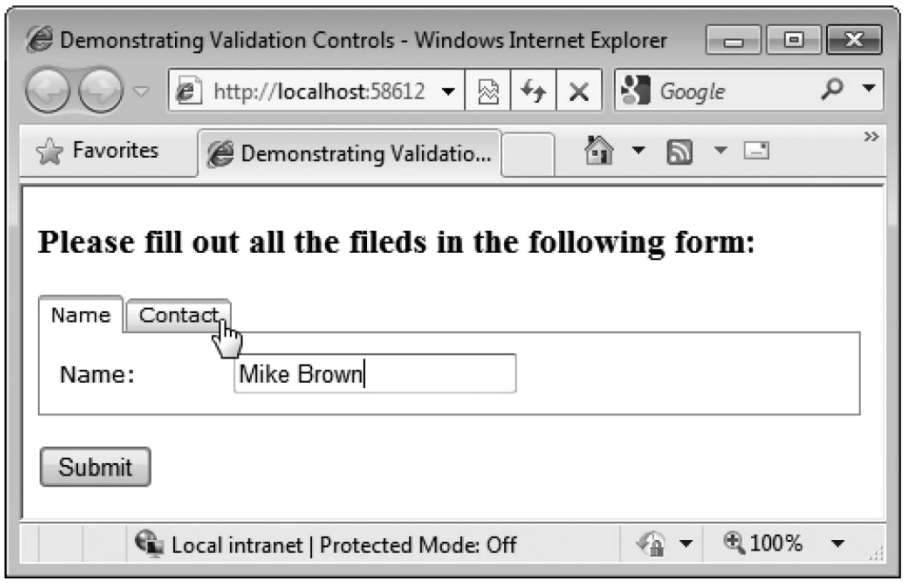

Fig. 27.21 | Validation application enhanced by ASP.NET Ajax. (Part 1 of 3.)

b) Entering an e-mail address in an incorrect format and pressing the *Tab* key to move to the next input field causes a callout to appear informing the user to enter an e-mail address in a valid format

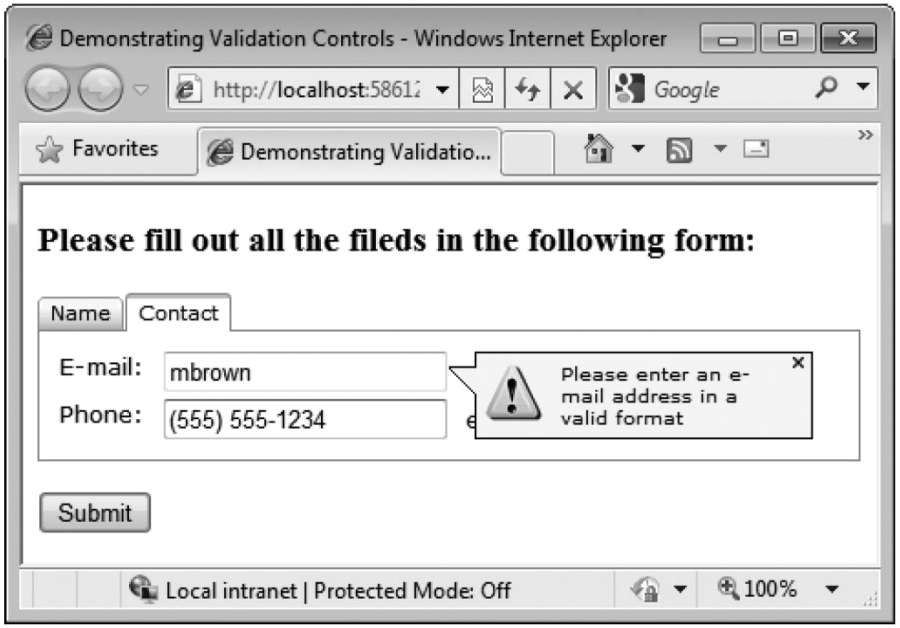

Fig. 27.21 | Validation application enhanced by ASP.NET Ajax. (Part 2 of 3.)

c) After filling out the form properly and clicking the **Submit** button, the submitted data is displayed at the bottom of the page with a partial page update

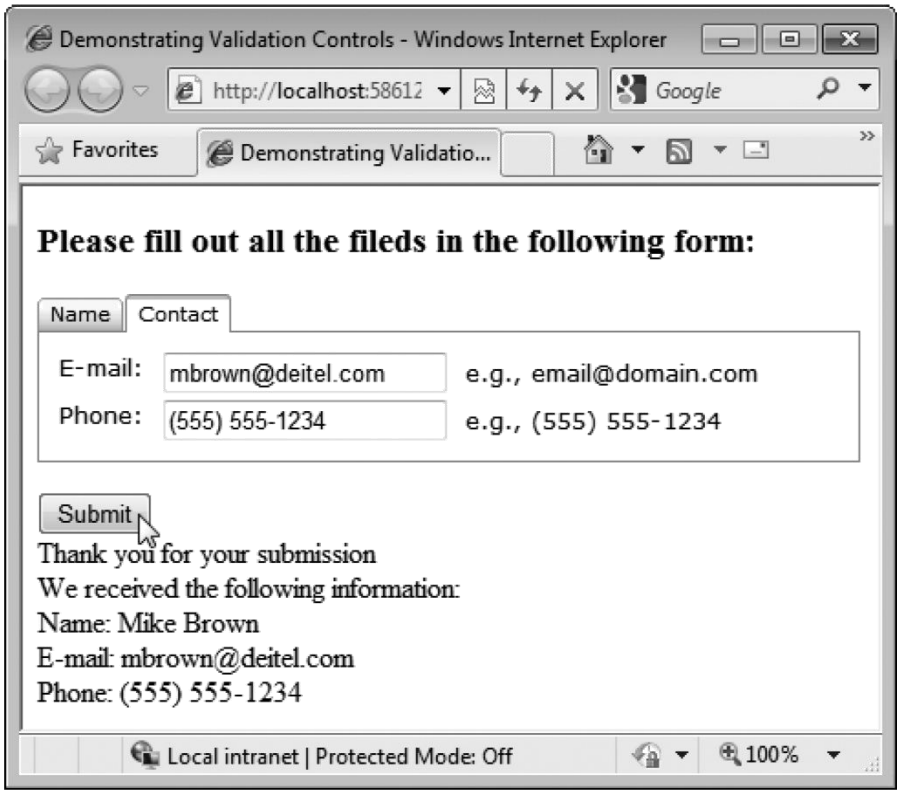

Fig. 27.21 | Validation application enhanced by ASP.NET Ajax. (Part 3 of 3.)

27.3.5 Using Controls from the Ajax Control Toolkit

Common Programming Error 27.1

Putting more than one `ScriptManager` and/or `ToolkitScriptManager` control on a Web Form causes the application to throw an `InvalidOperationException` when the page is initialized.

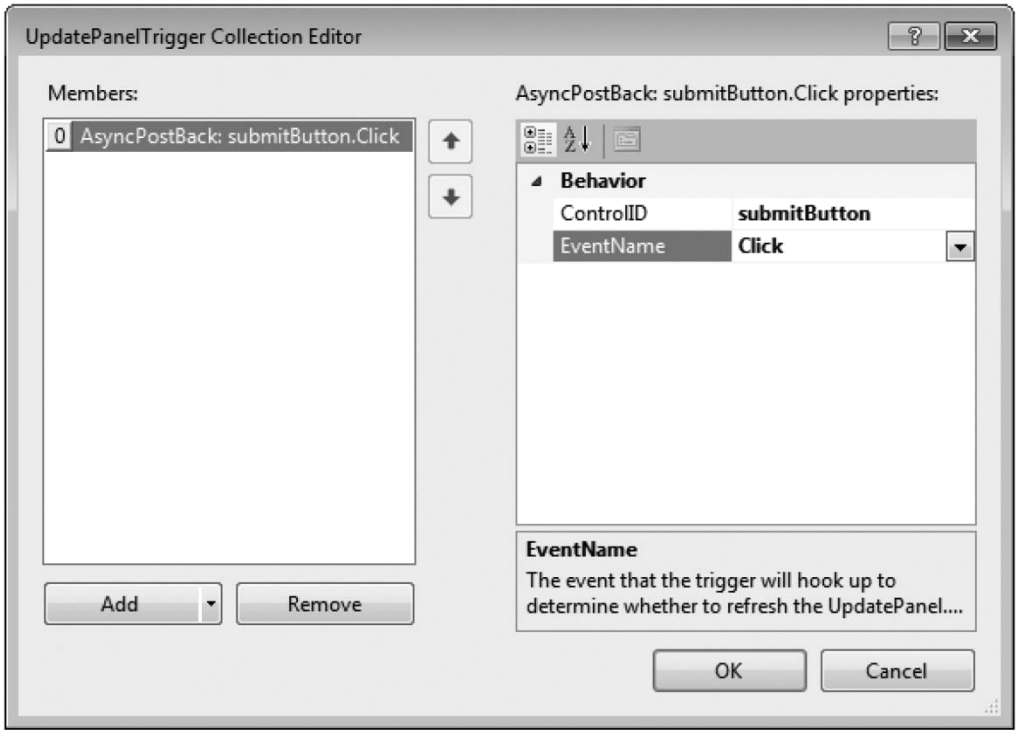

Fig. 27.22 | Creating a trigger for an `UpdatePanel`.

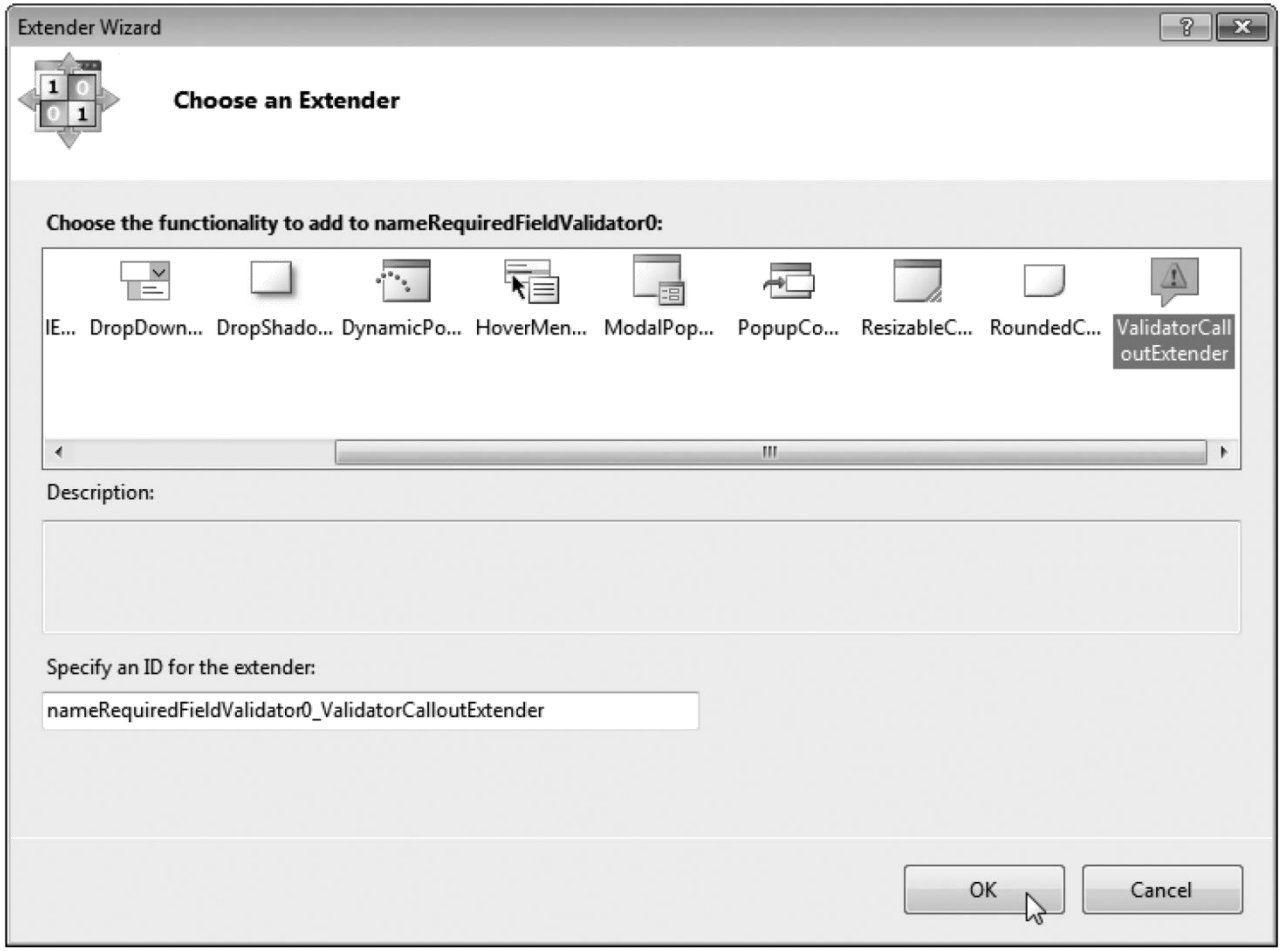

Fig. 27.23 | Creating a control extender using the **Extender**