Corrections for
*Murach’s ASP.NET 4.5 Web Programming with C# 2012*

These are the corrections for the significant errors in each printing of this book. In addition to the corrections listed here, you may find some trivial typos and formatting errors. All types of corrections will be made in the next printing of the book.

**How to tell which printing your book is in**

Below the copyright notation on the back of the title page (page iv), you’ll find a series of numbers like this:

10 9 8 7 6 5 4 3 2

The number on the right of this sequence tells which printing your book is. In this example, it’s the second printing (September 2014).
Corrections to the second printing

Chapter 9, page 303

The second bullet says that to create a content page with a name other than Default.aspx, you start by right-clicking on the master page in the Solution Explorer. It should say that you right-click on the project in the Solution Explorer.
Corrections to the first printing

Chapter 2, page 67
The expression for calculating the future value in the CalculateFutureValue method is incorrect. The addition of the monthly investment to the future value should be enclosed in a parenthetical like this:

\[ \text{futureValue} = (\text{futureValue} + \text{monthlyInvestment}) \times (1 + \text{monthlyInterestRate}) \]

Chapter 5, page 187
In step 3 of exercise 5-1, the second sentence says to point to the IsPostBack property to see that its value in the data tip is null. It should say that the data tip is false.

Chapter 6, page 198
The second sentence in the second to last paragraph says that the TextMode attribute can be set to tel if the text box is supposed to get a telephone number. It should say that this attribute can be set to Phone.

Chapter 6, page 199
The TextMode values for the HTML5 type attributes for input elements should all start with a capital letter. In addition, the tel attribute should be Phone.

Chapter 6, page 210
The first sentence of the second paragraph under the heading “How to create drop-down lists and list boxes” says that the Value attribute for a form only has to be coded when the value is different from the content for the list item. It should say that the Value attribute for a list item only has to be coded when the value is different from the content for the list item.

Chapter 6, page 216
The last sentence of the second paragraph should refer to the second example, not the first. Similarly, the third sentence of the third paragraph should refer to the first example, not the second.

Chapter 12, page 379
The first bullet indicates that the CategoryID in the Categories table is a 10-character code. It should say that CategoryID is a variable-length code of up to 10 characters. Similarly, the second bullet indicates that the ProductID in the Products table is a variable-length code of up to 10 characters, and it should say that ProductID is a 10-character code.

Chapter 15, page 508
The second sentence of the third paragraph says that you can use CSS as described in chapter 5. It should refer you to chapter 3 instead.

Chapter 16, page 549
Step 2 of exercise 16-1 says to add a ListView control to the main division of the page. It should say to add a ListView control to the form.

Chapter 20, page 661
The second method in the code on this page should be named GetConfirmationMessage as indicated in the description on page 660, not GetConfirmation. This affects the method declaration as well as the two references to the method in the SendConfirmation method.
Chapter 20, page 662
The first sentence of the last paragraph should refer to the GetConfirmationMessage method, not the GetConfirmation method.

Chapter 20, page 663
The second method in the code on this page should be named GetConfirmationMessage, not GetConfirmation. This affects the method declaration as well as the two references to the method in the SendConfirmation method. In addition, the Customer parameter of the GetConfirmationMessage method should be named c, not customer, so it’s consistent with the code on page 661.

Chapter 20, page 669
The ErrorHandler method in the ErrorHandler class shown in this figure won’t work if the error is caught by a try-catch block. That’s because the catch block automatically retrieves that last exception, which means it’s no longer available when the Server.GetLastError() method is called in the ErrorHandler class. To fix this problem, the catch block must pass the exception to the ErrorHandler method. Then, this method must check the exception parameter to see if it’s null, indicating that the error was caught at the page or application level rather than at the method level. If it is null, it should then retrieve the exception object using the Server.GetLastError() method. Here’s the revised code for making this work:

```csharp
private Exception ex;
public ErrorHandler(Exception e) {
    if (e == null) e = HttpContext.Current.Server.GetLastError();
    if (e.InnerException == null) ex = e;
    else ex = e.InnerException;
}
```

Chapter 22, page 730
The second sentence of the second to last paragraph refers to the DataKeys property of the GridView control. It should refer to the DataKeyNames property.

Chapter 23, page 741
The using statement for the Ch23ShoppingCartWCF.DataAccess namespace isn’t required and should be deleted.

Appendix A, pages 796 and 797
The second sentence of the last paragraph on page 796 refers to the C:\Murach\aspnet45_cs\exercise_starts directory. It should refer to the C:\Murach\aspnet45_cs\exercises directory. The same is true of the last bullet on page 797.