

note

About PostBack

You learned earlier that JavaScript is client-side code, while the C# code in code-behind files is server-side code. Server-side code can only run when the browser posts back to the server.

One advantage of this is that server-side code can't be seen by visitors to your site. C# server-side code can also do some things that JavaScript can't, such as accessing a database.

The down-side of PostBack is that the user has to send a request back to the server every time they want the page to update and the whole page needs to be re-loaded. If you have large pages, this can make them significantly slower.

JavaScript can change pages without posting back to the server, which makes it much faster. However, it is less secure than server-side code and has some limitations.

Using web services, it is possible for JavaScript and C# to work together and only update certain parts of the page. This is known as AJAX.

AJAX isn't covered by this book, but you'll learn all about it in the next book in this series: *Learn ASP.NET 4.0, C# and Visual Studio 2010 Expert Skills with The Smart Method*.

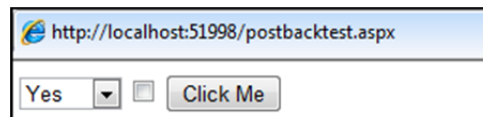
Lesson 3-8: Understand PostBack

In *Lesson 2-12: Work with HTML Forms*, you created an HTML form that sent (or *posted*) data to another page. ASP.NET normally uses a form to *post* data back to the same page; this is called *PostBack*.

By using *PostBack*, the user sends data to your web site without moving between pages. ASP.NET's controls expect this, and you can use it to your advantage.

- 1 Open *CSharpTest* from your sample files folder.
- 2 Open the code-behind file of *postbacktest.aspx*.
- 3 See PostBack in action.

1. View *postbacktest.aspx* in your browser.



2. Click the button that says *Click Me*.



You should notice a brief 'flicker' as the page reloads. This is because the page's form was submitted and the data 'posted back'.

3. Choose an item from the drop-down list.



You'll notice that this time there was no flicker. By default, changing the value of a drop-down doesn't cause the form to be submitted.

4. Click the check-box.



You'll notice this doesn't cause a post-back either.

5. Close your web browser and return to Visual Studio.
- 4 Use C# code to check if the page has posted back.

1. Add the following code to the *Page_Load* event handler:

```
LabelIsPostBack.Text = Page.IsPostBack.ToString();
```

```
protected void Page_Load(object sender, EventArgs e)
{
    LabelIsPostBack.Text = Page.IsPostBack.ToString();
}
```

Page.IsPostBack is a value that tells you whether the page has been posted back. *ToString* converts the value into text.

2. View *postbacktest.aspx* in your browser.



You will see that the label at the top of the page says *False*. This means that the page currently hasn't been posted back. The user has just loaded the page and hasn't clicked on any controls yet.

- Click the button.



You'll see that the label changes to say *true*. This means the user has submitted the form and posted back some data.

This is useful because it allows you to make some of your code only run when the user has posted back. You'll see how this can be done in: *Lesson 7-1: Use the if statement*.

- Close your web browser.

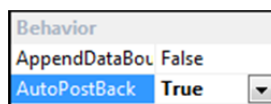
5 Make a non-PostBack control post back.

Earlier you saw that the *DropDownList* and *CheckBox* controls don't automatically post back when you click on them. If your page needs to be updated when the user changes one of these controls, you might want them to post back. Fortunately, it can be done.

- Open *postbacktest.aspx* in *Design* view.
- Select the *DropDownListPostBack* control.



- Set the control's *AutoPostBack* property to: **True**



You learned how to do this in: *Lesson 1-12: Change properties in Design view*.

- View the page in your browser.
- Try changing the value of the drop-down list.



You'll see that this time the page posts back and the label at the top confirms this.

The *AutoPostBack* property is available on most controls that don't automatically post back.

6 Close Visual Studio.

note

How ASP.NET posts back without a button

In the HTML form you created in *Lesson 2-12: Work with HTML Forms*, the only way to submit the form was to click a button.

In this lesson you've seen that ASP.NET can submit its form when other things happen. It does this by automatically generating JavaScript to submit the form.

If you open the page in your browser and view the source (right-click and *View Source* in Internet Explorer), you'll see that ASP.NET has added lines with *javascript:__doPostBack*. This JavaScript causes the form to be submitted.

Because JavaScript is client-side code, you should always be aware that someone browsing your site might have disabled JavaScript, in which case the post-back will not work.