**Redirecting Users to Another Page**

**Microsoft Visual Studio 2008/.NET Framework 3.5**

Other versions are also available for the following:

[Microsoft Visual Studio 2005/.NET Framework 2.0](http://msdn.microsoft.com/en-us/library/x3x8t37x(VS.80).aspx)

[.NET Framework 3.0](http://msdn.microsoft.com/en-us/library/x3x8t37x(VS.85).aspx)

[Microsoft Visual Studio 2010/.NET Framework 4.0](http://msdn.microsoft.com/en-us/library/x3x8t37x(VS.100).aspx)

You will often want to redirect users to other pages as part of your Web application. ASP.NET provides the following ways for you to build redirection into your Web pages:

* Using hyperlinks on pages.
* Configuring cross-page posting, which enables you to specify an alternate target page when the current page is submitted.
* Redirecting programmatically by forcing the browser to request a different page.
* Redirecting programmatically by transferring control to a different page in the same Web application.

Each of these options is described below. A table at the end of the topic summarizes the options and provides guidelines to help you decide when to use each option.

http://i.msdn.microsoft.com/Global/Images/clear.gif Hyperlinks

You can use an HTML anchor tag (**<a>**) on an ASP.NET Web page to create static links, or you can programmatically control the link text and target URL of hyperlinks using the [HyperLink](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.hyperlink.aspx) control. In this scenario, the user explicitly clicks a link and the browser transfers to the target page. The target page is invoked using an HTTP GET command. Therefore, no information about the source page is passed to the target page unless you specify a query string on the URL of the target page. If the source and target page are in the same Web application, they can share information using session state or application state.

http://i.msdn.microsoft.com/Global/Images/clear.gif Cross-Page Posting

By default, buttons in an ASP.NET Web page post the page to itself. Cross-page posting enables you to configure a button on an ASP.NET Web page to post the current page to a different page. A typical example is when creating a multi-page form. You can configure buttons on the page to move to the next and previous pages of the form.

Cross-page posting is similar to hyperlinks in that the transfer is initiated by a user action. However, in cross-page posting, the target page is invoked using an HTTP POST command, which sends the values of controls on the source page to the target page. In addition, if the source and target page are in the same Web application, the target page can access public properties of the source page. As always, all of the pages in the application can share information stored in session state or application state.

For more information, see [Cross-Page Posting in ASP.NET Web Pages](http://msdn.microsoft.com/en-us/library/ms178139.aspx) and [How to: Post ASP.NET Web Pages to a Different Page](http://msdn.microsoft.com/en-us/library/ms178140.aspx).

http://i.msdn.microsoft.com/Global/Images/clear.gif Redirecting Programmatically Using the Browser

You can redirect users to another page using the capabilities of the user's browser. In a browser redirect, the browser issues a new request to the target server in the form of an HTTP GET request.

You can trigger the redirect programmatically in client script or server code. In client script, you can call the **form.submit** method, provided the **<form>** element's **method** attribute value is **get**. In that case, if the current page contains form data, it is passed to the target server by appending it as a query string onto the requested URL.

In server code, you can programmatically redirect by calling the [Redirect](http://msdn.microsoft.com/en-us/library/system.web.httpresponse.redirect.aspx) method. The method sends a command to the user's browser that causes the browser to issue an HTTP GET command for the target page. Calling the server **Redirect** method is the programmatic equivalent of clicking a hyperlink, in that it results in a new request for the target page. Because you are calling the methods from your own code, you can dynamically define the target URL, including any query-string information. If the source and target pages are in the same Web application, you can share data between the source and target pages by adding server code to store the data in session state.

|  |
| --- |
| **NoteNote:** |
| Internet Explorer through version 6.0 can process only up to 2,048 characters in the URL, including data in the query string. If the URL exceeds 2,048 characters, an error may result, or data in the query string might be truncated or not sent with the request. In ASP.NET Web pages, a GET request that includes post data can easily exceed 2,048 characters if view state information (which is stored in a hidden field) is part of the request, resulting in errors. Other browsers may not have this limitation. For more information, see article 208427, "Maximum URL Length Is 2,083 Characters in Internet Explorer" in the Microsoft Knowledge Base at http://support.microsoft.com. To work around this limitation when you are trying to share information among pages, you can redirect users by using HTTP POST requests as discussed in this topic. If your application requires an HTTP GET request, you can store information in an alternative way, without using a query string, such as in session state. |

http://i.msdn.microsoft.com/Global/Images/clear.gif Redirecting Programmatically on the Server

You can also redirect programmatically to a target page on the server by calling the [Transfer](http://msdn.microsoft.com/en-us/library/system.web.httpserverutility.transfer.aspx) method. In this scenario, the server simply transfers the current source page context to the target page. The target page then renders in place of the source page. The source and target pages must be in the same Web application. As with cross-page posting, the **Transfer** method has the advantage that it enables the source page to read control values and public property values from the source page.

Because the transfer between source and target pages happens on the server, the browser has no information about the changed page, and it retains information about the original (source) URL. For example, the **Address** box in Internet Explorer does not change after a transfer, and instead continues to show the URL of the page it most recently requested (which is usually the source page). The browser's history is not updated to reflect the transfer. This can result in unexpected behavior if the user refreshes the page in the browser or clicks the browser's back button. Therefore, calling the **Transfer** method is a strategy that is best used in applications where you are presenting pages to the user with the URL hidden.

http://i.msdn.microsoft.com/Global/Images/clear.gif Selecting a Redirect Option

The following table summarizes the possible ways to redirect between pages.

|  |  |  |
| --- | --- | --- |
| **Strategy** | **Characteristics** | **Usage** |
| Hyperlinks | * Performs new request on the target page. * Does not pass current page information to the target page. * Requires user initiation. * Redirects to any page, not just pages in the same Web application. * Enables you to share information between pages using query string or session state. (The [HyperLink](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.hyperlink.aspx) control enables you to create URL and query strings programmatically.) | * For navigation without any additional processing, as in menus or lists of links. * When navigation should be under user control. |
| Cross-page posting | * Posts current page information to the target page. * Makes post information available in the target page. * Requires user initiation. * Redirects to any page, not just pages in the same Web application. * Enables the target page to read public properties of the source page if the pages are in the same Web application. | * To pass current page information to the target page (as in multi-page forms). * When navigation should be under user control. |
| Browser redirect | * Performs a new HTTP GET request on the target page. * Passes the query string (if any) to the target page. In Internet Explorer, the size of the query string is limited to 2,048 characters. * Provides programmatic and dynamic control over the target URL and query string. * Enables you to redirect to any page, not just pages in the same Web application. * Enables you to share information between source and target pages using session state. | * For conditional navigation, when you want to control the target URL and control when navigation takes place. For example, use this option if the application must determine which page to navigate to based on data provided by the user. |
| Server transfer | * Transfers control to a new page that renders in place of the source page. * Redirects only to target pages that are in the same Web application as the source page. * Enables you to read values and public properties from source page. * Does not update browser information with information about the target page. Pressing the refresh or back buttons in the browser can result in unexpected behavior. |  |