Client-Side Validation for ASP.NET Server Controls

Visual Studio 2010

If the user is working with a browser that supports dynamic HTML (DHTML), ASP.NET validation controls can perform validation using client script. Because the controls can provide immediate feedback without a round trip to the server, the user experience with the page is enhanced.

Under most circumstances, you do not have to make any changes to your page or to the validation controls to use client-side validation. The controls automatically detect if the browser supports DHTML and perform their checking accordingly. Client-side validation uses the same error display mechanism as server-side validation.

🔒 **Security Note:**
Validation is performed on the server even if it was already performed on the client. This enables you to determine validation status in server code and provides security against users bypassing client-side validation.

Differences in Client-Side Validation

If validation is performed on the client, validation controls include some additional features:

- If you are summarizing validation error messages, you can display them in a message box that appears when the user submits the page. For details, see How to: Control Validation Error Message Display for ASP.NET Server Controls.
- The object model for validation controls is slightly different on the client. See Client Validation Object Model later in this topic.

There are a few minor differences associated with client-side validation:

- If client-side validation is enabled, the page includes references to script libraries that are used to perform the client-side validation.
- When you use a RegularExpressionValidator control, the expressions can be checked on the client if an ECMAScript-compatible language (such as Microsoft JScript) is available. Client-side regular expressions differ in small details from the regular expression checking done on the server using the Regex class.
- The page includes a client-side method to intercept and handle the **Click** event before the page is submitted.

Client Validation Object Model

Validation controls present almost the same object model on the client as on the server. For example, you can test validation by reading a validation control's **IsValid** property the same way on both the client and the server.

However, there are differences in the validation information exposed at the page level. On the server, the page supports properties; on the client, it contains global variables. The following table compares the information exposed on the page.

Client Page Variable	Server Page Property
Page_IsValid	**IsValid**
Page_Validators (array) Contains references to all validation controls on the page.	**Validators** (collection) Contains references to all validation controls.
Page_ValidationActive A Boolean value that indicates whether validation should take place. Set this variable to **false** to turn off client-side validation programmatically.	(no equivalent)

Note:
All page-related validation information should be considered read-only.

Posting Pages with Client-Side Validation Errors

In some instances, you might prefer not to use client-side validation and to use only server-side validation, even if client-side validation is available. For example, client-side validation might not be possible if validation requires information or resources that are available only on the server, such as access to a database.

By default, when client-side validation is being performed, the user cannot post the page to the server if there are errors on the page. However, you might find it necessary to enable the user to post even with errors. For example, you might have a cancel button or a navigation button on a page, and you want the button to submit the page even if some controls would fail validation. For more information, see How to: Disable Validation for ASP.NET Server Controls.

See Also

Concepts

Types of Validation for ASP.NET Server Controls

Other Resources

Validation ASP.NET Controls

Community Additions