How to: Validate Programmatically for ASP.NET Server Controls

Visual Studio 2010

By default, ASP.NET validation controls perform validation automatically when the page is posted back to the server, after page initialization (that is, after view-state and postback data have been processed) and before your event-handling code is called. The controls might also perform validation in the browser if the browser supports client script.

However, there are times when you might want to perform validation yourself programmatically. You might validate programmatically under these circumstances:

- If validation values are not set until run time. For example, if you are working with a RangeValidator control, you might be setting its MinimumValue and MaximumValue properties at run time based on values entered by the user. In that case, default validation does not work, because at the time that the page calls the validation controls to perform validation, there is insufficient information in the RangeValidator control.
- If you want to determine the validity of a control (or of the page as a whole) in the **Page_Load** event handler. At that stage of page processing, the validation controls have not yet been invoked, so the **IsValid** property of the page or of individual controls is not set. (An exception is thrown if you try to get the value of this property.) However, if you want to determine validity, you can programmatically call validation.
- If you are adding controls (either input controls or validation controls) at run time.

More generally, you can validate programmatically any time when you want to have precise control over when validation is carried out.

Security Note:
By default, the ASP.NET Web pages automatically validate that malicious users are not attempting to send script or HTML elements to your application. If you have disabled this feature, you can call the ValidateInput method yourself. For more information, see Script Exploits Overview.

To validate programmatically

- Call the validation control's Validate method.

  The control will perform its check and will set the **IsValid** property of the control and the page. If an error has been detected, when the page is returned to the user, error messages will be displayed as usual.

  The following code example shows how you can set properties programmatically. In this case, an ASP.NET Web page takes reservations at a resort that features a free tour during every visit. Users must enter an arrival date and a departure date, and then schedule the tour during their visit. A RangeValidator control is used to ensure that the user entered a typical date format, and that the tour date falls between the arrival and departure dates.

Note:
If a user enters a value that cannot be converted to a date, the validator control will throw an exception. For clarity,

this example does not include error handling.

The arrival and departure dates come from two TextBox Web server controls on the page, `txtArrival` and `txtDeparture`. The tour date is entered in a third TextBox control, `txtTourDate`, which is validated by the RangeValidator control.

Note:

When validating programmatically, you should disable client script so that the control will not show the wrong error message before your server-side validation code executes. For details, see How to: Disable Validation for ASP.NET Server Controls.

C#

```
<%@ Page Language="C#"  %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

<head runat="server">
    <title>How to: Validate Programmatically for ASP.NET Server Controls</title>
</head>

<script runat="server">

    private void Button1_Click(object sender, System.EventArgs e)
    {
        RangeValidator1.MinimumValue = txtArrival.Text;
        RangeValidator1.MaximumValue = txtDeparture.Text;
        RangeValidator1.Type = ValidationDataType.Date;
        RangeValidator1.Validate();

        if (!RangeValidator1.IsValid)
        {
            RangeValidator1.ErrorMessage = "The tour date must " +
            "fall between the arrival and departure dates.";
        }
    }

</script>

<body>
    <form id="form1" runat="server">
    <div>
      Arrival:
      <asp:TextBox id="txtArrival" runat="server"></asp:TextBox><br />

      Departure:
      <asp:TextBox id="txtDeparture" runat="server"></asp:TextBox><br />

      Free Tour Date:
      <asp:TextBox id="txtFreeTour" runat="server"></asp:TextBox><br />
      <asp:RangeValidator EnableClientScript="false"
                          id="RangeValidator1"
                          runat="server"
```

```
                            ControlToValidate="txtFreeTour" >
        </asp:RangeValidator>

        <asp:Button Text="Book Trip" id="Button1" OnClick="Button1_Click" runat="server"
/>
    </div>
    </form>
</body>
</html>
```

See Also

Concepts

Types of Validation for ASP.NET Server Controls

Other Resources

Validation ASP.NET Controls

Community Additions