Validator Web Controls

Contents

[1 Validator Web Controls 1](#_Toc127022168)

[2 CustomValidator 2](#_Toc127022169)

[3 OnClientClick 2](#_Toc127022170)

[Appendix 1 n/a 3](#_Toc127022171)

# Validator Web Controls

Validator web controls are used to validate data (usually on the client side) before posting back to the server. If data is found to be invalid, an error message will be displayed beside the control where the error is, in a validation summary of all errors, or both. The first five validation controls show below all share these properties:

1. ControlToValidate – Specifies the ID of the control to be validated.
2. ErrorMessage – Gets or sets the text for the error message displayed in a ValidationSummary control when validation fails.
3. Text – Gets or sets the text displayed in the validation control when validation fails.

Note: the validator controls will not work unless you add this node to *web.config*:

<configuration>

// Add the node below inside the *configuration* node.

 <appSettings>

 <add key="ValidationSettings:UnobtrusiveValidationMode" value="None" />

 </appSettings>

1. **Control**: [CompareValidator](http://msdn.microsoft.com/en-us/library/vstudio/system.web.ui.webcontrols.comparevalidator.aspx)

**Description**: Compares the value of one input control to the value of another input control or to a fixed value.

**Properties**: ControlToCompare, Operator, Type

1. **Control**: [CustomValidator](http://msdn.microsoft.com/en-us/library/vstudio/system.web.ui.webcontrols.customvalidator.aspx)

**Description**: Does a client-side user-defined validation. Can also do server-side.

**Properties**: ClientValidationFunction

1. **Control**: [RangeValidator](http://msdn.microsoft.com/en-us/library/vstudio/system.web.ui.webcontrols.rangevalidator.aspx)

**Description**: Checks whether the value of an input control is within a specified range of values.

**Properties**: MinimumValue, MaximumValue, Type

1. **Control**: [RequiredFieldValidator](http://msdn.microsoft.com/en-us/library/vstudio/system.web.ui.webcontrols.requiredfieldvalidator.aspx)

**Description**: Checks to make sure a value was entered

1. **Control**: [RegularExpressionValidator](http://msdn.microsoft.com/en-us/library/vstudio/system.web.ui.webcontrols.regularexpressionvalidator.aspx)

**Description**: Checks to see if input matches the pattern specified by a regular expression

**Properties**: ValidationExpression

1. **Control**: [ValidationSummary](http://msdn.microsoft.com/en-us/library/vstudio/system.web.ui.webcontrols.validationsummary.aspx)

**Description**: Displays a summary of all validation errors inline on a Web page, in a message box, or both.

**Properties**: HeaderText

# CustomValidator

1. You can use the *CustomValidator* control to supply custom validation on the client side. Follow these steps:
2. Drag a CustomValidator (Data Tab in ToolBox) onto the page.
3. Set these properties: ErrorMessage and Text
4. Set the *ControlToValidate* property. Note, you may want to validate every control. For instance, suppose you want to add something to one ListBox provided it is not already in that or another ListBox. As you will see, it really doesn’t matter which ListBox you set to the *ControlToValidate*, just set it to one of them.
5. Set the *ClientValidationFunction* to the name of the client-side JavaScript function you will write. For the sake of this example, we will call this function: *MyValidator.*
6. Go to the head section of the source and write a function like this:

 **<script type="text/javascript" language="javascript">**

 **// Must have these two parameters: source and arguments**

 **function MyValidator(source, arguments)**

 **{**

 **// Must set IsValid property to either true or false.**

**// “True” allows post-back, “False” stops it and displays**

**// the error message defined in CustomValidator.**

 **arguments.IsValid = true;**

**// or…**

 **arguments.IsValid = false;**

 **}**

 **</script>**

# OnClientClick

1. You can set the OnClientClick property on some controls, like the Button, so that it fires client-side JavaScript code that can stop a postback. This is similar to the CustomValidator above, but you don’t get the built-in display of the error message. You would have to program your own, for instance with an Alert statement or setting the innerHTML of some tag.
2. Suppose we want to do custom validation on the client side when a button is pressed. Follow these steps:
3. Select the Button and set its *OnClientClick* property to this: “return MyValidator();” Of course, *MyValidator* can be any name you choose.
4. Verify that the *OnClientClick* property is set in the source:

**<asp:Button ID="btnSelName" runat="server" onclick="btnSelName\_Click"**

**Text="Select" onclientclick="return MyValidator();" />**

1. Go to the head section of the source and write a function like this:

 **<script type="text/javascript" language="javascript">**

 **function MyValidator() {**

 **// Must return true or false.**

**// “True” allows post-back, “False” stops it**

 **return true;**

**// or…**

**return false;**

 **}**

 **</script>**

1. Hint: put “alert(‘hello world’);” in the first line of a JavaScript function to make sure it is being called.

Appendix

1. n/a