**ASP.NET server controls overview**

This article refers to the following Microsoft .NET Framework Class Library namespaces: 

* System.Web.UI.HtmlControls.HtmlControl
* System.Web.UI.WebControls.WebControl

**Summary**

This article provides an introduction to the ASP.NET server controls.   
  
For additional ASP.NET overviews, refer to the following Microsoft Knowledge Base article: 

[305140](https://support.microsoft.com/EN-US/help/305140)INFO: ASP.NET Roadmap

**More Information**

The ASP.NET page framework includes a number of built-in server controls that are designed to provide a more structured programming model for the Web. These controls provide the following features: 

* Automatic state management.
* Simple access to object values without having to use the Request object.
* Ability to react to events in server-side code to create applications that are better structured.
* Common approach to building user interfaces for Web pages.
* Output is automatically customized based on the capabilities of the browser.

In addition to the built-in controls, the ASP.NET page framework also provides the ability to create user controls and custom controls. User controls and custom controls can enhance and extend existing controls to build a much richer user interface. 

**HTML Server Controls**

The HTML server controls are Hypertext Markup Language (HTML) elements that include a runat=server attribute. The HTML server controls have the same HTML output and the same properties as their corresponding HTML tags. In addition, HTML server controls provide automatic state management and server-side events. HTML server controls offer the following advantages: 

* The HTML server controls map one to one with their corresponding HTML tags.
* When the ASP.NET application is compiled, the HTML server controls with the runat=server attribute are compiled into the assembly.
* Most controls include an OnServerEvent for the most commonly used event for the control. For example, the <input type=button> control has an OnServerClick event.
* The HTML tags that are not implemented as specific HTML server controls can still be used on the server side; however, they are added to the assembly as HtmlGenericControl.
* When the ASP.NET page is reposted, the HTML server controls keep their values.

The System.Web.UI.HtmlControls.HtmlControl base class contains all of the common properties. HTML server controls derive from this class.   
  
To use an HTML server control, use the following syntax (which uses the HtmlInputText control as an example):

<input type="text" value="hello world" runat=server />

For more information about individual HTML server controls that are available in ASP.NET, refer to the following Microsoft Web sites:

[HtmlAnchor Control](http://msdn.microsoft.com/en-us/library/8ff86hxd(v=vs.71).aspx)  
  
[HtmlButton Control](http://msdn.microsoft.com/en-us/library/a8fd2268(v=vs.71).aspx)  
  
[HtmlForm Control](http://msdn.microsoft.com/en-us/library/dd120y50(v=vs.71).aspx)  
  
[HtmlImage Control](http://msdn.microsoft.com/en-us/library/8551b36z(v=vs.71).aspx)  
  
[HtmlInputButton Control](http://msdn.microsoft.com/en-us/library/s4dyt5wk(v=vs.71).aspx)  
  
[HtmlInputCheckBox Control](http://msdn.microsoft.com/en-us/library/31d4thc6(v=vs.71).aspx)  
  
[HtmlInputFile Control](http://msdn.microsoft.com/en-us/library/1s43z4wk(v=vs.71).aspx)  
  
[HtmlInputHidden Control](http://msdn.microsoft.com/en-us/library/k65s5xs3(v=vs.71).aspx)  
  
[HtmlInputImage Control](http://msdn.microsoft.com/en-us/library/44z2k814(v=vs.71).aspx)  
  
[HtmlInputRadioButton Control](http://msdn.microsoft.com/en-us/library/17tk0thz(v=vs.71).aspx)  
  
[HtmlInputText Control](http://msdn.microsoft.com/en-us/library/f8kdafb5(v=vs.71).aspx)  
  
[HtmlSelect Control](http://msdn.microsoft.com/en-us/library/807bc327(v=vs.71).aspx)  
  
[HtmlTable Control](http://msdn.microsoft.com/en-us/library/2962t2k8(v=vs.71).aspx)  
  
[HtmlTableCell Control](http://msdn.microsoft.com/en-us/library/5wsbhse3(v=vs.71).aspx)  
  
[HtmlTableCell Control](http://msdn.microsoft.com/en-us/library/405596yw(v=vs.71).aspx)  
  
[HtmlTextArea Control](http://msdn.microsoft.com/en-us/library/h8ff3dty(v=vs.71).aspx)

**Web Server Controls**

Web controls are very similar to the HTML server controls such as  
Button, TextBox, and Hyperlink, except that Web controls have a standardized set of property names. Web server controls offer the following advantages: 

* Make it easier for manufacturers and developers to build tools or applications that automatically generate the user interface.
* Simplify the process of creating interactive Web forms, which requires less knowledge of how HTML controls work and make the task of using them less prone to errors.

The System.Web.UI.WebControls.WebControl base class contains all of the common properties. Most of the Web server controls derive from this class.  
  
To use a Web server control, use the following syntax (which uses the TextBox control as an example):

<asp:textbox text="hello world" runat=server />

Web server controls can be divided into four categories: 

* [Basic Web Controls](https://support.microsoft.com/en-in/help/306459/asp-net-server-controls-overview#2a)
* [Validation Controls](https://support.microsoft.com/en-in/help/306459/asp-net-server-controls-overview#2b)
* [List Controls](https://support.microsoft.com/en-in/help/306459/asp-net-server-controls-overview#2c)
* [Rich Controls](https://support.microsoft.com/en-in/help/306459/asp-net-server-controls-overview#2d)

**Basic Web Controls**

Basic Web controls provide the same functionality as their HTML server control counterparts. However, basic Web control include additional methods, events, and properties against which you can program.  
  
For more information about individual Web controls that are available in ASP.NET, refer to the following Microsoft Web sites:

[Button Web Server Control](http://msdn.microsoft.com/en-us/library/dx5ybk93(v=vs.71).aspx)  
  
[CheckBox Web Server Control](http://msdn.microsoft.com/en-us/library/4s78d0k1(v=vs.71).aspx)  
  
[HyperLink Web Server Control](http://msdn.microsoft.com/en-us/library/k0b15efk(v=vs.71).aspx)  
  
[Image Web Server Control](http://msdn.microsoft.com/en-us/library/c6te4s54(v=vs.71).aspx)  
  
[ImageButton Web Server Control](http://msdn.microsoft.com/en-us/library/1z8fsbyh(v=vs.71).aspx)  
  
[Label Web Server Control](http://msdn.microsoft.com/en-us/library/2wwfb06z(v=vs.71).aspx)  
  
[LinkButton Web Server Control](http://msdn.microsoft.com/en-us/library/1cd4z1zs(v=vs.71).aspx)  
  
[Literal Web Server Control](http://msdn.microsoft.com/en-us/library/cc088zwa(v=vs.71).aspx)  
  
[Panel Web Server Control](http://msdn.microsoft.com/en-us/library/152e73cy(v=vs.71).aspx)  
  
[PlaceHolder Web Server Control](http://msdn.microsoft.com/en-us/library/as54k8b6(v=vs.71).aspx)  
  
[RadioButton Web Server Control](http://msdn.microsoft.com/en-us/library/xke2zw4x(v=vs.71).aspx)  
  
[Table Web Server Control](http://msdn.microsoft.com/en-us/library/9f65szta(v=vs.71).aspx)  
  
[TableCell Web Server Control](http://msdn.microsoft.com/en-us/library/x595ddwc(v=vs.71).aspx)  
  
[TableRow Web Server Control](http://msdn.microsoft.com/en-us/library/tk1zfd2e(v=vs.71).aspx)  
  
[TextBox Web Server Control](http://msdn.microsoft.com/en-us/library/fhc2c904(v=vs.71).aspx)

**Validation Controls**

Validation controls are used to validate the values that are entered into other controls of the page. Validation controls perform client-side validation, server-side validation, or both, depending on the capabilities of the browser in which the page is displayed. Validation controls offer the following advantages: 

* You can associate one or more validation controls with each control that you want to validate.
* The validation is performed when the page form is submitted.
* You can specify programmatically whether validation should occur, which is useful if you want to provide a cancel button so that the user can exit without having to fill valid data in all of the fields.
* The validation controls automatically detect whether validation should be performed on the client side or the server side.  
    
  Note A client-side validation catches errors before a postback operation is complete. Therefore, if you have combinations of client-side and server-side validation controls on a single page, the server-side validation will be preempted if a client-side validation fails.

For more information about individual validation controls that are available in ASP.NET, refer to the following Microsoft Web sites:

[RequiredFieldValidator Control](http://msdn.microsoft.com/en-us/library/cc295489.aspx)  
  
[RangeValidator Control](http://msdn.microsoft.com/en-us/library/cc295453.aspx)  
  
[CompareValidator Control](http://msdn.microsoft.com/en-us/library/cc295090.aspx)  
  
[RegularExpressionValidator Control](http://msdn.microsoft.com/en-us/library/cc295107.aspx)  
  
[CustomValidator Control](http://msdn.microsoft.com/en-us/library/cc295446.aspx)  
  
[ValidationSummary Control](http://msdn.microsoft.com/en-us/library/cc295561.aspx)

**List Controls**

List controls are special Web server controls that support binding to collections. You can use list controls to display rows of data in a customized, templated format. All list controls expose DataSource and DataMember properties, which are used to bind to collections.  
  
List controls can bind only to collections that support the IEnumerable, ICollection, or IListSource interfaces. For example, a Microsoft Visual C# .NET sample page appears as follows:

<%@ Page Language="C#" %>  
<script runat="server">  
Public void Page\_Load()  
{  
String[] myStringArray = new String[] {"one","two","three"};  
rptr.DataSource = myStringArray;  
rptr.DataBind();  
}  
</script>  
<html>  
<body>  
<asp:repeater id=rptr runat="server">  
<itemtemplate><%# Container.DataItem %><br></itemtemplate>  
</asp:repeater>  
</body>  
</html>

A Microsoft Visual Basic .NET sample page appears as follows:

<%@ Page Language="vb" %>  
<script runat="server">  
public sub Page\_Load()  
 Dim myStringArray as String()  
 myStringArray = new String() {"one","two","three"}  
 rptr.DataSource = myStringArray  
 rptr.DataBind()  
end sub  
</script>  
<html>  
<body>  
<asp:repeater id=rptr runat="server">  
 <itemtemplate><%# Container.DataItem %><br></itemtemplate>  
</asp:repeater>  
</body>  
</html>

The output appears as follows:

one  
  
two  
  
three

For more information about individual list controls that are available in ASP.NET, refer to the following Microsoft Web sites:

[ListBox Web Server Control](http://msdn.microsoft.com/en-us/library/z4d7ktzs(v=vs.71).aspx)  
  
[CheckBoxList Web Server Control](http://msdn.microsoft.com/en-us/library/8bw4x4wa(v=vs.71).aspx)  
  
[RadioButtonList Web Server Control](http://msdn.microsoft.com/en-us/library/y7k30eyz(v=vs.71).aspx)  
  
[Repeater Web Server Control](http://msdn.microsoft.com/en-us/library/c012haty(v=vs.71).aspx)  
  
[DataList Web Server Control](http://msdn.microsoft.com/en-us/library/yfx4t9t7(v=vs.71).aspx)  
  
[DataGrid Web Server Control](http://msdn.microsoft.com/en-us/library/64xx84kc(v=vs.71).aspx)  
  
[DropDownList Web Server Control](http://msdn.microsoft.com/en-us/library/0dzka5sf(v=vs.71).aspx)

**Rich Controls**

In addition to the preceding controls, the ASP.NET page framework provides a few, task-specific controls called rich controls. Rich controls are built with multiple HTML elements and contain rich functionality. Examples of rich controls are the Calendar control and the AdRotator control.  
  
For more information about individual rich controls that are available in ASP.NET, refer to the following Microsoft Web sites:

[AdRotator Web Server Control](http://msdn.microsoft.com/en-us/library/s5z9ks4y(v=vs.71).aspx)  
  
[Calendar Web Server Control](http://msdn.microsoft.com/en-us/library/dxf9k8sh(v=vs.71).aspx)  
  
[Xml Web Server Control](http://msdn.microsoft.com/en-us/library/day7x985(v=vs.71).aspx)

**User Controls**

Often, you may want to reuse the user interface of your Web Form without having to write any extra code. ASP.NET enables you to do this by converting your Web Forms into user controls. User controls, which have the .ascx file extension, can be used multiple times within a single Web Form.  
  
To convert a Web Form into a user control, follow these steps: 

1. Remove all <html>,<head>, <body> and <form> tags.
2. If the @ Page directive appears in the page, change it to @ Control.
3. Include a className attribute in the @ Control directive so that the user control is typed strongly when you instantiate it.
4. Give the control a descriptive file name, and change the file extension from .aspx to .ascx.

For more information about user controls, refer to the following topic in the Microsoft .NET Framework Software Development Kit (SDK) documentation:

[Web Forms User Controls](http://msdn.microsoft.com/en-us/library/y6wb1a0e(v=vs.71).aspx)

**Custom Controls**

In addition to the built-in Web controls, ASP.NET also allows you to create your own custom controls. It may be useful to develop custom controls if you are faced with one of these scenarios: 

* You need to combine the functionality of two or more built-in Web controls.
* You need to extend the functionality of a built-in control.
* You need a control that is completely different than any of the controls that currently exist.

For more information about developing custom controls, refer to the following topics in the .NET Framework SDK documentation:

[Developing ASP.NET Server Controls](http://msdn.microsoft.com/en-us/library/aa719973(vs.71).aspx)  
  
[Developing Custom Controls: Key Concepts](http://msdn.microsoft.com/en-us/library/aa720226(v=vs.71).aspx)  
  
[Developing a Composite Control](http://msdn.microsoft.com/en-us/library/aa719968(v=vs.71).aspx)