

ASP.NET Introduction

What You Should Already Know

Before you continue you should have a basic understanding of the following:

- WWW, HTML, XML and the basics of building Web pages
- Scripting languages like JavaScript or VBScript
- The basics of server side scripting like ASP or PHP

If you want to study these subjects first, find the tutorials on our [Home Page](#)

What is Classic ASP?

Microsoft's previous server side scripting technology ASP (Active Server Pages) is now often called classic ASP.

ASP 3.0 was the last version of classic ASP.

To learn more about classic ASP, you can study our [ASP tutorial](#).

ASP.NET is NOT ASP

ASP.NET is the next generation ASP, but it's not an upgraded version of ASP.

ASP.NET is an entirely new technology for server-side scripting. It was written from the ground up and is not backward compatible with classic ASP.

You can read more about the differences between ASP and ASP.NET in the next chapter of this tutorial.

ASP.NET is the major part of the Microsoft's .NET Framework.

What is ASP.NET?

ASP.NET is a server side scripting technology that enables scripts (embedded in web pages) to be executed by an Internet server.

- ASP.NET is a Microsoft Technology
- ASP stands for Active Server Pages
- ASP.NET is a program that runs inside IIS
- IIS (Internet Information Services) is Microsoft's Internet server
- IIS comes as a free component with Windows servers
- IIS is also a part of Windows 2000 and XP Professional

What is an ASP.NET File?

- An ASP.NET file is just the same as an HTML file
- An ASP.NET file can contain HTML, XML, and scripts
- Scripts in an ASP.NET file are executed on the server
- An ASP.NET file has the file extension ".aspx"

How Does ASP.NET Work?

- When a browser requests an HTML file, the server returns the file
 - When a browser requests an ASP.NET file, IIS passes the request to the ASP.NET engine on the server
 - The ASP.NET engine reads the file, line by line, and executes the scripts in the file
 - Finally, the ASP.NET file is returned to the browser as plain HTML
-

What is ASP+?

ASP+ is the same as ASP.NET.

ASP+ is just an early name used by Microsoft when they developed ASP.NET.

The Microsoft .NET Framework

The .NET Framework is the infrastructure for the Microsoft .NET platform.

The .NET Framework is an environment for building, deploying, and running Web applications and Web Services.

Microsoft's first server technology ASP (Active Server Pages), was a powerful and flexible "programming language". But it was too code oriented. It was not an application framework and not an enterprise development tool.

The Microsoft .NET Framework was developed to solve this problem.

<http://www.w3schools.com/aspnet/default.asp>

.NET Frameworks keywords:

- Easier and quicker programming
- Reduced amount of code
- Declarative programming model
- Richer server control hierarchy with events
- Larger class library
- Better support for development tools

The .NET Framework consists of 3 main parts:

Programming languages:

- C# (Pronounced C sharp)
- Visual Basic (VB .NET)
- J# (Pronounced J sharp)

Server technologies and client technologies:

- ASP .NET (Active Server Pages)
- Windows Forms (Windows desktop solutions)
- Compact Framework (PDA / Mobile solutions)

Development environments:

- Visual Studio .NET (VS .NET)
- Visual Web Developer

This tutorial is about ASP.NET.

ASP.NET 2.0

ASP.NET 2.0 improves upon ASP.NET by adding support for several new features.

You can read more about the differences between ASP.NET 2.0 and ASP.NET in the next chapter of this tutorial.

ASP.NET 3.0

ASP.NET 3.0 is not a new version of ASP.NET. It's just the name for a new ASP.NET 2.0 framework library with support for Windows Presentation Foundation, Windows Communication Foundation, Windows Workflow Foundation; and Windows CardSpace.

ASP.NET 3.0 is not covered in this tutorial.

ASP.NET vs. ASP

ASP.NET has better language support, a large set of new controls, XML-based components, and better user authentication.

ASP.NET provides increased performance by running compiled code.

ASP.NET code is not fully backward compatible with ASP.

New in ASP.NET

- Better language support
 - Programmable controls
 - Event-driven programming
 - XML-based components
 - User authentication, with accounts and roles
 - Higher scalability
 - Increased performance - Compiled code
 - Easier configuration and deployment
 - Not fully ASP compatible
-

Language Support

ASP.NET uses ADO.NET.

ASP.NET supports full Visual Basic, not VBScript.

ASP.NET supports C# (C sharp) and C++.

ASP.NET supports JScript.

ASP.NET Controls

ASP.NET contains a large set of HTML controls. Almost all HTML elements on a page can be defined as ASP.NET control objects that can be controlled by scripts.

ASP.NET also contains a new set of object-oriented input controls, like programmable list-boxes and validation controls.

A new data grid control supports sorting, data paging, and everything you can expect from a dataset control.

Event Aware Controls

All ASP.NET objects on a Web page can expose events that can be processed by ASP.NET code.

Load, Click and Change events handled by code makes coding much simpler and much better organized.

ASP.NET Components

ASP.NET components are heavily based on XML. Like the new AD Rotator, that uses XML to store advertisement information and configuration.

User Authentication

ASP.NET supports form-based user authentication, cookie management, and automatic redirecting of unauthorized logins.

User Accounts and Roles

ASP.NET allows user accounts and roles, to give each user (with a given role) access to different server code and executables.

High Scalability

Much has been done with ASP.NET to provide greater scalability.

Server-to-server communication has been greatly enhanced, making it possible to scale an application over several servers. One example of this is the ability to run XML parsers, XSL transformations and even resource hungry session objects on other servers.

Compiled Code

The first request for an ASP.NET page on the server will compile the ASP.NET code and keep a cached copy in memory. The result of this is greatly increased performance.

Easy Configuration

Configuration of ASP.NET is done with plain text files.

Configuration files can be uploaded or changed while the application is running. No need to restart the server. No more metabase or registry puzzle.

Easy Deployment

No more server-restart to deploy or replace compiled code. ASP.NET simply redirects all new requests to the new code.

Compatibility

ASP.NET is not fully compatible with earlier versions of ASP, so most of the old ASP code will need some changes to run under ASP.NET.

To overcome this problem, ASP.NET uses a new file extension ".aspx". This will make ASP.NET applications able to run side by side with standard ASP applications on the same server.

ASP.NET Installing

ASP.NET is easy to install. Just follow the instructions below.

<http://www.w3schools.com/aspnet/default.asp>

What You Need

A Windows Computer

ASP.NET is a Microsoft technology. To run ASP.NET you need a computer capable of running Windows.

Windows 2000 or XP

If you are serious about developing ASP.NET applications you should install Windows 2000 Professional or Windows XP Professional.

In both cases, make sure you install the Internet Information Services (IIS) from the Add/Remove Windows components dialog.

Service Packs and Updates

Before ASP.NET can be installed on your computer, it is necessary to have all relevant service packs and security updates installed.

The easiest way to do this is to activate your Windows Internet Update. When you access the Windows Update page, you will be instructed to install the latest service packs and all critical security updates. For Windows 2000, make sure you install Service Pack 2. I will also recommend that you install Internet Explorer 6.

Read the note about connection speed and download time at the bottom of this page.

Remove Your Beta Version

If you have a Beta version of ASP.NET installed, we recommend that you completely uninstall it. Or even better: start with a fresh Windows 2000 or XP installation.

Install .NET

From your Windows Update you can now select to install the Microsoft .NET Framework.

After download, the .NET framework will install itself on your computer - there are no options to select for installation.

You should now be ready to develop your first ASP.NET application!

The .NET Software Development Kit

If you have the necessary bandwidth to download over 130 MB, you might consider downloading the full Microsoft .NET Software Development Kit (SDK).

We fully recommend getting the SDK for learning more about .NET and for the documentation, samples, and tools included.

Connection Speed and Download Time

If you have a slow Internet connection, you might have problems downloading large files like the service packs, the SDK and the latest version of Internet Explorer.

If download speed is a problem, our best suggestion is to get the latest files from someone else, from a colleague, from a friend, or from one of the CDs that comes with many popular computer magazines. Look for Windows 2000 Service Pack 2, Internet Explorer 6, and the Microsoft .NET Framework.

ASP.NET - Web Pages

A simple ASP.NET page looks just like an ordinary HTML page.

Hello W3Schools

To start learning ASP.NET, we will construct a very simple HTML page that will display "Hello W3Schools" in an Internet browser like this:



Hello W3Schools!

Hello W3Schools in HTML

This code displays the example as an HTML page:

```
<html>
<body bgcolor="yellow">
<center>
<h2>Hello W3Schools!</h2>
</center>
</body>
</html>
```

If you want to try it yourself, save the code in a file called "**firstpage.htm**", and create a link to the file like this: [firstpage.htm](#)

Hello W3Schools in ASP.NET

The simplest way to convert an HTML page into an ASP.NET page is to copy the HTML file to a new file with an **.aspx** extension.

This code displays our example as an ASP.NET page:

```
<html>
<body bgcolor="yellow">
<center>
<h2>Hello W3Schools!</h2>
</center>
</body>
</html>
```

If you want to try it yourself, save the code in a file called "**firstpage.aspx**", and create a link to the file like this: [firstpage.aspx](#)

How Does it Work?

Fundamentally an ASP.NET page is just the same as an HTML page.

<http://www.w3schools.com/aspnet/default.asp>

An HTML page has the extension .htm. If a browser requests an HTML page from the server, the server sends the page to the browser without any modifications.

An ASP.NET page has the extension .aspx. If a browser requests an ASP.NET page, the server processes any executable code in the page, before the result is sent back to the browser.

The ASP.NET page above does not contain any executable code, so nothing is executed. In the next examples we will add some executable code to the page to demonstrate the difference between static HTML pages and dynamic ASP pages.

Classic ASP

Active Server Pages (ASP) has been around for several years. With ASP, executable code can be placed inside HTML pages.

Previous versions of ASP (before ASP .NET) are often called Classic ASP.

ASP.NET is not fully compatible with Classic ASP, but most Classic ASP pages will work fine as ASP.NET pages, with only minor changes.

If you want to learn more about Classic ASP, please visit our [ASP Tutorial](#).

Dynamic Page in Classic ASP

To demonstrate how ASP can display pages with dynamic content, we have added some executable code (in red) to the previous example:

```
<html>
<body bgcolor="yellow">
<center>
<h2>Hello W3Schools!</h2>
<p><%Response.Write(now())%></p>
</center>
</body>
</html>
```

The code inside the <% --%> tags is executed on the server.

Response.Write is ASP code for writing something to the HTML output stream.

Now() is a function returning the servers current date and time.

If you want to try it yourself, save the code in a file called "**dynpage.asp**", and create a link to the file like this: [dynpage.asp](#)

Dynamic Page in ASP .NET

This following code displays our example as an ASP.NET page:

```
<html>
<body bgcolor="yellow">
<center>
<h2>Hello W3Schools!</h2>
<p><%Response.Write(now())%></p>
</center>
</body>
</html>
```

If you want to try it yourself, save the code in a file called "**dynpage.aspx**", and create a link to the file like this: [dynpage.aspx](#)

ASP.NET vs Classic ASP

The previous examples didn't demonstrate any differences between ASP.NET and Classic ASP.

As you can see from the two latest examples there are no differences between the two ASP and ASP.NET pages.

In the next chapters you will see how server controls make ASP.NET more powerful than Classic ASP.

ASP.NET - Server Controls

Server controls are tags that are understood by the server.

Limitations in Classic ASP

The listing below was copied from the previous chapter:

```
<html>
<body bgcolor="yellow">
<center>
<h2>Hello W3Schools!</h2>
<p><%Response.Write(now())%></p>
</center>
</body>
</html>
```

The code above illustrates a limitation in Classic ASP: The code block has to be placed where you want the output to appear.

With Classic ASP it is impossible to separate executable code from the HTML itself. This makes the page difficult to read, and difficult to maintain.

ASP.NET - Server Controls

ASP.NET has solved the "spaghetti-code" problem described above with server controls.

Server controls are tags that are understood by the server.

There are three kinds of server controls:

- HTML Server Controls - Traditional HTML tags
- Web Server Controls - New ASP.NET tags
- Validation Server Controls - For input validation

ASP.NET - HTML Server Controls

HTML server controls are HTML tags understood by the server.

HTML elements in ASP.NET files are, by default, treated as text. To make these elements programmable, add a `runat="server"` attribute to the HTML element. This attribute indicates that

the element should be treated as a server control. The id attribute is added to identify the server control. The id reference can be used to manipulate the server control at run time.

Note: All HTML server controls must be within a <form> tag with the runat="server" attribute. The runat="server" attribute indicates that the form should be processed on the server. It also indicates that the enclosed controls can be accessed by server scripts.

In the following example we declare an HtmlAnchor server control in an .aspx file. Then we manipulate the HRef attribute of the HtmlAnchor control in an event handler (an event handler is a subroutine that executes code for a given event). The Page_Load event is one of many events that ASP.NET understands:

```
<script runat="server">
Sub Page_Load
link1.HRef="http://www.w3schools.com"
End Sub
</script>

<html>
<body>

<form runat="server">
<a id="link1" runat="server">Visit W3Schools!</a>
</form>

</body>
</html>
```

The executable code itself has been moved outside the HTML.

ASP.NET - Web Server Controls

Web server controls are special ASP.NET tags understood by the server.

Like HTML server controls, Web server controls are also created on the server and they require a runat="server" attribute to work. However, Web server controls do not necessarily map to any existing HTML elements and they may represent more complex elements.

The syntax for creating a Web server control is:

<http://www.w3schools.com/aspnet/default.asp>

```
<asp:control_name id="some_id" runat="server" />
```

In the following example we declare a Button server control in an .aspx file. Then we create an event handler for the Click event which changes the text on the button:

```
<script runat="server">  
Sub submit(Source As Object, e As EventArgs)  
    button1.Text="You clicked me!"  
End Sub  
</script>  
  
<html>  
<body>  
  
    <form runat="server">  
        <asp:Button id="button1" Text="Click me!"  
            runat="server" OnClick="submit"/>  
    </form>  
  
</body>  
</html>
```

ASP.NET - Validation Server Controls

Validation server controls are used to validate user-input. If the user-input does not pass validation, it will display an error message to the user.

Each validation control performs a specific type of validation (like validating against a specific value or a range of values).

By default, page validation is performed when a Button, ImageButton, or LinkButton control is clicked. You can prevent validation when a button control is clicked by setting the CausesValidation property to false.

The syntax for creating a Validation server control is:

```
<asp:control_name id="some_id" runat="server" />
```

In the following example we declare one TextBox control, one Button control, and one RangeValidator control in an .aspx file. If validation fails, the text "The value must be from 1 to 100!" will be displayed in the RangeValidator control:

Example

```
<html>
<body>

<form runat="server">
<p>Enter a number from 1 to 100:
<asp:TextBox id="tbox1" runat="server" />
<br /><br />
<asp:Button Text="Submit" runat="server" />
</p>

<p>
<asp:RangeValidator
ControlToValidate="tbox1"
MinimumValue="1"
MaximumValue="100"
Type="Integer"
Text="The value must be from 1 to 100!"
runat="server" />
</p>
</form>

</body>
</html>
```

ASP.NET - Events

An Event Handler is a subroutine that executes code for a given event.

ASP.NET - Event Handlers

Look at the following code:

<http://www.w3schools.com/aspnet/default.asp>

```
<%  
lbl1.Text="The date and time is " & now()  
%>  
  
<html>  
<body>  
<form runat="server">  
<h3><asp:label id="lbl1" runat="server" /></h3>  
</form>  
</body>  
</html>
```

When will the code above be executed? The answer is: "You don't know..."

The Page_Load Event

The Page_Load event is one of many events that ASP.NET understands. The Page_Load event is triggered when a page loads, and ASP.NET will automatically call the subroutine Page_Load, and execute the code inside it:

Example

```
<script runat="server">  
Sub Page_Load  
lbl1.Text="The date and time is " & now()  
End Sub  
</script>  
  
<html>  
<body>  
<form runat="server">  
<h3><asp:label id="lbl1" runat="server" /></h3>  
</form>  
</body>  
</html>
```

Note: The Page_Load event contains no object references or event arguments!

<http://www.w3schools.com/aspnet/default.asp>

The Page.IsPostBack Property

The Page_Load subroutine runs EVERY time the page is loaded. If you want to execute the code in the Page_Load subroutine only the FIRST time the page is loaded, you can use the Page.IsPostBack property. If the Page.IsPostBack property is false, the page is loaded for the first time, if it is true, the page is posted back to the server (i.e. from a button click on a form):

Example

```
<script runat="server">
Sub Page_Load
if Not Page.IsPostBack then
    lbl1.Text="The date and time is " & now()
end if
End Sub

Sub submit(s As Object, e As EventArgs)
lbl2.Text="Hello World!"
End Sub
</script>

<html>
<body>
<form runat="server">
<h3><asp:label id="lbl1" runat="server" /></h3>
<h3><asp:label id="lbl2" runat="server" /></h3>
<asp:button text="Submit" onclick="submit" runat="server" />
</form>
</body>
</html>
```

The example above will write the "The date and time is...." message only the first time the page is loaded. When a user clicks on the Submit button, the submit subroutine will write "Hello World!" to the second label, but the date and time in the first label will not change.

ASP.NET Web Forms

All server controls must appear within a <form> tag, and the <form> tag must contain the runat="server" attribute.

ASP.NET Web Forms

All server controls must appear within a <form> tag, and the <form> tag must contain the runat="server" attribute. The runat="server" attribute indicates that the form should be processed on the server. It also indicates that the enclosed controls can be accessed by server scripts:

```
<form runat="server">
```

```
...HTML + server controls
```

```
</form>
```

Note: The form is always submitted to the page itself. If you specify an action attribute, it is ignored. If you omit the method attribute, it will be set to method="post" by default. Also, if you do not specify the name and id attributes, they are automatically assigned by ASP.NET.

Note: An .aspx page can only contain ONE <form runat="server"> control!

If you select view source in an .aspx page containing a form with no name, method, action, or id attribute specified, you will see that ASP.NET has added these attributes to the form. It looks something like this:

```
<form name="_ctl0" method="post" action="page.aspx" id="_ctl0">
```

```
...some code
```

```
</form>
```

Submitting a Form

A form is most often submitted by clicking on a button. The Button server control in ASP.NET has the following format:

```
<asp:Button id="id" text="label" OnClick="sub" runat="server" />
```

The id attribute defines a unique name for the button and the text attribute assigns a label to the button. The onClick event handler specifies a named subroutine to execute.

In the following example we declare a Button control in an .aspx file. A button click runs a subroutine which changes the text on the button:

ASP .NET Maintaining the ViewState

You may save a lot of coding by maintaining the ViewState of the objects in your Web Form.

Maintaining the ViewState

When a form is submitted in classic ASP, all form values are cleared. Suppose you have submitted a form with a lot of information and the server comes back with an error. You will have to go back to the form and correct the information. You click the back button, and what happens.....ALL form values are CLEARED, and you will have to start all over again! The site did not maintain your ViewState.

When a form is submitted in ASP .NET, the form reappears in the browser window together with all form values. How come? This is because ASP .NET maintains your ViewState. The ViewState indicates the status of the page when submitted to the server. The status is defined through a hidden field placed on each page with a <form runat="server"> control. The source could look something like this:

```
<form name="_ctl0" method="post" action="page.aspx" id="_ctl0">  
<input type="hidden" name="__VIEWSTATE"  
value="dDwtNTI0ODU5MDE1Ozs+ZBCF2ryjMpeVgUrY2eTj79HNI4Q=" />
```

.....some code

```
</form>
```

Maintaining the ViewState is the default setting for ASP.NET Web Forms. If you want to NOT maintain the ViewState, include the directive <%@ Page EnableViewState="false" %> at the top of an .aspx page or add the attribute EnableViewState="false" to any control.

Look at the following .aspx file. It demonstrates the "old" way to do it. When you click on the submit button, the form value will disappear:

Example

```
<html>
<body>

<form action="demo_classicasp.aspx" method="post">
Your name: <input type="text" name="fname" size="20">
<input type="submit" value="Submit">
</form>
<%
dim fname
fname=Request.Form("fname")
If fname<>"" Then
Response.Write("Hello " & fname & "!")
End If
%>

</body>
</html>
```

Here is the new ASP .NET way. When you click on the submit button, the form value will NOT disappear:

Example

Click view source in the right frame of the example to see that ASP .NET has added a hidden field in the form to maintain the ViewState

```
<script runat="server">
Sub submit(sender As Object, e As EventArgs)
lbl1.Text="Hello " & txt1.Text & "!"
End Sub
</script>

<html>
<body>

<form runat="server">
Your name: <asp:TextBox id="txt1" runat="server" />
<asp:Button OnClick="submit" Text="Submit" runat="server" />
<p><asp:Label id="lbl1" runat="server" /></p>
</form>

</body>
</html>
```

ASP .NET - The TextBox Control

```
<script runat="server">
Sub change(sender As Object, e As EventArgs)
lbl1.Text="You changed text to " & txt1.Text
End Sub
</script>

<html>
<body>

<form runat="server">
Enter your name:
<asp:TextBox id="txt1" runat="server"
```

```
text="Hello World!"
ontextchanged="change" autopostback="true"/>
<p><asp:Label id="lbl1" runat="server" /></p>
</form>

</body>
</html>
```

ASP.NET - The Button Control

The Button control is used to display a push button.

The Button Control

The Button control is used to display a push button. The push button may be a submit button or a command button. By default, this control is a submit button.

A submit button does not have a command name and it posts the page back to the server when it is clicked. It is possible to write an event handler to control the actions performed when the submit button is clicked.

A command button has a command name and allows you to create multiple Button controls on a page. It is possible to write an event handler to control the actions performed when the command button is clicked.

The Button control's attributes and properties are listed in our [web controls reference page](#).

The example below demonstrates a simple Button control:

```
<html>
<body>

<form runat="server">
<asp:Button id="b1" Text="Submit" runat="server" />
</form>

</body>
</html>
```

Add a Script

A form is most often submitted by clicking on a button.

In the following example we declare one TextBox control, one Button control, and one Label control in an .aspx file. When the submit button is triggered, the submit subroutine is executed. The submit subroutine writes a text to the Label control:

Example

```
<script runat="server">
Sub submit(sender As Object, e As EventArgs)
lbl1.Text="Your name is " & txt1.Text
End Sub
</script>

<html>
<body>

<form runat="server">
Enter your name:
<asp:TextBox id="txt1" runat="server" />
<asp:Button OnClick="submit" Text="Submit" runat="server" />
<p><asp:Label id="lbl1" runat="server" /></p>
</form>

</body>
</html>
```

ASP.NET - Data Binding

ASP.NET - Data Binding

We may use data binding to fill lists with selectable items from an imported data source, like a database, an XML file, or a script.

Data Binding

The following controls are list controls which support data binding:

- `asp:RadioButtonList`
- `asp:CheckBoxList`
- `asp:DropDownList`
- `asp:Listbox`

The selectable items in each of the above controls are usually defined by one or more `asp:ListItem` controls, like this:

```
<html>
<body>

<form runat="server">
<asp:RadioButtonList id="countrylist" runat="server">
<asp:ListItem value="N" text="Norway" />
<asp:ListItem value="S" text="Sweden" />
<asp:ListItem value="F" text="France" />
<asp:ListItem value="I" text="Italy" />
</asp:RadioButtonList>
</form>

</body>
</html>
```

However, with data binding we may use a separate source, like a database, an XML file, or a script to fill the list with selectable items.

By using an imported source, the data is separated from the HTML, and any changes to the items are made in the separate data source.

In the next three chapters, we will describe how to bind data from a scripted data source.

ASP.NET - The ArrayList Object

<http://www.w3schools.com/aspnet/default.asp>

The ArrayList object is a collection of items containing a single data value.

Create an ArrayList

The ArrayList object is a collection of items containing a single data value.

Items are added to the ArrayList with the Add() method.

The following code creates a new ArrayList object named mycountries and four items are added:

```
<script runat="server">  
Sub Page_Load  
if Not Page.IsPostBack then  
    dim mycountries=New ArrayList  
    mycountries.Add("Norway")  
    mycountries.Add("Sweden")  
    mycountries.Add("France")  
    mycountries.Add("Italy")  
end if  
end sub  
</script>
```

By default, an ArrayList object contains 16 entries. An ArrayList can be sized to its final size with the TrimToSize() method:

```
<script runat="server">  
Sub Page_Load  
if Not Page.IsPostBack then  
    dim mycountries=New ArrayList  
    mycountries.Add("Norway")  
    mycountries.Add("Sweden")  
    mycountries.Add("France")  
    mycountries.Add("Italy")  
    mycountries.TrimToSize()  
end if  
end sub  
</script>
```

An ArrayList can also be sorted alphabetically or numerically with the Sort() method:

<http://www.w3schools.com/aspnet/default.asp>

```
<script runat="server">  
Sub Page_Load  
if Not Page.IsPostBack then  
    dim mycountries=New ArrayList  
    mycountries.Add("Norway")  
    mycountries.Add("Sweden")  
    mycountries.Add("France")  
    mycountries.Add("Italy")  
    mycountries.TrimToSize()  
    mycountries.Sort()  
end if  
end sub  
</script>
```

To sort in reverse order, apply the Reverse() method after the Sort() method:

```
<script runat="server">  
Sub Page_Load  
if Not Page.IsPostBack then  
    dim mycountries=New ArrayList  
    mycountries.Add("Norway")  
    mycountries.Add("Sweden")  
    mycountries.Add("France")  
    mycountries.Add("Italy")  
    mycountries.TrimToSize()  
    mycountries.Sort()  
    mycountries.Reverse()  
end if  
end sub  
</script>
```

Data Binding to an ArrayList

An ArrayList object may automatically generate the text and values to the following controls:

- `asp:RadioButtonList`
- `asp:CheckBoxList`
- `asp:DropDownList`
- `asp:Listbox`

To bind data to a `RadioButtonList` control, first create a `RadioButtonList` control (without any `asp:ListItem` elements) in an `.aspx` page:

```
<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server" />
</form>

</body>
</html>
```

Then add the script that builds the list and binds the values in the list to the `RadioButtonList` control:

Example

```
<script runat="server">
Sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New ArrayList
    mycountries.Add("Norway")
    mycountries.Add("Sweden")
    mycountries.Add("France")
    mycountries.Add("Italy")
    mycountries.TrimToSize()
    mycountries.Sort()
    rb.DataSource=mycountries
    rb.DataBind()
end if
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server" />
</form>

</body>
</html>
```

The DataSource property of the RadioButtonList control is set to the ArrayList and it defines the data source of the RadioButtonList control. The DataBind() method of the RadioButtonList control binds the data source with the RadioButtonList control.

Note: The data values are used as both the Text and Value properties for the control. To add Values that are different from the Text, use either the Hashtable object or the SortedList object

Example – array list dropdown list

```
<script runat="server">
sub Page_Load
```

<http://www.w3schools.com/aspnet/default.asp>

```

if Not Page.IsPostBack then
    dim mycountries=New ArrayList
    mycountries.Add("Norway")
    mycountries.Add("Sweden")
    mycountries.Add("France")
    mycountries.Add("Italy")
    mycountries.TrimToSize()
    mycountries.Sort()
    dd.DataSource=mycountries
    dd.DataBind()
end if
end sub

sub displayMessage(s as Object,e As EventArgs)
    lbl1.text="Your favorite country is: " & dd.SelectedItem.Text
end sub
</script>

```

```

<html>
<body>

<form runat="server">
<asp:DropDownList id="dd" runat="server"
AutoPostBack="True" onSelectedIndexChanged="displayMessage" />
<p><asp:label id="lbl1" runat="server" /></p>
</form>

</body>
</html>

```

Example – Array list Radio Button

```

<script runat="server">
Sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New ArrayList
    mycountries.Add("Norway")
    mycountries.Add("Sweden")
    mycountries.Add("France")
    mycountries.Add("Italy")
    mycountries.TrimToSize()
    mycountries.Sort()
    rb.DataSource=mycountries

```

```
        rb.DataBind()
    end if
end sub

sub displayMessage(s as Object,e As EventArgs)
    lbl1.text="Your favorite country is: " & rb.SelectedItem.Text
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server"
AutoPostBack="True" onSelectedIndexChanged="displayMessage" />
<p><asp:label id="lbl1" runat="server" /></p>
</form>

</body>
</html>
```

ASP.NET - The Hashtable Object

The Hashtable object contains items in key/value pairs.

Create a Hashtable

The Hashtable object contains items in key/value pairs. The keys are used as indexes, and very quick searches can be made for values by searching through their keys.

Items are added to the Hashtable with the Add() method.

The following code creates a Hashtable named mycountries and four elements are added:

```
<script runat="server">
Sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New Hashtable
    mycountries.Add("N","Norway")
    mycountries.Add("S","Sweden")
    mycountries.Add("F","France")
    mycountries.Add("I","Italy")
```

```
end if
end sub
</script>
```

Data Binding

A Hashtable object may automatically generate the text and values to the following controls:

- asp:RadioButtonList
- asp:CheckBoxList
- asp:DropDownList
- asp:Listbox

To bind data to a RadioButtonList control, first create a RadioButtonList control (without any asp:ListItem elements) in an .aspx page:

```
<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server" AutoPostBack="True" />
</form>

</body>
</html>
```

Then add the script that builds the list:

```
<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New Hashtable
    mycountries.Add("N","Norway")
    mycountries.Add("S","Sweden")
    mycountries.Add("F","France")
    mycountries.Add("I","Italy")
    rb.DataSource=mycountries
    rb.DataValueField="Key"
```

<http://www.w3schools.com/aspnet/default.asp>

```
    rb.DataTextField="Value"  
    rb.DataBind()  
end if  
end sub  
</script>  
  
<html>  
<body>  
  
<form runat="server">  
<asp:RadioButtonList id="rb" runat="server" AutoPostBack="True" />  
</form>  
  
</body>  
</html>
```

Then we add a sub routine to be executed when the user clicks on an item in the RadioButtonList control. When a radio button is clicked, a text will appear in a label:

Example

```
<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New Hashtable
    mycountries.Add("N","Norway")
    mycountries.Add("S","Sweden")
    mycountries.Add("F","France")
    mycountries.Add("I","Italy")
    rb.DataSource=mycountries
    rb.DataValueField="Key"
    rb.DataTextField="Value"
    rb.DataBind()
end if
end sub

sub displayMessage(s as Object,e As EventArgs)
lbl1.text="Your favorite country is: " & rb.SelectedItem.Text
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server"
AutoPostBack="True" onSelectedIndexChanged="displayMessage" />
<p><asp:label id="lbl1" runat="server" /></p>
</form>

</body>
</html>
```

Note: You cannot choose the sort order of the items added to the Hashtable. To sort items alphabetically or numerically, use the SortedList object.

Example – HashTable Radibutton list 1

<http://www.w3schools.com/aspnet/default.asp>

```

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New Hashtable
    mycountries.Add("N","Norway")
    mycountries.Add("S","Sweden")
    mycountries.Add("F","France")
    mycountries.Add("I","Italy")
    rb.DataSource=mycountries
    rb.DataValueField="Key"
    rb.DataTextField="Value"
    rb.DataBind()
end if
end sub

sub displayMessage(s as Object,e As EventArgs)
lbl1.text="Your favorite country is: " & rb.SelectedItem.Text
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server"
AutoPostBack="True" onSelectedIndexChanged="displayMessage" />
<p><asp:label id="lbl1" runat="server" /></p>
</form>

</body>
</html>

```

Example – HashTable Radio button list 2

```

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim navigate=New Hashtable
    navigate.Add("RadioButtonList","control_radiobuttonlist.asp")
    navigate.Add("CheckBoxList","control_checkboxlist.asp")
    navigate.Add("DropDownList","control_dropdownlist.asp")
    navigate.Add("ListBox","control_listbox.asp")
    rb.DataSource=navigate

```

```

        rb.DataValueField="Value"
        rb.DataTextField="Key"
        rb.DataBind()
    end if
end sub

sub navigate(s as Object, e As EventArgs)
response.redirect(rb.SelectedItem.Value)
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server"
AutoPostBack="True" onSelectedIndexChanged="navigate" />
</form>

</body>
</html>

```

Example – Hash Table Drop Down list

```

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New Hashtable
    mycountries.Add("N","Norway")
    mycountries.Add("S","Sweden")
    mycountries.Add("F","France")
    mycountries.Add("I","Italy")
    dd.DataSource=mycountries
    dd.DataValueField="Key"
    dd.DataTextField="Value"
    dd.DataBind()
end if
end sub

sub displayMessage(s as Object,e As EventArgs)
lbl1.text="Your favorite country is: " & dd.SelectedItem.Text
http://www.w3schools.com/aspnet/default.asp

```

```
end sub
</script>

<html>
<body>

<form runat="server">
<asp:DropDownList id="dd" runat="server"
AutoPostBack="True" onSelectedIndexChanged="displayMessage" />
<p><asp:label id="lbl1" runat="server" /></p>
</form>

</body>
</html>
```

ASP.NET - The SortedList Object

The SortedList object combines the features of both the ArrayList object and the Hashtable object.

The SortedList Object

The SortedList object contains items in key/value pairs. A SortedList object automatically sort the items in alphabetic or numeric order.

Items are added to the SortedList with the Add() method. A SortedList can be sized to its final size with the TrimToSize() method.

The following code creates a SortedList named mycountries and four elements are added:

```
<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New SortedList
    mycountries.Add("N","Norway")
    mycountries.Add("S","Sweden")
    mycountries.Add("F","France")
    mycountries.Add("I","Italy")
end if
end sub
```

</script>

Data Binding

A SortedList object may automatically generate the text and values to the following controls:

- asp:RadioButtonList
- asp:CheckBoxList
- asp:DropDownList
- asp:Listbox

To bind data to a RadioButtonList control, first create a RadioButtonList control (without any asp:ListItem elements) in an .aspx page:

```
<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server" AutoPostBack="True" />
</form>

</body>
</html>
```

Then add the script that builds the list:

```
<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New SortedList
    mycountries.Add("N","Norway")
    mycountries.Add("S","Sweden")
    mycountries.Add("F","France")
    mycountries.Add("I","Italy")
    rb.DataSource=mycountries
    rb.DataValueField="Key"
    rb.DataTextField="Value"
    rb.DataBind()
end sub
end script
```

<http://www.w3schools.com/aspnet/default.asp>

```
end if
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server" AutoPostBack="True" />
</form>

</body>
</html>
```

Then we add a sub routine to be executed when the user clicks on an item in the RadioButtonList control. When a radio button is clicked, a text will appear in a label:

Example

```
<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New SortedList
    mycountries.Add("N","Norway")
    mycountries.Add("S","Sweden")
    mycountries.Add("F","France")
    mycountries.Add("I","Italy")
    rb.DataSource=mycountries
    rb.DataValueField="Key"
    rb.DataTextField="Value"
    rb.DataBind()
end if
end sub

sub displayMessage(s as Object,e As EventArgs)
lbl1.text="Your favorite country is: " & rb.SelectedItem.Text
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server"
AutoPostBack="True" onSelectedIndexChanged="displayMessage" />
<p><asp:label id="lbl1" runat="server" /></p>
</form>

</body>
</html>
```

Example – Sortlist Radio button List 1

```
<script runat="server">
sub Page_Load
```

```

if Not Page.IsPostBack then
    dim mycountries=New SortedList
    mycountries.Add("N","Norway")
    mycountries.Add("S","Sweden")
    mycountries.Add("F","France")
    mycountries.Add("I","Italy")
    rb.DataSource=mycountries
    rb.DataValueField="Key"
    rb.DataTextField="Value"
    rb.DataBind()
end if
end sub

sub displayMessage(s as Object,e As EventArgs)
    lbl1.text="Your favorite country is: " & rb.SelectedItem.Text
end sub
</script>

```

```

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server"
AutoPostBack="True" onSelectedIndexChanged="displayMessage" />
<p><asp:label id="lbl1" runat="server" /></p>
</form>

</body>
</html>

```

Example – Sort list Radio button List 2

```

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim navigate=New SortedList
    navigate.Add("RadioButtonList","control_radiobuttonlist.asp")
    navigate.Add("CheckBoxList","control_checkboxlist.asp")

```



```

    navigate.Add("DropDownList","control_dropdownlist.asp")
    navigate.Add("ListBox","control_listbox.asp")
    rb.DataSource=navigate
    rb.DataValueField="Value"
    rb.DataTextField="Key"
    rb.DataBind()
end if
end sub

sub navigate(s as Object, e As EventArgs)
response.redirect(rb.SelectedItem.Value)
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server"
AutoPostBack="True" onSelectedIndexChanged="navigate" />
</form>

</body>
</html>

```

Example – sort list Drop down list

```

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New SortedList
    mycountries.Add("N","Norway")
    mycountries.Add("S","Sweden")
    mycountries.Add("F","France")
    mycountries.Add("I","Italy")
    dd.DataSource=mycountries
    dd.DataValueField="Key"
    dd.DataTextField="Value"

```

```
        dd.DataBind()  
    end if  
end sub  
  
sub displayMessage(s as Object,e As EventArgs)  
    lbl1.text="Your favorite country is: " & dd.SelectedItem.Text  
end sub  
</script>  
  
<html>  
<body>  
  
<form runat="server">  
    <asp:DropDownList id="dd" runat="server"  
        AutoPostBack="True" onSelectedIndexChanged="displayMessage" />  
    <p><asp:label id="lbl1" runat="server" /></p>  
</form>  
  
</body>  
</html>
```

ASP .NET - XML Files

We can bind an XML file to a list control.

An XML File

Here is an XML file named "countries.xml":

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
  
<countries>  
  
    <country>  
        <text>Norway</text>  
        <value>N</value>  
    </country>
```

```
<country>
  <text>Sweden</text>
  <value>S</value>
</country>
```

```
<country>
  <text>France</text>
  <value>F</value>
</country>
```

```
<country>
  <text>Italy</text>
  <value>I</value>
</country>
```

```
</countries>
```

Take a look at the XML file: [countries.xml](#)

Bind a DataSet to a List Control

First, import the "System.Data" namespace. We need this namespace to work with DataSet objects. Include the following directive at the top of an .aspx page:

```
<%@ Import Namespace="System.Data" %>
```

Next, create a DataSet for the XML file and load the XML file into the DataSet when the page is first loaded:

```
<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
  dim mycountries=New DataSet
  mycountries.ReadXml(MapPath("countries.xml"))
end if
```

end sub

To bind the DataSet to a RadioButtonList control, first create a RadioButtonList control (without any asp:ListItem elements) in an .aspx page:

```
<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server" AutoPostBack="True" />
</form>

</body>
</html>
```

Then add the script that builds the XML DataSet:

```
<%@ Import Namespace="System.Data" %>

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New DataSet
    mycountries.ReadXml(MapPath("countries.xml"))
    rb.DataSource=mycountries
    rb.DataValueField="value"
    rb.DataTextField="text"
    rb.DataBind()
end if
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server"
```

```
AutoPostBack="True" onSelectedIndexChanged="displayMessage" />
</form>

</body>
</html>
```

Then we add a sub routine to be executed when the user clicks on an item in the RadioButtonList control. When a radio button is clicked, a text will appear in a label:

Example

```
<%@ Import Namespace="System.Data" %>

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New DataSet
    mycountries.ReadXml(MapPath("countries.xml"))
    rb.DataSource=mycountries
    rb.DataValueField="value"
    rb.DataTextField="text"
    rb.DataBind()
end if
end sub

sub displayMessage(s as Object,e As EventArgs)
lbl1.text="Your favorite country is: " & rb.SelectedItem.Text
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server"
AutoPostBack="True" onSelectedIndexChanged="displayMessage" />
<p><asp:label id="lbl1" runat="server" /></p>
</form>

</body>
</html>
```

ASP.NET - The Repeater Control

The Repeater control is used to display a repeated list of items that are bound to the control.

Bind a DataSet to a Repeater Control

The Repeater control is used to display a repeated list of items that are bound to the control. The Repeater control may be bound to a database table, an XML file, or another list of items. Here we will show how to bind an XML file to a Repeater control.

We will use the following XML file in our examples ("cdcatalog.xml"):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<catalog>
```

```
<cd>
```

```
<title>Empire Burlesque</title>
```

```
<artist>Bob Dylan</artist>
```

```
<country>USA</country>
```

```
<company>Columbia</company>
```

```
<price>10.90</price>
```

```
<year>1985</year>
```

```
</cd>
```

```
<cd>
```

```
<title>Hide your heart</title>
```

```
<artist>Bonnie Tyler</artist>
```

```
<country>UK</country>
```

```
<company>CBS Records</company>
```

```
<price>9.90</price>
```

```
<year>1988</year>
```

```
</cd>
```

```
<cd>
```

```
<title>Greatest Hits</title>
```

```
<artist>Dolly Parton</artist>
```

```
<country>USA</country>
```

```
<company>RCA</company>
```

```
<price>9.90</price>
```

```
<year>1982</year>
```

```
</cd>
```

```
<cd>
```

```
<title>Still got the blues</title>
```

```
<artist>Gary Moore</artist>
```

```
<country>UK</country>
<company>Virgin records</company>
<price>10.20</price>
<year>1990</year>
</cd>
<cd>
  <title>Eros</title>
  <artist>Eros Ramazzotti</artist>
  <country>EU</country>
  <company>BMG</company>
  <price>9.90</price>
  <year>1997</year>
</cd>
</catalog>
```

Take a look at the XML file: [cdcatalog.xml](#)

First, import the "System.Data" namespace. We need this namespace to work with DataSet objects. Include the following directive at the top of an .aspx page:

```
<%@ Import Namespace="System.Data" %>
```

Next, create a DataSet for the XML file and load the XML file into the DataSet when the page is first loaded:

```
<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
  dim mycdcatalog=New DataSet
  mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
end if
end sub
```

Then we create a Repeater control in an .aspx page. The contents of the <HeaderTemplate> element are rendered first and only once within the output, then the contents of the <ItemTemplate> element are repeated for each "record" in the DataSet, and last, the contents of the <FooterTemplate> element are rendered once within the output:


```
<html>
<body>

<form runat="server">
<asp:Repeater id="cdcatalog" runat="server">

<HeaderTemplate>
...
</HeaderTemplate>

<ItemTemplate>
...
</ItemTemplate>

<FooterTemplate>
...
</FooterTemplate>

</asp:Repeater>
</form>

</body>
</html>
```

Then we add the script that creates the DataSet and binds the mycdc catalog DataSet to the Repeater control. We also fill the Repeater control with HTML tags and bind the data items to the cells in the <ItemTemplate> section with the <%#Container.DataItem("fieldname")%> method:

Example

```
<%@ Import Namespace="System.Data" %>

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycdcatalog=New DataSet
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
    cdcatalog.DataSource=mycdcatalog
    cdcatalog.DataBind()
end if
end sub
</script>

<html>
<body>

<form runat="server">
<asp:Repeater id="cdcatalog" runat="server">

<HeaderTemplate>
<table border="1" width="100%">
<tr>
<th>Title</th>
<th>Artist</th>
<th>Country</th>
<th>Company</th>
<th>Price</th>
<th>Year</th>
</tr>
</HeaderTemplate>

<ItemTemplate>
<tr>
<td><%#Container.DataItem("title")%></td>
<td><%#Container.DataItem("artist")%></td>
<td><%#Container.DataItem("country")%></td>
```

<http://www.w3schools.com/aspnet/default.asp>

```
<td><%#Container.DataItem("company")%></td>
<td><%#Container.DataItem("price")%></td>
<td><%#Container.DataItem("year")%></td>
</tr>
</ItemTemplate>

<FooterTemplate>
</table>
</FooterTemplate>

</asp:Repeater>
</form>

</body>
</html>
```

Using the <AlternatingItemTemplate>

You can add an <AlternatingItemTemplate> element after the <ItemTemplate> element to describe the appearance of alternating rows of output. In the following example each other row in the table will be displayed in a light grey color:

Example

```
<%@ Import Namespace="System.Data" %>

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycdcatalog=New DataSet
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
    cdcatalog.DataSource=mycdcatalog
    cdcatalog.DataBind()
end if
end sub
</script>

<html>
<body>

<form runat="server">
<asp:Repeater id="cdcatalog" runat="server">

<HeaderTemplate>
<table border="1" width="100%">
<tr>
<th>Title</th>
<th>Artist</th>
<th>Country</th>
<th>Company</th>
<th>Price</th>
<th>Year</th>
</tr>
</HeaderTemplate>

<ItemTemplate>
<tr>
<td><%#Container.DataItem("title")%></td>
<td><%#Container.DataItem("artist")%></td>
<td><%#Container.DataItem("country")%></td>
```

<http://www.w3schools.com/aspnet/default.asp>

```

<td><%#Container.DataItem("company")%></td>
<td><%#Container.DataItem("price")%></td>
<td><%#Container.DataItem("year")%></td>
</tr>
</ItemTemplate>

```

```

<AlternatingItemTemplate>
<tr bgcolor="#e8e8e8">
<td><%#Container.DataItem("title")%></td>
<td><%#Container.DataItem("artist")%></td>
<td><%#Container.DataItem("country")%></td>
<td><%#Container.DataItem("company")%></td>
<td><%#Container.DataItem("price")%></td>
<td><%#Container.DataItem("year")%></td>
</tr>
</AlternatingItemTemplate>

```

```

<FooterTemplate>
</table>
</FooterTemplate>

```

```

</asp:Repeater>
</form>

```

```

</body>
</html>

```

Using the <SeparatorTemplate>

The <SeparatorTemplate> element can be used to describe a separator between each record. The following example inserts a horizontal line between each table row:

Example

```
<%@ Import Namespace="System.Data" %>

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycdcatalog=New DataSet
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
    cdcatalog.DataSource=mycdcatalog
    cdcatalog.DataBind()
end if
end sub
</script>

<html>
<body>

<form runat="server">
<asp:Repeater id="cdcatalog" runat="server">

<HeaderTemplate>
<table border="0" width="100%">
<tr>
<th>Title</th>
<th>Artist</th>
<th>Country</th>
<th>Company</th>
<th>Price</th>
<th>Year</th>
</tr>
</HeaderTemplate>

<ItemTemplate>
<tr>
<td><%#Container.DataItem("title")%></td>
<td><%#Container.DataItem("artist")%></td>
<td><%#Container.DataItem("country")%></td>
```

<http://www.w3schools.com/aspnet/default.asp>

```
<td><%#Container.DataItem("company")%></td>
<td><%#Container.DataItem("price")%></td>
<td><%#Container.DataItem("year")%></td>
</tr>
</ItemTemplate>

<SeparatorTemplate>
<tr>
<td colspan="6"><hr /></td>
</tr>
</SeparatorTemplate>

<FooterTemplate>
</table>
</FooterTemplate>

</asp:Repeater>
</form>

</body>
</html>
```

ASP.NET - The DataList Control

The DataList control is, like the Repeater control, used to display a repeated list of items that are bound to the control. However, the DataList control adds a table around the data items by default.

Bind a DataSet to a DataList Control

The DataList control is, like the Repeater control, used to display a repeated list of items that are bound to the control. However, the DataList control adds a table around the data items by default. The DataList control may be bound to a database table, an XML file, or another list of items. Here we will show how to bind an XML file to a DataList control.

We will use the following XML file in our examples ("cdcatalog.xml"):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

<http://www.w3schools.com/aspnet/default.asp>

```
<catalog>
<cd>
  <title>Empire Burlesque</title>
  <artist>Bob Dylan</artist>
  <country>USA</country>
  <company>Columbia</company>
  <price>10.90</price>
  <year>1985</year>
</cd>
<cd>
  <title>Hide your heart</title>
  <artist>Bonnie Tyler</artist>
  <country>UK</country>
  <company>CBS Records</company>
  <price>9.90</price>
  <year>1988</year>
</cd>
<cd>
  <title>Greatest Hits</title>
  <artist>Dolly Parton</artist>
  <country>USA</country>
  <company>RCA</company>
  <price>9.90</price>
  <year>1982</year>
</cd>
<cd>
  <title>Still got the blues</title>
  <artist>Gary Moore</artist>
  <country>UK</country>
  <company>Virgin records</company>
  <price>10.20</price>
  <year>1990</year>
</cd>
<cd>
  <title>Eros</title>
  <artist>Eros Ramazzotti</artist>
```



```
<country>EU</country>
<company>BMG</company>
<price>9.90</price>
<year>1997</year>
</cd>
</catalog>
```

Take a look at the XML file: [cdcatalog.xml](#)

First, import the "System.Data" namespace. We need this namespace to work with DataSet objects. Include the following directive at the top of an .aspx page:

```
<%@ Import Namespace="System.Data" %>
```

Next, create a DataSet for the XML file and load the XML file into the DataSet when the page is first loaded:

```
<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycdcatalog=New DataSet
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
end if
end sub
```

Then we create a DataList in an .aspx page. The contents of the <HeaderTemplate> element are rendered first and only once within the output, then the contents of the <ItemTemplate> element are repeated for each "record" in the DataSet, and last, the contents of the <FooterTemplate> element are rendered once within the output:

```
<html>
<body>

<form runat="server">
<asp:DataList id="cdcatalog" runat="server">

<HeaderTemplate>
...

```

```
</HeaderTemplate>

<ItemTemplate>
...
</ItemTemplate>

<FooterTemplate>
...
</FooterTemplate>

</asp:DataList>
</form>

</body>
</html>
```

Then we add the script that creates the DataSet and binds the mycdcatalog DataSet to the DataList control. We also fill the DataList control with a <HeaderTemplate> that contains the header of the table, an <ItemTemplate> that contains the data items to display, and a <FooterTemplate> that contains a text. Note that the gridlines attribute of the DataList is set to "both" to display table borders:

Example

```
<%@ Import Namespace="System.Data" %>

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycdcatalog=New DataSet
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
    cdcatalog.DataSource=mycdcatalog
    cdcatalog.DataBind()
end if
end sub
</script>

<html>
<body>

<form runat="server">
<asp:DataList id="cdcatalog"
gridlines="both" runat="server">

<HeaderTemplate>
My CD Catalog
</HeaderTemplate>

<ItemTemplate>
"<%=Container.DataItem("title")%>" of
<%=Container.DataItem("artist")%> -
$<%=Container.DataItem("price")%>
</ItemTemplate>

<FooterTemplate>
Copyright Hege Refsnes
</FooterTemplate>

</asp:DataList>
</form>
```

<http://www.w3schools.com/aspnet/default.asp>

```
</body>  
</html>
```

Using Styles

You can also add styles to the DataList control to make the output more fancy:

Example

```
<%@ Import Namespace="System.Data" %>

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycdcatalog=New DataSet
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
    cdcatalog.DataSource=mycdcatalog
    cdcatalog.DataBind()
end if
end sub
</script>

<html>
<body>

<form runat="server">
<asp:DataList id="cdcatalog"
runat="server"
cellpadding="2"
cellspacing="2"
borderstyle="inset"
backcolor="#e8e8e8"
width="100%"
headerstyle-font-name="Verdana"
headerstyle-font-size="12pt"
headerstyle-horizontalalign="center"
headerstyle-font-bold="true"
itemstyle-backcolor="#778899"
itemstyle-forecolor="ffffff"
footerstyle-font-size="9pt"
footerstyle-font-italic="true">

<HeaderTemplate>
My CD Catalog
</HeaderTemplate>
```

<http://www.w3schools.com/aspnet/default.asp>

```
<ItemTemplate>
"<%#Container.DataItem("title")%>" of
<%#Container.DataItem("artist")%> -
$<%#Container.DataItem("price")%>
</ItemTemplate>

<FooterTemplate>
Copyright Hege Refsnes
</FooterTemplate>

</asp:DataList>
</form>

</body>
</html>
```

Using the <AlternatingItemTemplate>

You can add an <AlternatingItemTemplate> element after the <ItemTemplate> element to describe the appearance of alternating rows of output. You may style the data in the <AlternatingItemTemplate> section within the DataList control:

Example

```
<%@ Import Namespace="System.Data" %>

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
dim mycdcatalog=New DataSet
mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
cdcatalog.DataSource=mycdcatalog
cdcatalog.DataBind()
end if
end sub
</script>

<html>
<body>

<form runat="server">
<asp:DataList id="cdcatalog"
runat="server"
cellpadding="2"
cellspacing="2"
borderstyle="inset"
backcolor="#e8e8e8"
width="100%"
headerstyle-font-name="Verdana"
headerstyle-font-size="12pt"
headerstyle-horizontalalign="center"
headerstyle-font-bold="True"
itemstyle-backcolor="#778899"
itemstyle-forecolor="#ffffff"
alternatingitemstyle-backcolor="#e8e8e8"
alternatingitemstyle-forecolor="#000000"
footerstyle-font-size="9pt"
footerstyle-font-italic="True">

<HeaderTemplate>
```

<http://www.w3schools.com/aspnet/default.asp>

My CD Catalog

</HeaderTemplate>

<ItemTemplate>

"<#Container.DataItem("title")%>" of

<#Container.DataItem("artist")%> -

\$<#Container.DataItem("price")%>

</ItemTemplate>

<AlternatingItemTemplate>

"<#Container.DataItem("title")%>" of

<#Container.DataItem("artist")%> -

\$<#Container.DataItem("price")%>

</AlternatingItemTemplate>

<FooterTemplate>

© Hege Refsnes

</FooterTemplate>

</asp:DataList>

</form>

</body>

</html>

ASP.NET - Database Connection

ADO.NET is also a part of the .NET Framework. ADO.NET is used to handle data access. With ADO.NET you can work with databases.

What is ADO.NET?

- ADO.NET is a part of the .NET Framework
- ADO.NET consists of a set of classes used to handle data access
- ADO.NET is entirely based on XML
- ADO.NET has, unlike ADO, no Recordset object

Create a Database Connection

We are going to use the Northwind database in our examples.

First, import the "System.Data.OleDb" namespace. We need this namespace to work with Microsoft Access and other OLE DB database providers. We will create the connection to the database in the Page_Load subroutine. We create a dbconn variable as a new OleDbConnection class with a connection string which identifies the OLE DB provider and the location of the database. Then we open the database connection:

```
<%@ Import Namespace="System.Data.OleDb" %>

<script runat="server">
sub Page_Load
dim dbconn
dbconn=New OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;
data source=" & server.mappath("northwind.mdb"))
dbconn.Open()
end sub
</script>
```

Note: The connection string must be a continuous string without a line break!

Create a Database Command

To specify the records to retrieve from the database, we will create a dbcomm variable as a new OleDbCommand class. The OleDbCommand class is for issuing SQL queries against database tables:

```
<%@ Import Namespace="System.Data.OleDb" %>

<script runat="server">
sub Page_Load
dim dbconn,sql,dbcomm
dbconn=New OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;
data source=" & server.mappath("northwind.mdb"))
dbconn.Open()
sql="SELECT * FROM customers"
```

```
dbcomm=New OleDbCommand(sql,dbconn)
end sub
</script>
```

Create a DataReader

The OleDbDataReader class is used to read a stream of records from a data source. A DataReader is created by calling the ExecuteReader method of the OleDbCommand object:

```
<%@ Import Namespace="System.Data.OleDb" %>

<script runat="server">
sub Page_Load
dim dbconn,sql,dbcomm,dbread
dbconn=New OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;
data source=" & server.mappath("northwind.mdb"))
dbconn.Open()
sql="SELECT * FROM customers"
dbcomm=New OleDbCommand(sql,dbconn)
dbread=dbcomm.ExecuteReader()
end sub
</script>
```

Bind to a Repeater Control

Then we bind the DataReader to a Repeater control:

Example

```
<%@ Import Namespace="System.Data.OleDb" %>

<script runat="server">
sub Page_Load
dim dbconn,sql,dbcomm,dbread
dbconn=New OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;
data source=" & server.mappath("northwind.mdb"))
dbconn.Open()
sql="SELECT * FROM customers"
dbcomm=New OleDbCommand(sql,dbconn)
dbread=dbcomm.ExecuteReader()
customers.DataSource=dbread
customers.DataBind()
dbread.Close()
dbconn.Close()
end sub
</script>

<html>
<body>

<form runat="server">
<asp:Repeater id="customers" runat="server">

<HeaderTemplate>
<table border="1" width="100%">
<tr>
<th>Companyname</th>
<th>Contactname</th>
<th>Address</th>
<th>City</th>
</tr>
</HeaderTemplate>

<ItemTemplate>
<tr>
```

```

<td><%#Container.DataItem("companyname")%></td>
<td><%#Container.DataItem("contactname")%></td>
<td><%#Container.DataItem("address")%></td>
<td><%#Container.DataItem("city")%></td>
</tr>
</ItemTemplate>

<FooterTemplate>
</table>
</FooterTemplate>

</asp:Repeater>
</form>

</body>
</html>

```

Close the Database Connection

Always close both the DataReader and database connection after access to the database is no longer required:

```

dbread.Close()
dbconn.Close()

```

Example – Database connection bind to a datalist control

```

<%@ Import Namespace="System.Data.OleDb" %>

<script runat="server">
sub Page_Load
dim dbconn,sql,dbcomm,dbread
dbconn=New OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;data
source=" & server.mappath("/db/northwind.mdb"))
dbconn.Open()
sql="SELECT * FROM customers"
dbcomm=New OleDbCommand(sql,dbconn)
dbread=dbcomm.ExecuteReader()
customers.DataSource=dbread
http://www.w3schools.com/aspnet/default.asp

```

```
customers.DataBind()  
dbread.Close()  
dbconn.Close()  
end sub  
</script>
```

```
<html>  
<body>
```

```
<form runat="server">  
<asp:DataList  
id="customers"  
runat="server"  
cellpadding="2"  
cellspacing="2"  
borderstyle="inset"  
backcolor="#e8e8e8"  
width="100%"  
headerstyle-font-name="Verdana"  
headerstyle-font-size="12pt"  
headerstyle-horizontalalign="center"  
headerstyle-font-bold="True"  
itemstyle-backcolor="#778899"  
itemstyle-forecolor="#ffffff"  
footerstyle-font-size="9pt"  
footerstyle-font-italic="True">
```

```
<HeaderTemplate>  
Customers Table  
</HeaderTemplate>
```

```
<ItemTemplate>  
<%#Container.DataItem("companyname")%> in  
<%#Container.DataItem("address")%> ,  
<%#Container.DataItem("city")%>  
</ItemTemplate>
```

```
<FooterTemplate>  
Source: Northwind Database  
</FooterTemplate>
```

```
</asp:DataList>  
</form>
```

```
</body>  
</html>
```

Example – Database connection bind to a Repeater control

```
<%@ Import Namespace="System.Data.OleDb" %>  
  
<script runat="server">  
sub Page_Load  
dim dbconn,sql,dbcomm,dbread  
dbconn=New OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;data  
source=" & server.mappath("/db/northwind.mdb"))  
dbconn.Open()  
sql="SELECT * FROM customers"  
dbcomm=New OleDbCommand(sql,dbconn)  
dbread=dbcomm.ExecuteReader()  
customers.DataSource=dbread  
customers.DataBind()  
dbread.Close()  
dbconn.Close()  
end sub  
</script>  
  
<html>  
<body>  
  
<form runat="server">  
<asp:Repeater id="customers" runat="server">  
  
<HeaderTemplate>  
<table border="1" width="100%">
```

```
<tr bgcolor="#b0c4de">
<th>Companyname</th>
<th>Contactname</th>
<th>Address</th>
<th>City</th>
</tr>
</HeaderTemplate>

<ItemTemplate>
<tr bgcolor="#f0f0f0">
<td><%#Container.DataItem("companyname")%> </td>
<td><%#Container.DataItem("contactname")%> </td>
<td><%#Container.DataItem("address")%> </td>
<td><%#Container.DataItem("city")%> </td>
</tr>
</ItemTemplate>

<FooterTemplate>
</table>
</FooterTemplate>

</asp:Repeater>
</form>

</body>
</html>
```

ASP.NET 2.0 - New Features

ASP.NET 2.0 improves ASP.NET by adding several new features.

Improvements in ASP.NET 2.0

ASP.NET 2.0 was designed to make web development easier and quicker.

Design goals for ASP.NET 2.0:

- Increase productivity by removing 70% of the code

<http://www.w3schools.com/aspnet/default.asp>

- Use the same controls for all types of devices
 - Provide a faster and better web server platform
 - Simplify compilation and installation
 - Simplify the administration of web applications
-

What's New in ASP.NET 2.0?

Some of the new features in ASP.NET 2.0 are:

- Master Pages, Themes, and Web Parts
- Standard controls for navigation
- Standard controls for security
- Roles, personalization, and internationalization services
- Improved and simplified data access controls
- Full support for XML standards like, XHTML, XML, and WSDL
- Improved compilation and deployment (installation)
- Improved site management
- New and improved development tools

The new features are described below.

Master Pages

ASP.NET didn't have a method for applying a consistent look and feel for a whole web site.

Master pages in ASP.NET 2.0 solves this problem.

A master page is a template for other pages, with shared layout and functionality. The master page defines placeholders for content pages. The result page is a combination (merge) of the master page and the content page.

[Read more about master pages.](#)

Themes

Themes is another feature of ASP.NET 2.0. Themes, or skins, allow developers to create a customized look for web applications.

Design goals for ASP.NET 2.0 themes:

- Make it simple to customize the appearance of a site
 - Allow themes to be applied to controls, pages, and entire sites
 - Allow all visual elements to be customized
-

Web Parts

ASP.NET 2.0 Web Parts can provide a consistent look for a site, while still allowing user customization of style and content.

New controls:

- Zone controls - areas on a page where the content is consistent
 - Web part controls - content areas for each zone
-

Navigation

ASP.NET 2.0 has built-in navigation controls like

- Site Maps
 - Dynamic HTML menus
 - Tree Views
-

Security

Security is very important for protecting confidential and personal information.

In ASP.NET 2.0 the following controls has been added:

- A Login control, which provides login functionality
 - A LoginStatus control, to control the login status
 - A LoginName control to display the current user name
 - A LoginView control, to provide different views depending on login status
 - A CreateUser wizard, to allow creation of user accounts
 - A PasswordRecovery control, to provide the "I forgot my password" functionality
-

Roles and Personalization

Internet communities are growing very popular.

ASP.NET 2.0 has personalization features for storing user details. This provides an easy way to customize user (and user group) properties.

Internationalization

Reaching people with different languages is important if you want to reach a larger audience.

ASP.NET 2.0 has improved support for multiple languages.

Data Access

Many web sites are data driven, using databases or XML files as data sources.

With ASP.NET this involved code, and often the same code had to be used over and over in different web pages.

A key goal of ASP.NET 2.0 was to ease the use of data sources.

ASP.NET 2.0 has new data controls, removing much of the need for programming and in-depth knowledge of data connections.

Mobility Support

The problem with Mobile devices is screen size and display capabilities.

In ASP.NET, the Microsoft Mobile Internet Toolkit (MMIT) provided this support.

In ASP.NET 2.0, MMIT is no longer needed because mobile support is built into all controls.

Images

ASP.NET 2.0 has new controls for handling images:

<http://www.w3schools.com/aspnet/default.asp>

- The ImageMap control - image map support
- The DynamicImage control - image support for different browsers

These controls are important for better image display on mobile devices, like hand-held computers and cell phones.

Automatic Compilation

ASP.NET 2.0 provides automatic compilation. All files within a directory will be compiled on the first run, including support for WSDL, and XSD files.

Compiled Deployment (Installation) and Source Protection

ASP.NET 2.0 also provides pre-compilation. An entire web site can be pre-compiled. This provides an easy way to deploy (upload to a server) compiled applications, and because only compiled files are deployed, the source code is protected.

Site Management

ASP.NET 2.0 has three new features for web site configuration and management:

- New local management console
 - New programmable management functions (API)
 - New web-based management tool
-

Development Tools

With ASP.NET Visual Studio.NET was released with project and design features targeted at corporate developers.

With ASP.NET 2.0, Visual Studio 2005 was released.

Key design features for Visual Studio 2005 include:

- Support for the features described above
- Upload files from anywhere (FTP, File System, Front Page....)

- No project files, allowing code to be manipulated outside Visual Studio
- Integrated Web Site Administration Tool
- No "build" step - ability to compile on first run

Visual Web Developer is a new free ASP.NET 2.0 tool for non-corporate developers who don't have access to Visual Studio.NET.

ASP.NET 2.0 - Master Pages

Master pages provide templates for other pages on your web site.

Master Pages

Master pages allow you to create a consistent look and behavior for all the pages (or group of pages) in your web application.

A master page provides a template for other pages, with shared layout and functionality. The master page defines placeholders for the content, which can be overridden by content pages. The output result is a combination of the master page and the content page.

The content pages contains the content you want to display.

When users request the content page, ASP.NET merges the pages to produce output that combines the layout of the master page with the content of the content page.

Master Page Example

```
<%@ Master %>
```

```
<html>
<body>
<h1>Standard Header For All Pages</h1>
<asp:ContentPlaceholder id="CPH1" runat="server">
</asp:ContentPlaceholder>
</body>
</html>
```

The master page above is a normal HTML page designed as a template for other pages.

The @ **Master** directive defines it as a master page.

The master page contains a placeholder tag **<asp:ContentPlaceHolder>** for individual content.

The **id="CPH1"** attribute identifies the placeholder, allowing many placeholders in the same master page.

This master page was saved with the name "**master1.master**".

 Note: The master page can also contain code, allowing dynamic content.

Content Page Example

```
<%@ Page MasterPageFile="master1.master" %>
```

```
<asp:Content ContentPlaceHolderId="CPH1" runat="server">  
  <h2>Individual Content</h2>  
  <p>Paragraph 1</p>  
  <p>Paragraph 2</p>  
</asp:Content>
```

The content page above is one of the individual content pages of the web.


The @ **Page** directive defines it as a standard content page.

The content page contains a content tag **<asp:Content>** with a reference to the master page (ContentPlaceHolderId="CPH1").

This content page was saved with the name "**mypage1.aspx**".

When the user requests this page, ASP.NET merges the content page with the master page.

[Click to display mypage1.aspx.](#)

 Note: The content text must be inside the **<asp:Content>** tag. No content is allowed outside the tag.

Content Page With Controls

```
<%@ Page MasterPageFile="master1.master" %>
```

```
<asp:Content ContentPlaceHolderId="CPH1" runat="server">
  <h2>W3Schools</h2>
  <form runat="server">
    <asp:TextBox id="textbox1" runat="server" />
    <asp:Button id="button1" runat="server" text="Button" />
  </form>
</asp:Content>
```

The content page above demonstrates how .NET controls can be inserted into the content page just like an into an ordinary page.

ASP.NET 2.0 - Navigation

ASP.NET 2.0 has built-in navigation controls

Web Site Navigation

Maintaining the menu of a large web site is difficult and time consuming.

In ASP.NET 2.0 the menu can be stored in a file to make it easier to maintain. This file is normally called **web.sitemap**, and is stored in the root directory of the web.

In addition, ASP.NET 2.0 has three new navigation controls:

- Dynamic menus
- TreeViews
- Site Map Path

The Sitemap File


The following sitemap file is used in this tutorial:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<siteMap>
  <siteMapNode title="Home" url="/aspnet/w3home.aspx">
    <siteMapNode title="Services" url="/aspnet/w3services.aspx">
```

```
<siteMapNode title="Training" url="/aspnet/w3training.aspx"/>
<siteMapNode title="Support" url="/aspnet/w3support.aspx"/>
</siteMapNode>
</siteMapNode>
</siteMap>
```

Rules for creating a sitemap file:

- The XML file must contain a <siteMap> tag surrounding the content
- The <siteMap> tag can only have one <siteMapNode> child node (the "home" page)
- Each <siteMapNode> can have several child nodes (web pages)
- Each <siteMapNode> has attributes defining page title and URL

 **Note:** The sitemap file must be placed in the root directory of the web and the URL attributes must be relative to the root directory.

Dynamic Menu

The <asp:Menu> control displays a standard site navigation menu.

Code Example:

```
<asp:SiteMapDataSource id="nav1" runat="server" />

<form runat="server">
<asp:Menu runat="server" DataSourceId="nav1" />
</form>
```

The <asp:Menu> control in the example above is a placeholder for a server created navigation menu.

The data source of the control is defined by the **DataSourceId** attribute. The **id="nav1"** connects it to the <asp:SiteMapDataSource> control.

The <asp:SiteMapDataSource> control automatically connects to the default sitemap file (**web.sitemap**).

[Click here to see a demo of Menu, TreeView, and SiteMapPath](#)

TreeView

The <asp:TreeView> control displays a multi level navigation menu.

The menu looks like a tree with branches that can be opened or closed with + or - symbol.

Code Example:

```
<asp:SiteMapDataSource id="nav1" runat="server" />

<form runat="server">
<asp:TreeView runat="server" DataSourceId="nav1" />
</form>
```

The <asp:TreeView> control in the example above is a placeholder for a server created navigation menu.

The data source of the control is defined by the **DataSourceId** attribute. The **id="nav1"** connects it to the <asp:SiteMapDataSource> control.

The <asp:SiteMapDataSource> control automatically connects to the default sitemap file (**web.sitemap**).

[Click here to see a demo of Menu, TreeView, and SiteMapPath](#)

SiteMapPath

The SiteMapPath control displays the trail (navigation path) to the current page. The path acts as clickable links to previous pages.

Unlike the TreeView and Menu control the SiteMapPath control does **NOT** use a SiteMapDataSource. The SiteMapPath control uses the web.sitemap file by default.

💡 **Tips:** If the SiteMapPath displays incorrectly, most likely there is an URL error (typo) in the web.sitemap file.

Code Example:

```
<form runat="server">
```

<http://www.w3schools.com/aspnet/default.asp>


```
<asp:SiteMapPath runat="server" />
</form>
```

The **<asp:SiteMapPath>** control in the example above is a placeholder for a server created site path display.

HTML Server Controls

HTML server controls are HTML tags understood by the server.

HTML Server Controls

HTML elements in ASP.NET files are, by default, treated as text. To make these elements programmable, add a `runat="server"` attribute to the HTML element. This attribute indicates that the element should be treated as a server control.

Note: All HTML server controls must be within a `<form>` tag with the `runat="server"` attribute!

Note: ASP.NET requires that all HTML elements must be properly closed and properly nested.

HTML Server Control	Description
HtmlAnchor	Controls an <code><a></code> HTML element
HtmlButton	Controls a <code><button></code> HTML element
HtmlForm	Controls a <code><form></code> HTML element
HtmlGeneric	Controls other HTML element not specified by a specific HTML server control, like <code><body></code> , <code><div></code> , <code></code> , etc.
HtmlImage	Controls an <code><image></code> HTML element
HtmlInputButton	Controls <code><input type="button"></code> , <code><input type="submit"></code> , and <code><input type="reset"></code> HTML elements
HtmlInputCheckBox	Controls an <code><input type="checkbox"></code> HTML element
HtmlInputFile	Controls an <code><input type="file"></code> HTML element

HtmlInputHidden	Controls an <input type="hidden"> HTML element
HtmlInputImage	Controls an <input type="image"> HTML element
HtmlInputRadioButton	Controls an <input type="radio"> HTML element
HtmlInputText	Controls <input type="text"> and <input type="password"> HTML elements
HtmlSelect	Controls a <select> HTML element
HtmlTable	Controls a <table> HTML element
HtmlTableCell	Controls <td>and <th> HTML elements
HtmlTableRow	Controls a <tr> HTML element
HtmlTextArea	Controls a <textarea> HTML element

Web Server Controls

Web server controls are special ASP.NET tags understood by the server.

Web Server Controls

Like HTML server controls, Web server controls are also created on the server and they require a `runat="server"` attribute to work. However, Web server controls do not necessarily map to any existing HTML elements and they may represent more complex elements.

The syntax for creating a Web server control is:

```
<asp:control_name id="some_id" runat="server" />
```

Web Server Control	Description
AdRotator	Displays a sequence of images

Button	Displays a push button
Calendar	Displays a calendar
CalendarDay	A day in a calendar control
CheckBox	Displays a check box
CheckBoxList	Creates a multi-selection check box group
DataGrid	Displays fields of a data source in a grid
DataList	Displays items from a data source by using templates
DropDownList	Creates a drop-down list
HyperLink	Creates a hyperlink
Image	Displays an image
ImageButton	Displays a clickable image
Label	Displays static content which is programmable (lets you apply styles to its content)
LinkButton	Creates a hyperlink button
ListBox	Creates a single- or multi-selection drop-down list
ListItem	Creates an item in a list
Literal	Displays static content which is programmable(does not let you apply styles to its content)
Panel	Provides a container for other controls
Placeholder	Reserves space for controls added by code
RadioButton	Creates a radio button
RadioButtonList	Creates a group of radio buttons
BulletedList	Creates a list in bullet format

Repeater	Displays a repeated list of items bound to the control
Style	Sets the style of controls
Table	Creates a table
TableCell	Creates a table cell
TableRow	Creates a table row
TextBox	Creates a text box
Xml	Displays an XML file or the results of an XSL transform

Validation Server Controls

Validation server controls are used to validate user-input.

Validation Server Controls

A Validation server control is used to validate the data of an input control. If the data does not pass validation, it will display an error message to the user.

The syntax for creating a Validation server control is:

```
<asp:control_name id="some_id" runat="server" />
```

Validation Server Control	Description
CompareValidator	Compares the value of one input control to the value of another input control or to a fixed value
CustomValidator	Allows you to write a method to handle the validation of the value entered
RangeValidator	Checks that the user enters a value that falls between two values
RegularExpressionValidator	Ensures that the value of an input control matches a

	specified pattern
RequiredFieldValidator	Makes an input control a required field
ValidationSummary	Displays a report of all validation errors occurred in a Web page

ASP.NET Examples

1) HTML Anchor

```

<script runat="server">
Sub Page_Load
    link1.HRef="http://www.w3schools.com"
    link1.Target="_blank"
    link1.Title="W3Schools"

    link2.HRef="http://www.microsoft.com"
    link2.Target="_blank"
    link2.Title="Microsoft"
End Sub
</script>

<html>
<body>

<form runat="server">
<a id="link1" runat="server">Visit W3Schools!</a>
<br />
<a id="link2" runat="server">Visit Microsoft!</a>
</form>

</body>
</html>

```

Output

Output Result:

[Visit W3Schools!](#)
[Visit Microsoft!](#)

2) Html Button

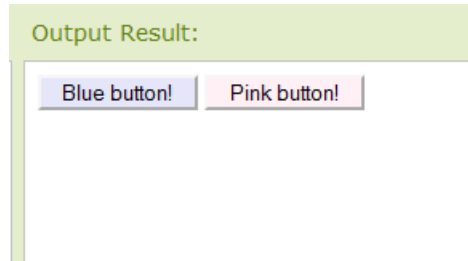
```
<script runat="server">
Sub button1(Source As Object, e As EventArgs)
    p1.InnerHtml="You clicked the blue button!"
End Sub
Sub button2(Source As Object, e As EventArgs)
    p1.InnerHtml="You clicked the pink button!"
End Sub
</script>
```

```
<html>
<body>
```

```
<form runat="server">
<button id="b1" OnServerClick="button1"
style="background-color: #e6e6fa;
height=25;width:100" runat="server">
Blue button!
</button>
<button id="b2"
OnServerClick="button2"
style="background-color: #fff0f5;
height=25;width:100" runat="server">
Pink button!
</button>
<p id="p1" runat="server" />
</form>
```

```
</body>  
</html>
```

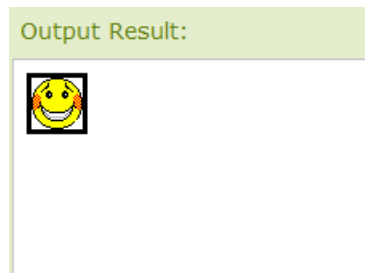
Output



3) HTML Image

```
<script runat="server">  
Sub Page_Load(Sender As Object,E As EventArgs)  
    image1.Src="smiley.gif"  
    image1.Alt="Smiley"  
    image1.Border="3"  
End Sub  
</script>  
  
<html>  
<body>  
  
<form runat="server">  
<img id="image1" runat="server" />  
</form>  
  
</body>  
</html>
```

Output



4) HTML Image 2

```
<script runat="server">
Sub choose_image(Sender As Object, e As EventArgs)
    image1.Src = select1.Value
End Sub
</script>

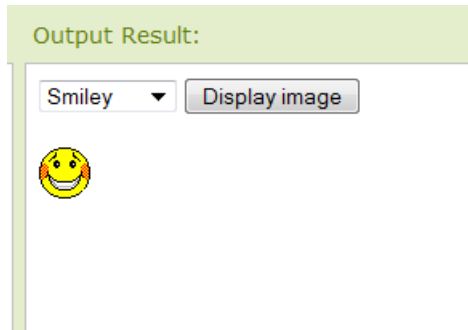
<html>
<body>

<form runat="server">
<select id="select1" runat="server">
    <option value="smiley.gif">Smiley</option>
    <option value="angry.gif">Angry</option>
    <option value="stickman.gif">Stickman</option>
</select>
<input type="submit" runat="server" value="Display image"
OnServerClick="choose_image">
<br /><br />

</form>

</body>
</html>
```

Output



5) HTML Input Button

```
<script runat="server">
Sub submit(sender As Object, e as EventArgs)
if name.value<>"" then
    p1.InnerHtml="Welcome " & name.value & "!"
end if
End Sub
</script>

<html>
<body>

<form runat="server">
Enter your name: <input id="name" type="text" size="30" runat="server"
/>
<br /><br />
<input type="submit" value="Submit" OnServerClick="submit"
runat="server" />
<p id="p1" runat="server" />
</form>

</body>
</html>
```

Output

Output Result:

Enter your name:

6) HTML Input CheckBox

```
<script runat="server">
Sub submit(Source As Object, e As EventArgs)
if red.Checked=True then
    p1.InnerHtml="You prefer red!"
else
    p1.InnerHtml="You prefer blue!"
end if
red.checked=false
blue.checked=false
End Sub
</script>

<html>
<body>

<form runat="server">
What color do you prefer?
<br />
<input id="red" type="checkbox" runat="server" /> Red
<br />
<input id="blue" type="checkbox" runat="server" /> Blue
<br />
<input type="button" value="Submit" OnServerClick="submit"
runat="server"/>
<p id="p1" runat="server" />
</form>
```

```
</body>
</html>
```

Output

Output Result:

What color do you prefer?

☐ Red

☐ Blue

7) HTML Input Hidden

```
<script runat="server">
Sub submit(Source As Object, e As EventArgs)
    hidden1.Value=string1.Value
    p1.InnerHtml="Hidden value= " + hidden1.Value
End Sub
</script>

<html>
<body>

<form runat="server">
Enter some text: <input id="string1" type="text" size="25" runat="server"
/>
<input type="submit" value="Submit" OnServerClick="submit"
runat="server" />
<input id="hidden1" type="hidden" runat="server" />
<p id="p1" runat="server" />
</form>

</body>
</html>
```

Output

Output Result:

Enter some text:

Hidden value= karan

8) HTML Input Image

```

<script runat="server">
Sub button1(Source As Object, e As ImageClickEventArgs)
    p1.InnerHtml="You clicked the smiley button!"
End Sub
Sub button2(Source As Object, e As ImageClickEventArgs)
    p1.InnerHtml="You clicked the angry button!"
End Sub
</script>

<html>
<body>

<form runat="server">
<p>Click on one of the images:</p>
<p>
<input type="image" src="smiley.gif"
OnServerClick="button1" runat="server" width="32" height="32" />
</p>
<p>
<input type="image" src="angry.gif"
OnServerClick="button2" runat="server" width="32" height="32" />
</p>
<p id="p1" runat="server" />
</form>

</body>
</html>

```

Output

Output Result:

Click on one of the images:



You clicked the smiley button!

9) HTML Input Radio Button

```
<script runat="server">
Sub submit(Source As Object, e As EventArgs)
if r1.Checked=True then
    p1.InnerHtml="Your favorite color is red"
else
    if r2.Checked=True then
        p1.InnerHtml="Your favorite color is green"
    else
        if r3.Checked=True then
            p1.InnerHtml="Your favorite color is blue"
        end if
    end if
end if
End Sub
</script>
```

```
<html>
<body>
```

```
<form runat="server">
<p>Select your favorite color:
<br />
<input id="r1" name="col" type="radio" runat="server">Red</input>
<br />
<input id="r2" name="col" type="radio" runat="server">Green</input>
<br />
<input id="r3" name="col" type="radio" runat="server">Blue</input>
```

<http://www.w3schools.com/aspnet/default.asp>

Page 93 of 147

```

<br />
<input type="button" value="Submit" OnServerClick="submit"
runat="server"/>
<p id="p1" runat="server" />
</form>

</body>
</html>

```

Output

Output Result:

Select your favorite color:

☒ Red
☐ Green
☐ Blue

Your favorite color is red

10) HTML Table

```

<script runat="server">
Sub submit(sender As Object, e As EventArgs)
Dim row,numrows,numcells,j,i
row=0
numrows=cint(rows1.Value)
numcells=cint(cells1.Value)
for j=1 to numrows
    Dim r As New HtmlTableRow()
    row=row+1
    for i=1 to numcells
        Dim c As New HtmlTableCell()
        c.Controls.Add(New LiteralControl("row " & j & ", cell " & i))
        r.Cells.Add(c)
    next
    t1.Rows.Add(r)
    t1.Visible=true
next

```

<http://www.w3schools.com/aspnet/default.asp>

End Sub
</script>

```
<html>
<body>

<form runat="server">
<p>Table rows:
<select id="rows1" runat="server">
  <option value="1">1</option>
  <option value="2">2</option>
  <option value="3">3</option>
</select>
<br />Table cells:
<select id="cells1" runat="server">
  <option value="1">1</option>
  <option value="2">2</option>
  <option value="3">3</option>
</select>
<br /><br />
<input type="submit" value="Display Table" runat="server"
OnServerClick="submit">
</p>
<table id="t1" border="1" runat="server" visible="false"/>
</form>

</body>
</html>
```

Output

Output Result:

Table rows: 2 ▼
 Table cells: 3 ▼

Display Table

row 1, cell 1	row 1, cell 2	row 1, cell 3
row 2, cell 1	row 2, cell 2	row 2, cell 3

11) HTML Table 2

```
<script runat="server">
Sub submit(sender As Object, e As EventArgs)
dim i,j
table1.BGColor="yellow"
table1.BorderColor="red"
for i=0 To table1.Rows.Count-1
    for j=0 To table1.Rows(i).Cells.Count-1
        table1.Rows(i).Cells(j).InnerHtml="Row " & i
    next
next
End Sub
</script>
```

```
<html>
<body>
```

```
<form runat="server">
<table id="table1" border="1" runat="server">
    <tr>
        <td>Cell 1</td>
        <td>Cell 2</td>
    </tr>
    <tr>
        <td>Cell 3</td>
        <td>Cell 4</td>
    </tr>
</table>
```



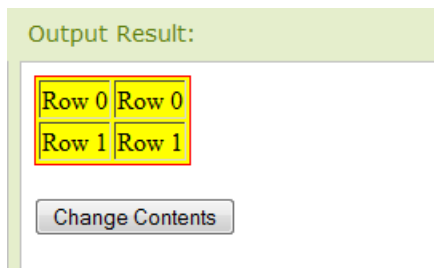
```

<br />
<input type="button" value="Change Contents" OnServerClick="submit"
runat="server"/>
</form>

</body>
</html>

```

Output



12) HTML Text Area

```

<script runat="server">
Sub submit(sender As Object, e As EventArgs)
    p1.InnerHtml = "<b>You wrote:</b> " & textarea1.Value
End Sub
</script>

<html>
<body>

<form runat="server">
Enter some text:<br />
<textarea id="textarea1" cols="35" rows="6" runat="server" />
<input type="submit" value="Submit" OnServerClick="submit"
runat="server" />
<p id="p1" runat="server" />
</form>

</body>
</html>

```

Output

Output Result:

Enter some text:

welcome to my page

Submit

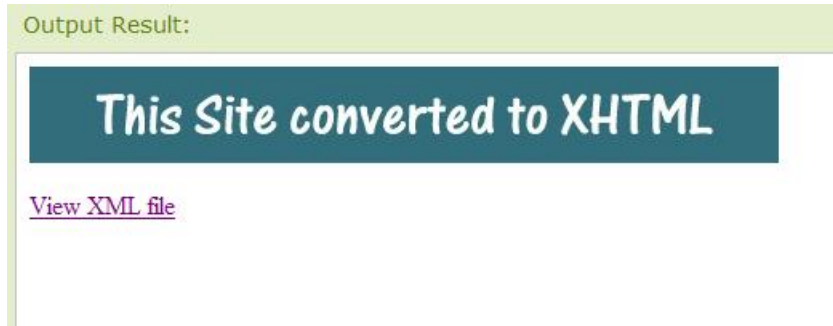
You wrote: welcome to my page

ASP .Net Web Controls

1) Adrotator

```
<script runat="server">  
  Sub change_url(sender As Object, e As AdCreatedEventArgs)  
    e.NavigateUrl="http://www.w3schools.com"  
  End Sub  
</script>  
  
<html>  
<body>  
  
  <form runat="server">  
    <asp:AdRotator AdvertisementFile="Ad1.xml"  
      runat="server" OnAdCreated="change_url"  
      target="_blank" />  
  </form>  
  
  <p><a href="ad1.xml" target="_blank">View XML file</a></p>  
  
</body>  
</html>
```

Output



2) Button

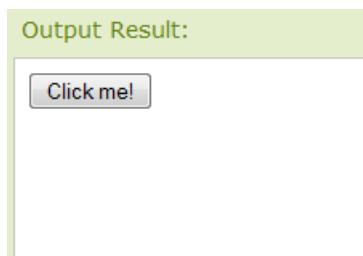
```
<script runat="server">
Sub submit(Source As Object, e As EventArgs)
    button1.Text="You clicked me!"
End Sub
</script>

<html>
<body>

<form runat="server">
<asp:Button id="button1" Text="Click me!" runat="server"
OnClick="submit" />
</form>

</body>
</html>
```

Output



3) Button 2

```
<script runat="server">
Sub submit(Source As Object, e As EventArgs)
```

<http://www.w3schools.com/aspnet/default.asp>

```
button1.Style("background-color")="#0000ff"
button1.Style("color")="#ffffff"
button1.Style("width")="200px"
button1.Style("cursor")="pointer"
button1.Style("font-family")="verdana"
button1.Style("font-weight")="bold"
button1.Style("font-size")="14pt"
button1.Text="You clicked me!"
End Sub
</script>

<html>
<body>

<form runat="server">
<asp:Button id="button1" Text="Click me!" runat="server"
OnClick="submit" />
</form>

</body>
</html>
```

Output

Output Result:



4) Calender

```
<html>
<body>

<form runat="server">
<asp:Calendar runat="server" />
</form>
```

```
</body>
</html>
```

Output

Output Result:

July 2009						
<						>
Sun	Mon	Tue	Wed	Thu	Fri	Sat
<u>28</u>	<u>29</u>	<u>30</u>	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>
<u>5</u>	<u>6</u>	<u>7</u>	<u>8</u>	<u>9</u>	<u>10</u>	<u>11</u>
<u>12</u>	<u>13</u>	<u>14</u>	<u>15</u>	<u>16</u>	<u>17</u>	<u>18</u>
<u>19</u>	<u>20</u>	<u>21</u>	<u>22</u>	<u>23</u>	<u>24</u>	<u>25</u>
<u>26</u>	<u>27</u>	<u>28</u>	<u>29</u>	<u>30</u>	<u>31</u>	<u>1</u>
<u>2</u>	<u>3</u>	<u>4</u>	<u>5</u>	<u>6</u>	<u>7</u>	<u>8</u>

5) Calendar 2

```
<html>
<body>
```

```
<form runat="server">
  <asp:Calendar DayNameFormat="Full" runat="server">
    <WeekendDayStyle BackColor="#fafad2" ForeColor="#ff0000"
  />
    <DayHeaderStyle ForeColor="#0000ff" />
    <TodayDayStyle BackColor="#00ff00" />
  </asp:Calendar>
</form>
```

```
</body>
</html>
```

Output

Output Result:

< July 2009 >						
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
<u>28</u>	<u>29</u>	<u>30</u>	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>
<u>5</u>	<u>6</u>	<u>7</u>	<u>8</u>	<u>9</u>	<u>10</u>	<u>11</u>
<u>12</u>	<u>13</u>	<u>14</u>	<u>15</u>	<u>16</u>	<u>17</u>	<u>18</u>
<u>19</u>	<u>20</u>	<u>21</u>	<u>22</u>	<u>23</u>	<u>24</u>	<u>25</u>
<u>26</u>	<u>27</u>	<u>28</u>	<u>29</u>	<u>30</u>	<u>31</u>	<u>1</u>
<u>2</u>	<u>3</u>	<u>4</u>	<u>5</u>	<u>6</u>	<u>7</u>	<u>8</u>

6) Calendar 3

<html>

<body>

<form runat="server">

<asp:Calendar DayNameFormat="Full" runat="server"

SelectionMode="DayWeekMonth"

SelectMonthText="<*>"

SelectWeekText="<->"/>

<SelectorStyle BackColor="#f5f5f5" />

</asp:Calendar>

</form>

</body>

</html>

Output

Output Result:

<=>	July 2009							>=
<=>	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	>=
<=>	28	29	30	1	2	3	4	
<=>	5	6	7	8	9	10	11	
<=>	12	13	14	15	16	17	18	
<=>	19	20	21	22	23	24	25	
<=>	26	27	28	29	30	31	1	
<=>	2	3	4	5	6	7	8	

7) Checkbox

```

<script runat="server">
  Sub Check(sender As Object, e As EventArgs)
    if check1.Checked then
      work.Text=home.Text
    else
      work.Text=""
    end if
  End Sub
</script>

```

```

<html>

```

```

<body>

```

```

<form runat="server">

```

```

  <p>Home Phone:

```

```

  <asp:TextBox id="home" runat="server" />

```

```

  <br />

```

```

  Work Phone:

```

```

  <asp:TextBox id="work" runat="server" />

```

```

  <asp:CheckBox id="check1"

```

```

    Text="Same as home phone" TextAlign="Right"

```

```

    AutoPostBack="True" OnCheckedChanged="Check"

```

```

    runat="server" />

```

```

  </p>

```

```

</form>

```

```
</body>
</html>
```

Output

Output Result:

Home Phone:	<input type="text" value="04142- 00000"/>	
Work Phone:	<input type="text" value="04142- 00000"/>	<input checked="" type="checkbox"/> Same as home phone

8) Checkboxlist

```
<script runat="server">
Sub Check(sender As Object, e As EventArgs)
    dim i
    mess.Text="<p>Selected Item(s):</p>"
    for i=0 to check1.Items.Count-1
        if check1.Items(i).Selected then
            mess.Text+=check1.Items(i).Text + "<br />"
        end if
    next
End Sub
</script>

<html>
<body>

<form runat="server">
<asp:CheckBoxList id="check1" AutoPostBack="True"
TextAlign="Right" OnSelectedIndexChanged="Check"
runat="server">
<asp:ListItem>Item 1</asp:ListItem>
<asp:ListItem>Item 2</asp:ListItem>
<asp:ListItem>Item 3</asp:ListItem>
<asp:ListItem>Item 4</asp:ListItem>
<asp:ListItem>Item 5</asp:ListItem>
```



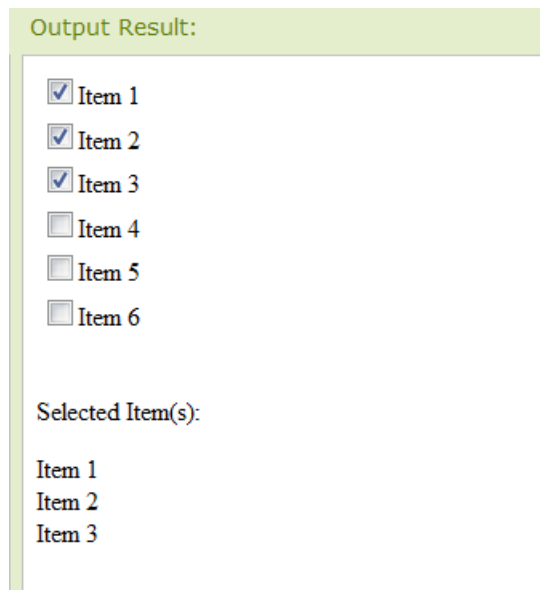
```

<asp:ListItem>Item 6</asp:ListItem>
</asp:CheckBoxList>
<br />
<asp:label id="mess" runat="server"/>
</form>

</body>
</html>

```

Output



9) Datalist

```

<%@ Import Namespace="System.Data" %>

<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
  dim mycdcatalog=New DataSet
  mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
  cdcatalog.DataSource=mycdcatalog
  cdcatalog.DataBind()
end if
end sub
</script>

```

```
<html>
<body>

<form runat="server">
<asp:DataList
id="cdcatalog"
gridlines="Both"
runat="server">

<HeaderTemplate>
My CD Catalog
</HeaderTemplate>

<ItemTemplate>
"<%#Container.DataItem("title")%>" of
<%#Container.DataItem("artist")%> -
$<%#Container.DataItem("price")%>
</ItemTemplate>

<FooterTemplate>
© Hege Refsnes
</FooterTemplate>

</asp:DataList>
</form>

</body>
</html>
```

Output

Output Result:

My CD Catalog
"Empire Burlesque" of Bob Dylan - \$10.90
"Hide your heart" of Bonnie Tyler - \$9.90
"Greatest Hits" of Dolly Parton - \$9.90
"Still got the blues" of Gary Moore - \$10.20
"Eros" of Eros Ramazzotti - \$9.90
© Hege Refsnes

10) Datalist With Styles

```
<%@ Import Namespace="System.Data" %>
```

```
<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycdcatalog=New DataSet
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
    cdcatalog.DataSource=mycdcatalog
    cdcatalog.DataBind()
end if
end sub
</script>
```

```
<html>
<body>
```

```
<form runat="server">
<asp:DataList id="cdcatalog"
runat="server"
cellpadding="2"
cellspacing="2"
borderstyle="inset"
backcolor="#e8e8e8"
width="100%"
headerstyle-font-name="Verdana"
headerstyle-font-size="12pt"
headerstyle-horizontalalign="center"
```

```

headerstyle-font-bold="True"
itemstyle-backcolor="#778899"
itemstyle-forecolor="#ffffff"
footerstyle-font-size="9pt"
footerstyle-font-italic="True">

<HeaderTemplate>
My CD Catalog
</HeaderTemplate>

<ItemTemplate>
"<%#Container.DataItem("title")%>" of
<%#Container.DataItem("artist")%> -
$<%#Container.DataItem("price")%>
</ItemTemplate>

<FooterTemplate>
© Hege Refsnes
</FooterTemplate>

</asp:DataList>
</form>

</body>
</html>

```

Output

Output Result:

My CD Catalog
"Empire Burlesque" of Bob Dylan - \$10.90
"Hide your heart" of Bonnie Tyler - \$9.90
"Greatest Hits" of Dolly Parton - \$9.90
"Still got the blues" of Gary Moore - \$10.20
"Eros" of Eros Ramazzotti - \$9.90
© Hege Refsnes

11) Datalist with alternating item Templates

```
<%@ Import Namespace="System.Data" %>
```

```
<script runat="server">  
sub Page_Load  
if Not Page.IsPostBack then  
    dim mycdcatalog=New DataSet  
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))  
    cdcatalog.DataSource=mycdcatalog  
    cdcatalog.DataBind()  
end if  
end sub  
</script>
```

```
<html>  
<body>
```

```
<form runat="server">  
<asp:DataList id="cdcatalog"  
runat="server"  
cellpadding="2"  
cellspacing="2"  
borderstyle="inset"  
backcolor="#e8e8e8"  
width="100%"  
headerstyle-font-name="Verdana"  
headerstyle-font-size="12pt"  
headerstyle-horizontalalign="center"  
headerstyle-font-bold="True"  
itemstyle-backcolor="#778899"  
itemstyle-forecolor="#ffffff"  
alternatingitemstyle-backcolor="#e8e8e8"  
alternatingitemstyle-forecolor="#000000"  
footerstyle-font-size="9pt"  
footerstyle-font-italic="True">
```

```
<HeaderTemplate>
```

```

My CD Catalog
</HeaderTemplate>

<ItemTemplate>
"<%#Container.DataItem("title")%>" of
<%#Container.DataItem("artist")%> -
$<%#Container.DataItem("price")%>
</ItemTemplate>

<AlternatingItemTemplate>
"<%#Container.DataItem("title")%>" of
<%#Container.DataItem("artist")%> -
$<%#Container.DataItem("price")%>
</AlternatingItemTemplate>

<FooterTemplate>
© Hege Refsnes
</FooterTemplate>

</asp:DataList>
</form>

</body>
</html>

```

Output

Output Result:

My CD Catalog	
"Empire Burlesque" of Bob Dylan - \$10.90	
"Hide your heart" of Bonnie Tyler - \$9.90	
"Greatest Hits" of Dolly Parton - \$9.90	
"Still got the blues" of Gary Moore - \$10.20	
"Eros" of Eros Ramazzotti - \$9.90	
© Hege Refsnes	

12) Dropdown List

```

<script runat="server">
Sub submit(sender As Object, e As EventArgs)
    mess.Text="You selected " & drop1.SelectedItem.Text
End Sub
</script>

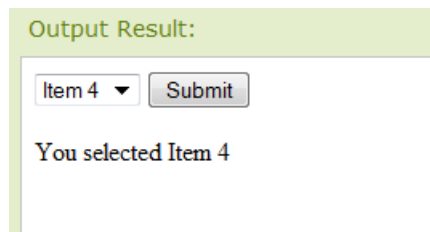
<html>
<body>

<form runat="server">
<asp:DropDownList id="drop1" runat="server">
<asp:ListItem>Item 1</asp:ListItem>
<asp:ListItem>Item 2</asp:ListItem>
<asp:ListItem>Item 3</asp:ListItem>
<asp:ListItem>Item 4</asp:ListItem>
<asp:ListItem>Item 5</asp:ListItem>
<asp:ListItem>Item 6</asp:ListItem>
</asp:DropDownList>
<asp:Button Text="Submit" OnClick="submit" runat="server"/>
<p><asp:label id="mess" runat="server"/></p>
</form>

</body>
</html>

```

Output



13) Hyperlink

```

<html>
<body>

<form runat="server">

```

```
<asp:HyperLink  
ImageUrl="/banners/w6.gif"  
NavigateUrl="http://www.w3schools.com"  
Text="Visit W3Schools!"  
Target="_blank"  
runat="server" />  
</form>
```

```
</body>
```

```
</html>
```

Output

Output Result:

A rectangular banner with a light gray background and a thin black border. It contains the text "W3Schools.com" in a large, bold, red serif font. Below it, in a smaller, black, sans-serif font, is the tagline "the best things in life ARE free".

W3Schools.com
the best things in life ARE free

14) Image

```
<html>
```

```
<body>
```

```
<form runat="server">
```

```
<asp:Image
```

```
runat="server"
```

```
AlternateText="W3Schools"
```

```
ImageUrl="/banners/w6.gif"/>
```

```
</form>
```

```
</body>
```

```
</html>
```

Output

Output Result:



W3Schools.com
the best things in life ARE free

15) Imagebutton

```
<script runat="server">
Sub getCoordinates(sender As Object, e As ImageClickEventArgs)
    mess.Text="Coordinates: " & e.x & ", " & e.y
End Sub
</script>
```

```
<html>
<body>
```

```
<form runat="server">
<p>Click on the image:</p>
<asp:ImageButton
runat="server"
ImageUrl="smiley.gif"
OnClick="getCoordinates"/>
<p><asp:label id="mess" runat="server"/></p>
</form>
```

```
</body>
</html>
```

Output

Output Result:

Click on the image:



Coordinates: 18, 13

16) Label

```

<script runat="server">
Sub submit(Sender As Object, e As EventArgs)
    label1.Text=txt1.Text
End Sub
</script>

<html>
<body>

<form runat="server">
Write some text:
<asp:TextBox id="txt1" Width="200" runat="server" />
<asp:Button id="b1" Text="Copy to Label" OnClick="submit"
runat="server" />
<p><asp:Label id="label1" runat="server" /></p>
</form>

</body>
</html>

```

Output

Output Result:

Write some text:

karan

17) LinkButton

```
<script runat="server">
Sub lblClick(sender As Object, e As EventArgs)
    Label1.Text="You clicked the LinkButton control"
End Sub
</script>

<html>
<body>

<form runat="server">
<asp:LinkButton Text="Click me!" OnClick="lblClick"
runat="server" />
<p><asp:Label id="Label1" runat="server" /></p>
</form>

</body>
</html>
```

Output

Output Result:

[Click me!](#)

You clicked the LinkButton control

18) Listbox

```
<script runat="server">
Sub submit(Sender As Object,e As EventArgs)
mess.Text="You selected " & drop1.SelectedItem.Text
End Sub
</script>

<html>
<body>
```

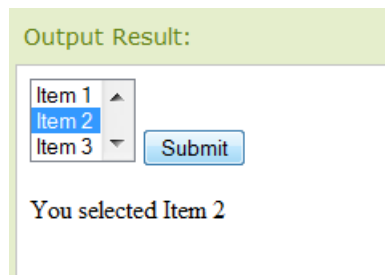
```

<form runat="server">
<asp:ListBox id="drop1" rows="3" runat="server">
<asp:ListItem selected="true">Item 1</asp:ListItem>
<asp:ListItem>Item 2</asp:ListItem>
<asp:ListItem>Item 3</asp:ListItem>
<asp:ListItem>Item 4</asp:ListItem>
<asp:ListItem>Item 5</asp:ListItem>
<asp:ListItem>Item 6</asp:ListItem>
</asp:ListBox>
<asp:Button Text="Submit" OnClick="submit" runat="server" />
<p><asp:label id="mess" runat="server" /></p>
</form>

</body>
</html>

```

Output



19) Literal

```

<html>
<body>

<form runat="server">
<asp:Literal Text="I love ASP. NET!" runat="server" />
</form>

</body>
</html>

```

Output

Output Result:

I love ASP. NET!

20) Literal 2

```

<script runat="server">
Sub submit(sender As Object, e As EventArgs)
    Literal1.Text="I love ASP.NET!"
End Sub
</script>

<html>
<body>

<form runat="server">
<asp:Literal id="Literal1" Text="I love ASP!" runat="server" />
<br /><br />
<asp:Button Text="Change Text" OnClick="submit" runat="server"
/>
</form>

</body>
</html>

```

Output

Output Result:

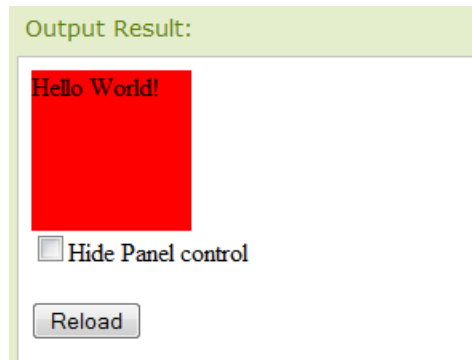
I love ASP.NET!

21) Panel

```
<script runat="server">  
Sub Page_Load(sender As Object, e As EventArgs)  
    if check1.Checked then  
        panel1.Visible=false  
    else  
        panel1.Visible=true  
    end if  
End Sub  
</script>
```

```
<html>  
<body>  
  
<form runat="server">  
<asp:Panel id="panel1"  
runat="server" BackColor="#ff0000"  
Height="100px" Width="100px">  
Hello World!  
</asp:Panel>  
<asp:CheckBox id="check1"  
Text="Hide Panel control"  
runat="server"/>  
<br /><br />  
<asp:Button Text="Reload" runat="server" />  
</form>  
  
</body>  
</html>
```

Output



22) RadioButton

```
<script runat="server">
Sub submit(Sender As Object, e As EventArgs)
if red.Checked then
    Label1.Text="You selected " & red.Text
elseif green.Checked then
    Label1.Text="You selected " & green.Text
elseif blue.Checked then
    Label1.Text="You selected " & blue.Text
end if
End Sub
</script>
```

```
<html>
<body>
```

```
<form runat="server">
Select your favorite color:
<br />
<asp:RadioButton id="red" Text="Red" Checked="True"
GroupName="colors" runat="server"/>
<br />
<asp:RadioButton id="green" Text="Green"
GroupName="colors" runat="server"/>
<br />
<asp:RadioButton id="blue" Text="Blue"
GroupName="colors" runat="server"/>
<br />
```

```

<asp:Button text="Submit" OnClick="submit" runat="server"/>
<p><asp:Label id="Label1" runat="server"/></p>
</form>

</body>
</html>

```

Output

Output Result:

Select your favorite color:

☐ Red

☒ Green

☐ Blue

You selected Green

23) RadioButton List

```

<script runat="server">
Sub submit(sender As Object, e As EventArgs)
    label1.Text="You selected " & radiolist1.SelectedItem.Text
End Sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="radiolist1" runat="server">
    <asp:ListItem selected="true">Item 1</asp:ListItem>
    <asp:ListItem>Item 2</asp:ListItem>
    <asp:ListItem>Item 3</asp:ListItem>
    <asp:ListItem>Item 4</asp:ListItem>
</asp:RadioButtonList>
<br />
<asp:Button text="Submit" OnClick="submit" runat="server"/>
<p><asp:Label id="Label1" runat="server"/></p>

```



```
</form>
```

```
</body>
```

```
</html>
```

Output

Output Result:

☐ Item 1

☐ Item 2

☐ Item 3

☒ Item 4

You selected Item 4

24) Repeater

```
<%@ Import Namespace="System.Data" %>
```

```
<script runat="server">
```

```
sub Page_Load
```

```
if Not Page.IsPostBack then
```

```
    dim mycdcatalog=New DataSet
```

```
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
```

```
    cdcatalog.DataSource=mycdcatalog
```

```
    cdcatalog.DataBind()
```

```
end if
```

```
end sub
```

```
</script>
```

```
<html>
```

```
<body>
```

```
<form runat="server">
```

```
<asp:Repeater id="cdcatalog" runat="server">
```

```
<HeaderTemplate>
```

```

<table border="1" width="100%">
<tr>
<th>Title</th>
<th>Artist</th>
<th>Company</th>
<th>Price</th>
</tr>
</HeaderTemplate>

<ItemTemplate>
<tr>
<td><%#Container.DataItem("title")%> </td>
<td><%#Container.DataItem("artist")%> </td>
<td><%#Container.DataItem("company")%> </td>
<td><%#Container.DataItem("price")%> </td>
</tr>
</ItemTemplate>

<FooterTemplate>
</table>
</FooterTemplate>

</asp:Repeater>
</form>

</html>
</body>

```

Output

Output Result:

Title	Artist	Company	Price
Empire Burlesque	Bob Dylan	Columbia	10.90
Hide your heart	Bonnie Tyler	CBS Records	9.90
Greatest Hits	Dolly Parton	RCA	9.90
Still got the blues	Gary Moore	Virgin records	10.20
Eros	Eros Ramazzotti	BMG	9.90

25) Repeater with Alternating Item Templates

```
<%@ Import Namespace="System.Data" %>
```

```
<script runat="server">  
sub Page_Load  
if Not Page.IsPostBack then  
    dim mycdcatalog=New DataSet  
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))  
    cdcatalog.DataSource=mycdcatalog  
    cdcatalog.DataBind()  
end if  
end sub  
</script>
```

```
<html>  
<body>
```

```
<form runat="server">  
<asp:Repeater id="cdcatalog" runat="server">
```

```
<HeaderTemplate>  
<table border="1" width="100%">  
<tr>  
<th>Title</th>  
<th>Artist</th>  
<th>Company</th>  
<th>Price</th>  
</tr>  
</HeaderTemplate>
```

```
<ItemTemplate>  
<tr>  
<td><%#Container.DataItem("title")%> </td>  
<td><%#Container.DataItem("artist")%> </td>
```

```

<td><%#Container.DataItem("company")%> </td>
<td><%#Container.DataItem("price")%> </td>
</tr>
</ItemTemplate>

<AlternatingItemTemplate>
<tr bgcolor="#e8e8e8">
<td><%#Container.DataItem("title")%> </td>
<td><%#Container.DataItem("artist")%> </td>
<td><%#Container.DataItem("company")%> </td>
<td><%#Container.DataItem("price")%> </td>
</tr>
</AlternatingItemTemplate>

<FooterTemplate>
</table>
</FooterTemplate>

</asp:Repeater>
</form>

</html>
</body>

```

Output

Output Result:

Title	Artist	Company	Price
Empire Burlesque	Bob Dylan	Columbia	10.90
Hide your heart	Bonnie Tyler	CBS Records	9.90
Greatest Hits	Dolly Parton	RCA	9.90
Still got the blues	Gary Moore	Virgin records	10.20
Eros	Eros Ramazzotti	BMG	9.90

26) Repeater with Separator Templates

```
<%@ Import Namespace="System.Data" %>
```

```
<script runat="server">
sub Page_Load
if Not Page.IsPostBack then
    dim mycdcatalog=New DataSet
    mycdcatalog.ReadXml(MapPath("cdcatalog.xml"))
    cdcatalog.DataSource=mycdcatalog
    cdcatalog.DataBind()
end if
end sub
</script>

<html>
<body>

<form runat="server">
<asp:Repeater id="cdcatalog" runat="server">

<HeaderTemplate>
<table border="0" width="100%">
<tr>
<th align="left">Title</th>
<th align="left">Artist</th>
<th align="left">Company</th>
<th align="left">Price</th>
</tr>
</HeaderTemplate>

<ItemTemplate>
<tr>
<td><%#Container.DataItem("title")%> </td>
<td><%#Container.DataItem("artist")%> </td>
<td><%#Container.DataItem("company")%> </td>
<td><%#Container.DataItem("price")%> </td>
</tr>
</ItemTemplate>

<SeparatorTemplate>
```

```

<tr>
<td colspan="6"><hr /></td>
</tr>
</SeparatorTemplate>

<FooterTemplate>
</table>
</FooterTemplate>

</asp:Repeater>
</form>

</html>
</body>

```

Output

Output Result:

Title	Artist	Company	Price
Empire Burlesque	Bob Dylan	Columbia	10.90
Hide your heart	Bonnie Tyler	CBS Records	9.90
Greatest Hits	Dolly Parton	RCA	9.90
Still got the blues	Gary Moore	Virgin records	10.20
Eros	Eros Ramazzotti	BMG	9.90

27) Table

```
<html>
<body>

<form runat=server>
<asp:Table runat="server" CellPadding="5"
GridLines="horizontal" HorizontalAlign="Center">
  <asp:TableRow>
    <asp:TableCell> 1 </asp:TableCell>
    <asp:TableCell> 2 </asp:TableCell>
  </asp:TableRow>
  <asp:TableRow>
    <asp:TableCell> 3 </asp:TableCell>
    <asp:TableCell> 4 </asp:TableCell>
  </asp:TableRow>
</asp:Table>
<br />
<asp:Table runat="server" CellPadding="5"
GridLines="vertical" HorizontalAlign="Center">
  <asp:TableRow>
    <asp:TableCell> 1 </asp:TableCell>
    <asp:TableCell> 2 </asp:TableCell>
  </asp:TableRow>
  <asp:TableRow>
    <asp:TableCell> 3 </asp:TableCell>
    <asp:TableCell> 4 </asp:TableCell>
  </asp:TableRow>
</asp:Table>
</form>

</body>
</html>
```

Output

Output Result:

1	2
3	4

1	2
3	4

28) Tabel 2

```

<script runat="server">
Sub Page_Load(sender As Object, e As EventArgs)
dim rows,cells,j,i
rows=3
cells=2
For j=0 To rows-1
    dim r As New TableRow()
    For i=0 To cells-1
        dim c As New TableCell()
        c.Controls.Add(New LiteralControl("row " & j & ", cell " & i))
        r.Cells.Add(c)
    Next
    Table1.Rows.Add(r)
Next
End Sub
</script>

```

```

<html>
<body>

```

```

<form runat="server">
<asp:Table id="Table1" BorderWidth="1" GridLines="Both"
runat="server" />
</form>

```

```

</body>
</html>

```


Output

Output Result:

row 0, cell 0	row 0, cell 1
row 1, cell 0	row 1, cell 1
row 2, cell 0	row 2, cell 1

29) Text Box

```
<script runat="server">
Sub submit(sender As Object, e As EventArgs)
    lbl1.Text="Your name is " & txt1.Text
End Sub
</script>
```

```
<html>
<body>
```

```
<form runat="server">
Enter your name:
<asp:TextBox id="txt1" runat="server" />
<asp:Button OnClick="submit" Text="Submit" runat="server" />
<p><asp:Label id="lbl1" runat="server" /></p>
</form>
```

```
</body>
</html>
```

Output

Output Result:

Enter your name:

Your name is fgdfgdfg

30) Textbox 2

```
<script runat="server">
sub submit(sender As Object, e As EventArgs)
lbl1.Text=txt1.Text
end sub
</script>

<html>
<body>

<form runat="server">
<asp:TextBox id="txt1" Text="Hello World!"
Font_Face="verdana" BackColor="#0000ff"
ForeColor="white" TextMode="MultiLine"
Height="50" runat="server" />
<asp:Button OnClick="submit"
Text="Copy Text to Label" runat="server" />
<p><asp:Label id="lbl1" runat="server" /></p>
</form>

</body>
</html>
```

Output

Output Result:

31) Textbox 3

```

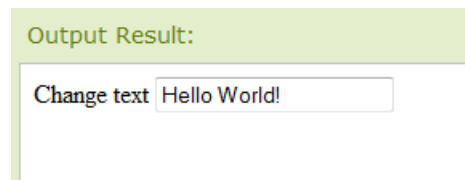
<script runat="server">
Sub change(sender As Object, e As EventArgs)
lbl1.Text="You changed text to " & txt1.Text
End Sub
</script>

<html>
<body>

<form runat="server">
Change text
<asp:TextBox id="txt1" runat="server"
text="Hello World!" ontextchanged="change"
autopostback="true"/>
<p><asp:Label id="lbl1" runat="server" /></p>
</form>

</body>
</html>

```

Output**32) XML**

```

<html>
<body>

<form runat="server">
<asp:Xml DocumentSource="cdcatalog.xml"
TransformSource="cdcatalog.xsl" runat="server" />
</form>

```

```
<p><a href="cdcatalog.xml" target="_blank">View XML
file</a></p>
```

```
<p><a href="cdcatalog.xsl" target="_blank">View XSL
file</a></p>
```

```
</body>
```

```
</html>
```

Output

Output Result:

My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan
Hide your heart	Bonnie Tyler
Greatest Hits	Dolly Parton
Still got the blues	Gary Moore
Eros	Eros Ramazzotti

[View XML file](#)

[View XSL file](#)

ASP .NET Validation Controls

1) Compare Validator

```
<html>
```

```
<body>
```

```
<form runat="server">
```

```
<table border="0" bgcolor="#b0c4de">
```

```
<tr valign="top">
```

```
<td colspan="4"><h4>Compare two values</h4></td>
```

```
</tr>
```

```
<tr valign="top">
```

```
<td><asp:TextBox id="txt1" runat="server" /></td>
```

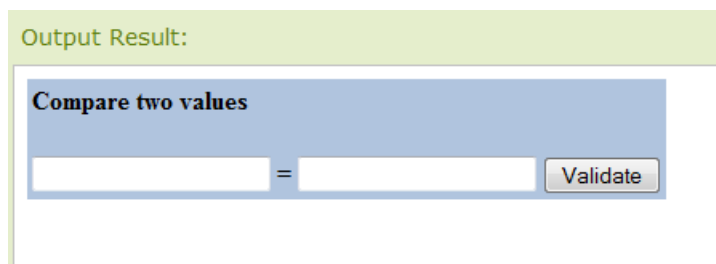
```

        <td> = </td>
        <td><asp:TextBox id="txt2" runat="server" /></td>
        <td><asp:Button Text="Validate" runat="server" /></td>
    </tr>
</table>
<br />
<asp:CompareValidator
id="compval"
Display="dynamic"
ControlToValidate="txt1"
ControlToCompare="txt2"
ForeColor="red"
BackColor="yellow"
Type="String"
EnableClientScript="false"
Text="Validation Failed!"
runat="server" />
</form>

</body>
</html>

```

Output



2) CompareValidator

```

<script runat="server">
sub check_operator(sender As Object, e As EventArgs)

compval.Operator=CType(list.SelectedIndex,ValidationCompareOperator)
compval.Validate()

```

```

end sub
</script>

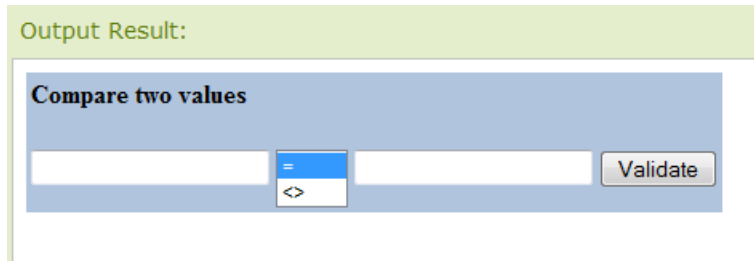
<html>
<body>

<form runat="server">
<table border="0" bgcolor="#b0c4de">
  <tr valign="top">
    <td colspan="4"><h4>Compare two values</h4></td>
  </tr>
  <tr valign="top">
    <td><asp:TextBox id="txt1" runat="server" /></td>
    <td>
      <asp:ListBox id="list" rows="2"
OnSelectedIndexChanged="check_operator" runat="server">
        <asp:ListItem value="Equal" selected>=</asp:ListItem>
        <asp:ListItem value="NotEqual"><></asp:ListItem>
      </asp:ListBox>
    </td>
    <td><asp:TextBox id="txt2" runat="server" /></td>
    <td><asp:Button Text="Validate" runat="server" /></td>
  </tr>
</table>
<br />
<asp:CompareValidator
id="compval"
Display="dynamic"
ControlToValidate="txt1"
ControlToCompare="txt2"
ForeColor="red"
BackColor="yellow"
Type="String"
EnableClientScript="false"
Text="Validation Failed!"
runat="server" />
</form>

```

```
</body>
</html>
```

Output



3) Custom Validator

```
<script runat="server">
Sub user(source As object,args As ServerValidateEventArgs)
  if len(args.Value)<8 or len(args.Value)>16 then
    args.IsValid=false
  else
    args.IsValid=true
  end if
End Sub
</script>
```

```
<html>
<body>
```

```
<form runat="server">
<asp:Label runat="server" Text="Enter a username: " />
<asp:TextBox id="txt1" runat="server" />
<asp:Button Text="Submit" runat="server"/>
<br />
<asp:Label id="mess" runat="server"/>
<br />
<asp:CustomValidator
ControlToValidate="txt1"
OnServerValidate="user"
Text="A username must be between 8 and 16 characters!"
```

```

runat="server"/>
</form>

</body>
</html>

```

Output

Output Result:

Enter a username:

A username must be between 8 and 16 characters!

4) Range Validator

```

<html>
<body>

<form runat="server">
Enter a date between 2005-01-01 and 2005-12-31:
<br />
<asp:TextBox id="tbox1" runat="server" />
<br /><br />
<asp:Button Text="Submit" runat="server" />
<br /><br />
<asp:RangeValidator
ControlToValidate="tbox1"
MinimumValue="2005-01-01"
MaximumValue="2005-12-31"
Type="Date"
EnableClientScript="false"
Text="The date must be between 2005-01-01 and 2005-12-31!"

```



```
runat="server" />
</form>
```

```
</body>
</html>
```

Output

Output Result:

Enter a date between 2005-01-01 and 2005-12-31:

The date must be between 2005-01-01 and 2005-12-31!

5) Range Validator 2

```
<script runat="server">
Sub submit(sender As Object, e As EventArgs)
If Page.IsValid Then
    lbl1.Text="Page is valid."
Else
    lbl1.Text="Page is not valid!!"
End If
End Sub
</script>
```

```
<html>
<body>
```

```
<form runat="server">
Enter a number from 1 to 100:
<asp:TextBox id="tbx1" runat="server" />
<br /><br />
<asp:Button Text="Submit" OnClick="submit" runat="server" />
<br /><br />
<asp:Label id="lbl1" runat="server" />
```

```

<br />
<asp:RangeValidator
ControlToValidate="tbx1"
MinimumValue="1"
MaximumValue="100"
Type="Integer"
EnableClientScript="false"
Text="The value must be from 1 to 100!"
runat="server" />
</form>

</body>
</html>

```

Output

Output Result:

Enter a number from 1 to 100:

Page is valid.

6) Regular Expression Validator

```

<script runat="server">
sub submit(sender As Object, e As EventArgs)
if Page.IsValid then
    lbl.Text="The page is valid!"
else
    lbl.Text="The page is not valid!"
end if
end sub
</script>

<html>
<body>

<form runat="server">

```

Enter a US zip code:

```
<asp:TextBox id="txtbox1" runat="server" />
<br /><br />
<asp:Button text="Submit" OnClick="submit" runat="server" />
<br /><br />
<asp:Label id="lbl" runat="server" />
<br />
<asp:RegularExpressionValidator
ControlToValidate="txtbox1"
ValidationExpression="\d{5}"
EnableClientScript="false"
ErrorMessage="The zip code must be 5 numeric digits!"
runat="server" />
</form>

</body>
</html>
```

Output

Output Result:

Enter a US zip code:

7) Regular Field Validator

```
<html>
<body>

<form runat="server">
Name: <asp:TextBox id="name" runat="server" />
<br />
Age: <asp:TextBox id="age" runat="server" />
<br /><br />
<asp:Button runat="server" Text="Submit" />
<br /><br />
<asp:RequiredFieldValidator
```

```

ControlToValidate="name"
Text="The name field is required!"
runat="server" />
</form>

</body>
</html>

```

Output

Output Result:

Name:

Age:

8) Validation Summary

```

<html>
<body>

<form runat="server">
<table>
<tr>
<td>
<table bgcolor="#b0c4de" cellpadding="10">
<tr>
<td align="right">Name: </td>
<td><asp:TextBox id="txt_name" runat="server"/> </td>
<td>
<asp:RequiredFieldValidator
ControlToValidate="txt_name"
ErrorMessage="Name"
Text="*"
runat="server"/>
</td>
</tr>
<tr>

```

```

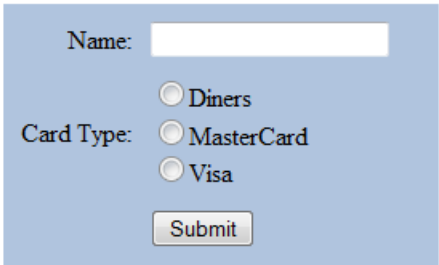
        <td align="right">Card Type: </td>
        <td>
        <asp:RadioButtonList id="rlist_type"
        RepeatLayout="Flow"
        runat="server">
        <asp:ListItem>Diners</asp:ListItem>
        <asp:ListItem>MasterCard</asp:ListItem>
        <asp:ListItem>Visa</asp:ListItem>
        </asp:RadioButtonList>
        </td>
        <td>
        <asp:RequiredFieldValidator
        ControlToValidate="rlist_type"
        ErrorMessage="Card Type"
        InitialValue=""
        Text="*"
        runat="server"/>
        </td>
    </tr>
    <tr>
        <td></td>
        <td><asp:Button id="b1" Text="Submit"
runat="server"/></td>
        <td></td>
    </tr>
</table>
</td>
</tr>
</table>
<br />
<asp:ValidationSummary
HeaderText="You must enter a value in the following fields:"
DisplayMode="BulletList"
EnableClientScript="true"
runat="server"/>
</form>

```

```
</body>
</html>
```

Output

Output Result:



9) Validation Summary 2

```
<html>
<body>

<form runat="server">
<table>
<tr>
<td>
<table bgcolor="#b0c4de" cellpadding="10">
<tr>
<td align="right">Name: </td>
<td><asp:TextBox id="txt_name" runat="server"/> </td>
<td>
<asp:RequiredFieldValidator
ControlToValidate="txt_name"
ErrorMessage="Name"
Text="*"
runat="server"/>
</td>
```

```

</tr>
<tr>
  <td align="right">Card Type:</td>
  <td>
    <asp:RadioButtonList id="rlist_type"
    RepeatLayout="Flow"
    runat="server">
      <asp:ListItem>Diners</asp:ListItem>
      <asp:ListItem>MasterCard</asp:ListItem>
      <asp:ListItem>Visa</asp:ListItem>
    </asp:RadioButtonList>
  </td>
  <td>
    <asp:RequiredFieldValidator
    ControlToValidate="rlist_type"
    ErrorMessage="Card Type"
    InitialValue=""
    Text="*"
    runat="server"/>
  </td>
</tr>
<tr>
  <td></td>
  <td><asp:Button id="b1" Text="Submit"
runat="server"/></td>
  <td></td>
</tr>
</table>
</td>
</tr>
</table>
<asp:ValidationSummary
ShowMessageBox="true"
ShowSummary="false"
HeaderText="You must enter a value in the following fields:"
EnableClientScript="true"
runat="server"/>

```

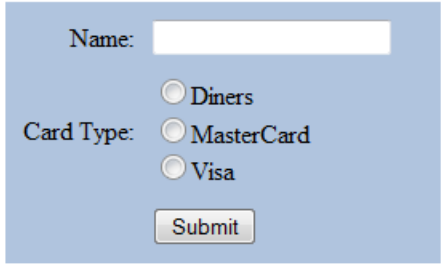
```
</form>
```

```
</body>
```

```
</html>
```

Output

Output Result:



ASP.NET Events

1) Page Load

```
<script runat="server">
```

```
Sub Page_Load
```

```
    lbl1.Text="The date and time is " & now()
```

```
End Sub
```

```
</script>
```

```
<html>
```

```
<body>
```

```
<form runat="server">
```

```
<h3><asp:label id="lbl1" runat="server" /></h3>
```

```
</form>
```

```
</body>
```

```
</html>
```

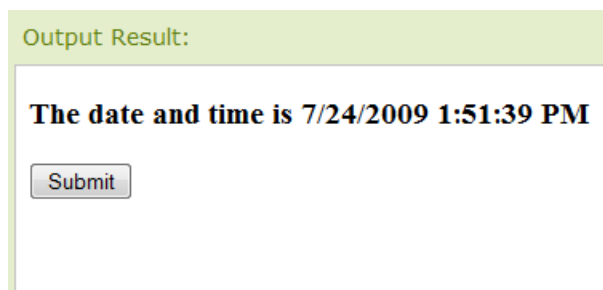
Output

The date and time is 7/24/2009 1:49:04 PM

2) Page Is Post back

```
<script runat="server">  
Sub Page_Load  
if Not Page.IsPostBack then  
    lbl1.Text="The date and time is " & now()  
end if  
End Sub  
  
Sub submit(s As Object, e As EventArgs)  
lbl2.Text="Hello World!"  
End Sub  
</script>  
  
<html>  
<body>  
<form runat="server">  
<h3><asp:label id="lbl1" runat="server" /></h3>  
<h3><asp:label id="lbl2" runat="server" /></h3>  
<asp:button text="Submit" onclick="submit" runat="server" />  
</form>  
</body>  
</html>
```

Output



ASP .NET Data Binding

1) Arraylist Radio Button List

```
<script runat="server">
Sub Page_Load
if Not Page.IsPostBack then
    dim mycountries=New ArrayList
    mycountries.Add("Norway")
    mycountries.Add("Sweden")
    mycountries.Add("France")
    mycountries.Add("Italy")
    mycountries.TrimToSize()
    mycountries.Sort()
    rb.DataSource=mycountries
    rb.DataBind()
end if
end sub

sub displayMessage(s as Object,e As EventArgs)
lbl1.text="Your favorite country is: " & rb.SelectedItem.Text
end sub
</script>

<html>
<body>

<form runat="server">
<asp:RadioButtonList id="rb" runat="server"
AutoPostBack="True"
onSelectedIndexChanged="displayMessage" />
<p><asp:label id="lbl1" runat="server" /></p>
</form>

</body>
</html>
```

Output

Output Result:

- ☐ France
- ☐ Italy
- ☐ Norway
- ☐ Sweden