

Html controls

1. HtmlInputText control
2. HtmlInputButton control
3. HtmlAnchor control
4. HtmlTextArea Control
5. HtmlInputCheckBox Control
6. HtmlInputRadioButton Control
7. HtmlSelect Control



System.Web.UI.HtmlControls

- The **System.Web.UI.HtmlControls** namespace contains classes that allow you to create HTML server controls on a Web Forms page.
- **HTML server controls run on the server and map directly to standard HTML tags supported by most browsers.**
- **This allows you to programmatically control the HTML elements on a Web Forms page.**



HtmlInputText control, an HtmlInputButton control, and an HtmlAnchor control

HtmlInputText control

```
<input id="Name" type="text" size="40" runat="server" />
```

HtmlInputButton control

```
<input type="submit" id="Enter" value="Enter"
runat="server" />
```

HtmlAnchor control

```
Click <a id="Anchor1" runat="server"
href="more.html">More </a>
```

HtmlTextArea Control

- Creates a server-side control that maps to the `<textarea>` HTML element and allows you *create a multiline text box*.
- `<textarea id="TextArea1" cols="20" name="S1" rows="2" onserverchange="onserverchangehandler" runat="server"></textarea>`



- Use the **HtmlTextArea** control to program against the HTML `<textarea>` element. This control allows you to create a multiline text box.
- The dimensions of the text box are controlled by the **Cols** and **Rows** properties.
- The **Cols** property determines the width of the control
- while the **Rows** property determines the height of the control
- The **HtmlTextArea** control contains a **ServerChange** event that is raised when the contents of the control change between posts to the server. The event is commonly used to validate the text entered in the control.

```
</head>
```

```
<body>
```

```
<h3>HtmlTextArea Example</h3>
```

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

```
What do you like best about ASP.NET?: <br>
```

```
<textarea id="TextArea1" cols=40 rows=4 runat=server  
>
```

```
<input type=submit value="Submit"
```

```
OnServerClick="SubmitBtn_Click" runat=server>
```

```
<p>
```

```
<span id="Span1" runat="server" />
```

```
</form>
```

```
</body>
```

```
</html>
```

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

```
<html>
```

```
<head>
```

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

```
void SubmitBtn_Click(Object sender, EventArgs e)
```

```
{
```

```
    Span1.InnerHtml = "You wrote: <br>" +  
TextArea1.Value;
```

```
}
```

```
</script>
```



HtmlTextArea Example

What do you like best about ASP.NET?:

Submit



HtmlTextArea Example

What do you like best about ASP.NET?:

hello everybody

Submit

You wrote:

hello everybody



HtmlSelect Class

- Allows programmatic access to the HTML `<select>` element on the server.
- **HtmlSelect** control create a **selection box**.
- Specify item listings in the control by placing HTML `<option>` elements between the opening and closing `<select>` tags.
- Each item is represented by a [System.Web.UI.WebControls.ListItem](#) object.
- To specify the text that is displayed for each item in the control, you can set the [ListItem.Text](#) property of the item, or simply place text between the opening and closing `<option>` tags.
- By the [ListItem.Value](#) property of the item, you can associate a value with the item that is different from the text.
- To select an item from the list by default, set the [ListItem.Selected](#) property of the item to **true**.

properties

- The **Size** property specifies the height (in rows) of the control.
- The **Multiple** property specifies whether more than one item can be concurrently selected in the **HtmlSelect** control.
- *By default*, the **HtmlSelect** control is displayed as a *drop-down list box*.
- If you *allow multiple selections* (by setting the **Multiple** property to **true**) or specify a height greater than one row (by setting the **Size** property to a value greater than **1**), *the control is displayed as a list box*.



HtmlSelect Sample

Select a color:

Don't see your color in the list above? You can add it here:

Click the button to apply a background color to this span.

Select a color:


```
<select>
  <option>SkyBlue</option>
  <option>LightGreen</option>
  <option>Gainsboro</option>
  <option>LemonChiffon</option>
</select>
<input type="button" runat="server"
  Value="Apply" OnServerClick="Apply_Click">
```

```
<p>
```

Don't see your color in the list above? You can add it here:


```
<input type="text" id="Text1" runat="server">
<input type="button" runat="server"
  Value="Add to List" OnServerClick="AddToList_Click">
```

```
<p>
```

```
<span id="Span1" runat="server">
```

Click the button to apply a background color to this span.

```
</span>
```

```
</form>
```

```
</body>
```

```
</html>
```

HtmlInputRadioButton Control

- Use the **HtmlInputRadioButton** control to create a radio button on a Web page.
- Creates a server-side control that maps to the HTML element and allows you to create a `<input type="radio">` radio button on a Web page.

```
<input type="radio" id="Radio1" name="Mode" runat="server"/>
```

Caption Text

```
<br>
```

- This is useful when you have multiple radio buttons and need to perform a calculation, based on the selection.
- **HtmlInputRadioButton** controls can be grouped together by specifying a common value for the Name property of each radio button you want to include in the group.
- To determine whether the **HtmlInputRadioButton** control is selected, test the Checked property.
- If you have a group of **HtmlInputRadioButton** controls, you must iterate through the controls and test the **Checked** property of each control individually.

HtmlInputRadioButton Sample

- Option 1
- Option 2
- Option 3

Radio 1 is checked



```
<input type="radio" id="Radio1" name="Mode"
  OnServerChange="Server_Change"
  runat="server" />
```

Option 1


```
<input type="radio" id="Radio2" name="Mode"
  OnServerChange="Server_Change"
  runat="server" />
```

Option 2


```
<input type="radio" id="Radio3" name="Mode"
  OnServerChange="Server_Change"
  runat="server" />
```

Option 3

```
<p> <span id=Span1 runat="server" /> <p>
```

```
<input type=submit id="Button1" value="Enter"
runat="server">
```



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

```
<html>
```

```
<head>
```

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

```
void Server_Change(object Source, EventArgs e)
```

```
{ if (Radio1.Checked == true)
```

```
    Span1.InnerHtml = "Radio1 is checked";
```

```
else if (Radio2.Checked == true)
```

```
    Span1.InnerHtml = "Radio2 is checked";
```

```
else if (Radio3.Checked == true)
```

```
    Span1.InnerHtml = "Radio3 is checked";
```

```
}
```

```
</script>
```

```
</head>
```



HtmlInputCheckBox control

- Use the **HtmlInputCheckBox** control to allow the user to select a **true** or **false** state.
- To determine whether the control is selected, use the Checked property.
- The **HtmlInputCheckBox** control provides a ServerChange event that is raised when the **Checked** property changes values between posts to the server.
- **The HtmlInputCheckBox control does not post back to the server when it is clicked.** You must provide another control on the Web page that supports posting to the server, such as an HtmlButton control, to send the state of the control back to the server.

HtmlInputCheckBox Sample

CheckBox1 Check1 is checked!

Enter




```
<%@ Page Language="C#" AutoEventWireup="True" %>
<html>
<head>
  <script runat="server">
    void Button1_Click(object Source, EventArgs e)
    { if (Check1.Checked == true)
      Span1.InnerHtml = "Check1 is checked!";
    else
      Span1.InnerHtml = "Check1 is not checked!";
    }
  </script>
</head>
```



HtmlInputImage Control

- the **HtmlInputImage** control to create a button that displays an image.
- You can programmatically control the action associated with the **HtmlInputImage** control by providing an event handler for the ServerClick event.
- The coordinates where the user clicks an **HtmlInputImage** control can be determined by using the ImageClickEventArgs.X and ImageClickEventArgs.Y properties of the System.Web.UI.ImageClickEventArgs object that is passed as a parameter to the user-defined event handler.

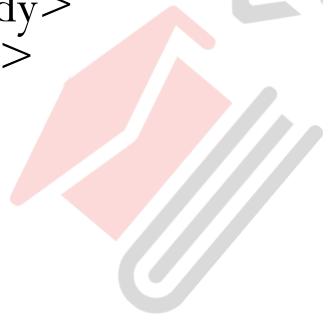
Script

```
<%@ Page Language="C#" %>
<html>
  <script runat="server">
    protected void SubmitBtn_Click(object sender,
    ImageClickEventArgs e)
    { Message.InnerHtml = "The Submit button was
    clicked.";
    }

    protected void ClearBtn_Click(object sender,
    ImageClickEventArgs e)
    { Message.InnerHtml = "The Clear button was
    clicked.";
    }
  </script>
```

Html code

```
<body>  
  <h3>HtmlInputImage Example</h3>  
  <form runat="server">  
    <input type="image" alt="Submit button"  
      src="~/images/1.jpg" OnServerClick="SubmitBtn_Click"  
      runat="server" id="Image1">  
    <input type="image" alt="Clear button"  
      src="~/images/2.jpg" OnServerClick="ClearBtn_Click"  
      runat="server id="Image2">  
    <h1>  
      <span id="Message" runat="server">  
    </span>  
    </h1>  
  </form>  
</body>  
</html>
```



Output

HtmlInputImage Example



The Submit button was clicked.



HtmlInputImage Control

- Creates a server-side control that maps to the **<input type=image>** HTML element and allows you to create an button that displays an image.

```
<input type=image  
id="programmaticID"  
src="imagepath"  
align="imagealignment"  
alt="alttext"  
OnServerClick="onserverclickhandler"  
width="borderwidthinpixels"  
runat="server" >
```