**Server Controls**

Controls are small building blocks of the graphical user interface, which include text boxes, buttons, check boxes, list boxes, labels, and numerous other tools. Using these tools, the users can enter data, make selections and indicate their preferences.

Controls are also used for structural jobs, like validation, data access, security, creating master pages, and data manipulation.

ASP.NET uses five types of web controls, which are:

* HTML controls
* HTML Server controls
* ASP.NET Server controls
* ASP.NET Ajax Server controls
* User controls and custom controls

ASP.NET server controls are the primary controls used in ASP.NET. These controls can be grouped into the following categories:

* **Validation controls** - These are used to validate user input and they work by running client-side script.
* **Data source controls** - These controls provides data binding to different data sources.
* **Data view controls** - These are various lists and tables, which can bind to data from data sources for displaying.
* **Personalization controls** - These are used for personalization of a page according to the user preferences, based on user information.
* **Login and security controls** - These controls provide user authentication.
* **Master pages** - These controls provide consistent layout and interface throughout the application.
* **Navigation controls** - These controls help in navigation. For example, menus, tree view etc.
* **Rich controls** - These controls implement special features. For example, AdRotator, FileUpload, and Calendar control.

The syntax for using server controls is:

<asp:controlType ID ="ControlID" runat="server" Property1=value1 [Property2=value2] />

In addition, visual studio has the following features, to help produce in error-free coding:

* Dragging and dropping of controls in design view
* IntelliSense feature that displays and auto-completes the properties
* The properties window to set the property values directly

Properties of the Server Controls

ASP.NET server controls with a visual aspect are derived from the WebControl class and inherit all the properties, events, and methods of this class.

The WebControl class itself and some other server controls that are not visually rendered are derived from the System.Web.UI.Control class. For example, PlaceHolder control or XML control.

ASP.Net server controls inherit all properties, events, and methods of the WebControl and System.Web.UI.Control class.

The following table shows the inherited properties, common to all server controls:

|  |  |
| --- | --- |
| **Property** | **Description** |
| AccessKey | Pressing this key with the Alt key moves focus to the control. |
| Attributes | It is the collection of arbitrary attributes (for rendering only) that do not correspond to properties on the control. |
| BackColor | Background color. |
| BindingContainer | The control that contains this control's data binding. |
| BorderColor | Border color. |
| BorderStyle | Border style. |
| BorderWidth | Border width. |
| CausesValidation | Indicates if it causes validation. |
| ChildControlCreated | It indicates whether the server control's child controls have been created. |
| ClientID | Control ID for HTML markup. |
| Context | The HttpContext object associated with the server control. |
| Controls | Collection of all controls contained within the control. |
| ControlStyle | The style of the Web server control. |
| CssClass | CSS class |
| DataItemContainer | Gets a reference to the naming container if the naming container implements IDataItemContainer. |
| DataKeysContainer | Gets a reference to the naming container if the naming container implements IDataKeysControl. |
| DesignMode | It indicates whether the control is being used on a design surface. |
| DisabledCssClass | Gets or sets the CSS class to apply to the rendered HTML element when the control is disabled. |
| Enabled | Indicates whether the control is grayed out. |
| EnableTheming | Indicates whether theming applies to the control. |
| EnableViewState | Indicates whether the view state of the control is maintained. |
| Events | Gets a list of event handler delegates for the control. |
| Font | Font. |
| Forecolor | Foreground color. |
| HasAttributes | Indicates whether the control has attributes set. |
| HasChildViewState | Indicates whether the current server control's child controls have any saved view-state settings. |
| Height | Height in pixels or %. |
| ID | Identifier for the control. |
| IsChildControlStateCleared | Indicates whether controls contained within this control have control state. |
| IsEnabled | Gets a value indicating whether the control is enabled. |
| IsTrackingViewState | It indicates whether the server control is saving changes to its view state. |
| IsViewStateEnabled | It indicates whether view state is enabled for this control. |
| LoadViewStateById | It indicates whether the control participates in loading its view state by ID instead of index. |
| Page | Page containing the control. |
| Parent | Parent control. |
| RenderingCompatibility | It specifies the ASP.NET version that the rendered HTML will be compatible with. |
| Site | The container that hosts the current control when rendered on a design surface. |
| SkinID | Gets or sets the skin to apply to the control. |
| Style | Gets a collection of text attributes that will be rendered as a style attribute on the outer tag of the Web server control. |
| TabIndex | Gets or sets the tab index of the Web server control. |
| TagKey | Gets the HtmlTextWriterTag value that corresponds to this Web server control. |
| TagName | Gets the name of the control tag. |
| TemplateControl | The template that contains this control. |
| TemplateSourceDirectory | Gets the virtual directory of the page or control containing this control. |
| ToolTip | Gets or sets the text displayed when the mouse pointer hovers over the web server control. |
| UniqueID | Unique identifier. |
| ViewState | Gets a dictionary of state information that saves and restores the view state of a server control across multiple requests for the same page. |
| ViewStateIgnoreCase | It indicates whether the StateBag object is case-insensitive. |
| ViewStateMode | Gets or sets the view-state mode of this control. |
| Visible | It indicates whether a server control is visible. |
| Width | Gets or sets the width of the Web server control. |

Methods of the Server Controls

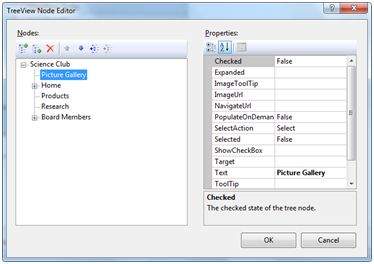
The following table provides the methods of the server controls:

|  |  |
| --- | --- |
| **Method** | **Description** |
| AddAttributesToRender | Adds HTML attributes and styles that need to be rendered to the specified HtmlTextWriterTag. |
| AddedControl | Called after a child control is added to the Controls collection of the control object. |
| AddParsedSubObject | Notifies the server control that an element, either XML or HTML, was parsed, and adds the element to the server control's control collection. |
| ApplyStyleSheetSkin | Applies the style properties defined in the page style sheet to the control. |
| ClearCachedClientID | Infrastructure. Sets the cached ClientID value to null. |
| ClearChildControlState | Deletes the control-state information for the server control's child controls. |
| ClearChildState | Deletes the view-state and control-state information for all the server control's child controls. |
| ClearChildViewState | Deletes the view-state information for all the server control's child controls. |
| CreateChildControls | Used in creating child controls. |
| CreateControlCollection | Creates a new ControlCollection object to hold the child controls. |
| CreateControlStyle | Creates the style object that is used to implement all style related properties. |
| DataBind | Binds a data source to the server control and all its child controls. |
| DataBind(Boolean) | Binds a data source to the server control and all its child controls with an option to raise the DataBinding event. |
| DataBindChildren | Binds a data source to the server control's child controls. |
| Dispose | Enables a server control to perform final clean up before it is released from memory. |
| EnsureChildControls | Determines whether the server control contains child controls. If it does not, it creates child controls. |
| EnsureID | Creates an identifier for controls that do not have an identifier. |
| Equals(Object) | Determines whether the specified object is equal to the current object. |
| Finalize | Allows an object to attempt to free resources and perform other cleanup operations before the object is reclaimed by garbage collection. |
| FindControl(String) | Searches the current naming container for a server control with the specified id parameter. |
| FindControl(String, Int32) | Searches the current naming container for a server control with the specified id and an integer. |
| Focus | Sets input focus to a control. |
| GetDesignModeState | Gets design-time data for a control. |
| GetType | Gets the type of the current instance. |
| GetUniqueIDRelativeTo | Returns the prefixed portion of the UniqueID property of the specified control. |
| HasControls | Determines if the server control contains any child controls. |
| HasEvents | Indicates whether events are registered for the control or any child controls. |
| IsLiteralContent | Determines if the server control holds only literal content. |
| LoadControlState | Restores control-state information. |
| LoadViewState | Restores view-state information. |
| MapPathSecure | Retrieves the physical path that a virtual path, either absolute or relative, maps to. |
| MemberwiseClone | Creates a shallow copy of the current object. |
| MergeStyle | Copies any nonblank elements of the specified style to the web control, but does not overwrite any existing style elements of the control. |
| OnBubbleEvent | Determines whether the event for the server control is passed up the page's UI server control hierarchy. |
| OnDataBinding | Raises the data binding event. |
| OnInit | Raises the Init event. |
| OnLoad | Raises the Load event. |
| OnPreRender | Raises the PreRender event. |
| OnUnload | Raises the Unload event. |
| OpenFile | Gets a Stream used to read a file. |
| RemovedControl | Called after a child control is removed from the controls collection of the control object. |
| Render | Renders the control to the specified HTML writer. |
| RenderBeginTag | Renders the HTML opening tag of the control to the specified writer. |
| RenderChildren | Outputs the contents of a server control's children to a provided HtmlTextWriter object, which writes the contents to be rendered on the client. |
| RenderContents | Renders the contents of the control to the specified writer. |
| RenderControl(HtmlTextWriter) | Outputs server control content to a provided HtmlTextWriter object and stores tracing information about the control if tracing is enabled. |
| RenderEndTag | Renders the HTML closing tag of the control into the specified writer. |
| ResolveAdapter | Gets the control adapter responsible for rendering the specified control. |
| SaveControlState | Saves any server control state changes that have occurred since the time the page was posted back to the server. |
| SaveViewState | Saves any state that was modified after the TrackViewState method was invoked. |
| SetDesignModeState | Sets design-time data for a control. |
| ToString | Returns a string that represents the current object. |
| TrackViewState | Causes the control to track changes to its view state so that they can be stored in the object's view state property. |

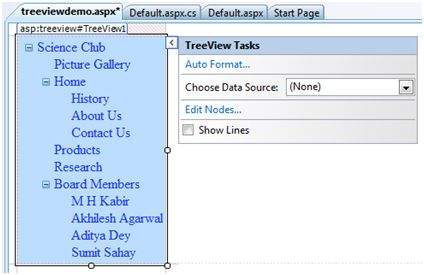
Example

Let us look at a particular server control - a tree view control. A Tree view control comes under navigation controls. Other Navigation controls are: Menu control and SiteMapPath control.

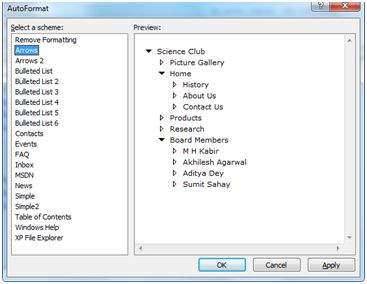
Add a tree view control on the page. Select Edit Nodes... from the tasks. Edit each of the nodes using the Tree view node editor as shown:



Once you have created the nodes, it looks like the following in design view:



The AutoFormat... task allows you to format the tree view as shown:



Add a label control and a text box control on the page and name them lblmessage and txtmessage respectively.

Write a few lines of code to ensure that when a particular node is selected, the label control displays the node text and the text box displays all child nodes under it, if any. The code behind the file should look like this:

using System;

using System.Collections;

using System.Configuration;

using System.Data;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Xml.Linq;

namespace eventdemo {

public partial class treeviewdemo : System.Web.UI.Page {

protected void Page\_Load(object sender, EventArgs e) {

txtmessage.Text = " ";

}

protected void TreeView1\_SelectedNodeChanged(object sender, EventArgs e) {

txtmessage.Text = " ";

lblmessage.Text = "Selected node changed to: " + TreeView1.SelectedNode.Text;

TreeNodeCollection childnodes = TreeView1.SelectedNode.ChildNodes;

if(childnodes != null) {

txtmessage.Text = " ";

foreach (TreeNode t in childnodes) {

txtmessage.Text += t.Value;

}

}

}

}

}

Execute the page to see the effects. You will be able to expand and collapse the nodes.

