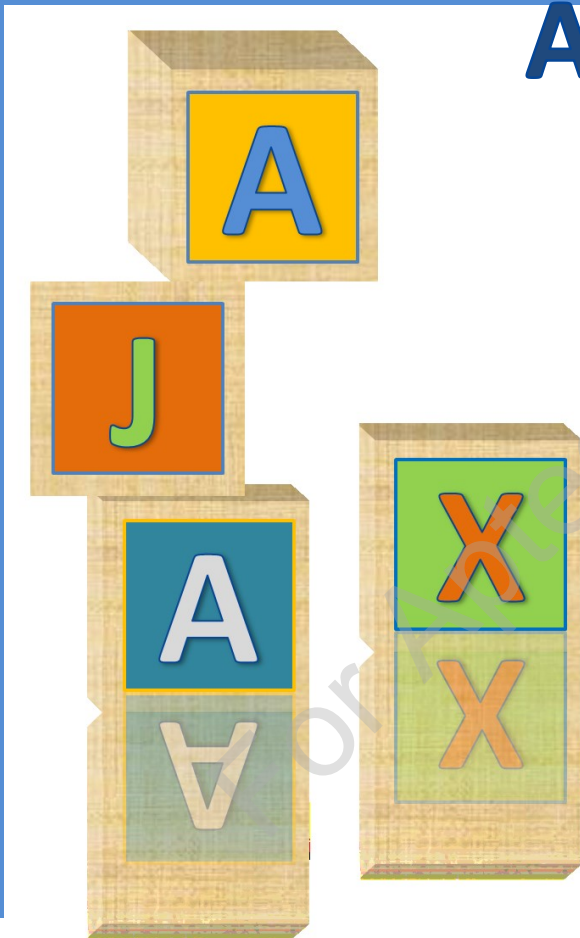


Programming ASP.NET AJAX

Session: 5

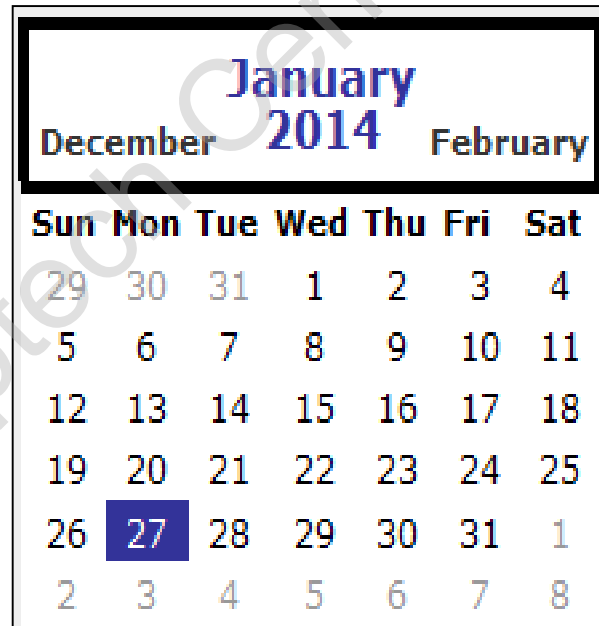
AJAX Control Toolkit



- ◆ Explain the Text Input Controls
- ◆ Describe the User Interface Controls
- ◆ Explain the AJAX Control Toolkit
- ◆ Explain Cascading Style Sheets

For Aptech Centre Use Only

- ◆ Developed by ASP.NET community under an open source domain.
- ◆ Is a library of server-side controls, which are called extender controls or control extenders, because they are an extension of existing ASP.NET server controls.
- ◆ Is needed because it:
 - ◆ Allows user to create ASP.NET AJAX extenders and controls
 - ◆ Provides AJAX functionality in Web sites
 - ◆ Allows creation of rich UI controls as shown in the figure



- ◆ The extender control associates the client component containing the new functionality with the server control.
- ◆ For example,
 - ◆ An `AutoCompleteExtender` extends the server control, `TextBox`.
 - ◆ Properties of the `AutoCompleteExtender` control will be set in order to associate it with the `TextBox`.
- ◆ Methods defined inside Web services are invoked by some extender controls as shown in the figure.

```
<%@ Register Assembly="AjaxControlToolkit"
    Namespace="AjaxControlToolkit"
    TagPrefix="ajaxToolkit"
%>
```

Image 1

```
<ajaxToolkit:AutoCompleteExtender
    runat="server"
    ID="autocomplexSearch"
    . . .
    . . .
/>
```

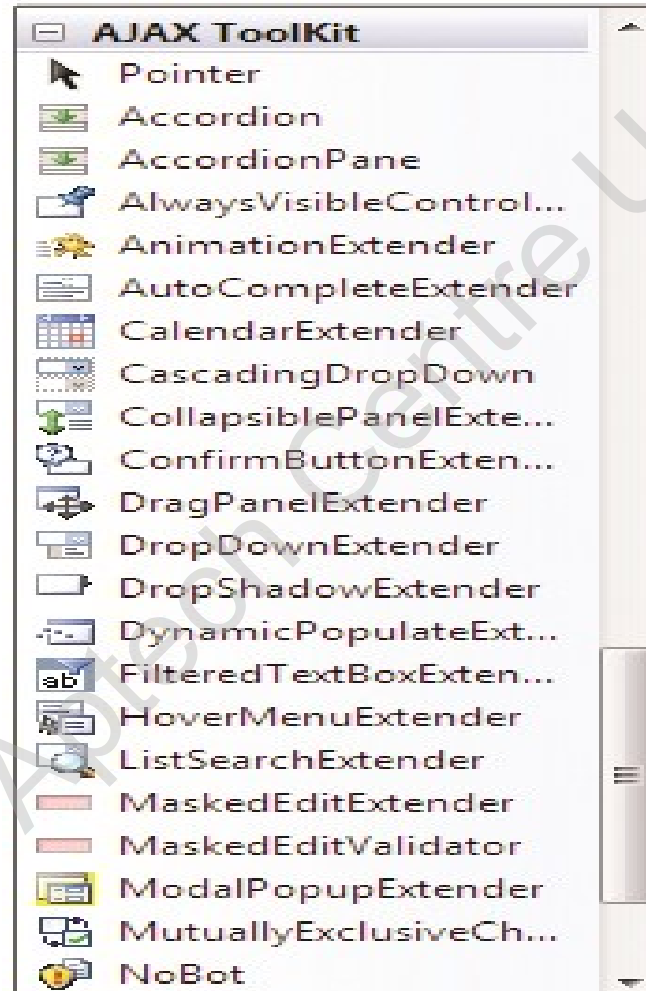
Image 2

- ◆ Toolkit controls are classified into two - Text Input and User Interface as shown in the table.

Category	Examples of Controls	Purpose
Text Input	AutoCompleteExtender, TextBoxWatermarkExtender, PasswordStrength, DynamicPopulateExtender , FilteredTextBox	These controls are used to extend along with Text Input controls such as TextBox, DropDownList and so forth.
User Interface	Accordion, CascadingDropDown, ConfirmButtonExtender, AlwaysVisibleControlExtender, Rating, and TabContainer ColorPicker, ComboBox, ConfirmButton, DragPanel, DropDown, DropShadow, DynamicPopulate, HoverMenu, HTMLEditor, ListSearch, MaskedEdit, ModalPopup, MultiHandleSlider	These controls enable you to create a better UI.

Category	Examples of Controls	Purpose
User Interface	MutuallyExclusiveCheckBox, NoBot, NumericUpDown, PagingBulletedList, PasswordStrength, Popup, Rating, ReorderList, Resizable, RoundedCorners, Seadragon, Slider, SlideShow, TabContainer, TabPanel, TextBoxWatermark, ToggleButton, UpdatePanelAnimation, ValidatorCallout	These controls enable you to create a better UI.

Following figure displays the AJAX Toolkit:



- ◆ Is a stylesheet language used for presenting elements written in markup languages like HTML.
- ◆ Help specify visual styles created using the controls in the AJAX Control Toolkit.
- ◆ Provides the following benefits:
 - ◆ It places formatting elements in a different file
 - ◆ It keeps the markup of the page simpler
 - ◆ It ensures better accessibility
- ◆ Can be used to perform tasks such as formatting the color, layout, and font of Web Pages.
- ◆ Has in-built support in Visual Studio 2013.

For Aptech Centre Use Only

- ◆ Consider an HTML file to which CSS needs to be added as shown in the following images:

- ◆ Image 1 shows part of the HTML file consisting of paragraph tags.
- ◆ To add CSS properties to the text enclosed between the `<p>...</p>` tags, you create CSS code as shown in Image 2. The code consists of three formatting elements or properties: `font-family`, `alignment`, and `color` with the values `Impact`, `center` and `Darkblue` respectively. Here, the entire code is called a rule and `p` is called a selector.
- ◆ To link the CSS class to the HTML file, in the `<head>...</head>` section, you will use the code as shown in Image 3. This code is combined with the code in Image 1 to form the HTML file.
- ◆ Finally, the output of the HTML file with CSS property applied is shown in Image 4.

```
<p>
I am trying css on my Web page
</p>
```

Image 1

```
p
{
    font-family: Impact;
    Text-align:  center;
    color      :  Darkblue;
}
```

Image 2

```
<head>
<link rel="stylesheet" type="text/css"
href="..\Test.css">
</head>
```

Image 3

I am trying css on my Web page

Image 4

- ◆ A class attribute in CSS is used to create unique styles with names.
- ◆ It is also used as a stylesheet selector to assign styles to a group of elements.
- ◆ Have various attributes of controls in the Toolkit related to them, which can be used to define styles as shown in the following images:

- ◆ Image 1 shows the specifying of the class name in the markup in the HTML file.

```
<HTML>
<link rel="stylesheet" type="text/css"
href="storystyle.css">
<body class="maintext">
There was once a wise king<br/>
who lived in the land of wise.
</body>
<br/><br/>
<p class="regulartext">Continued...</p>
</HTML>
```

Image 1

- ◆ Image 2 shows how the classes are defined as a rule in the CSS file.

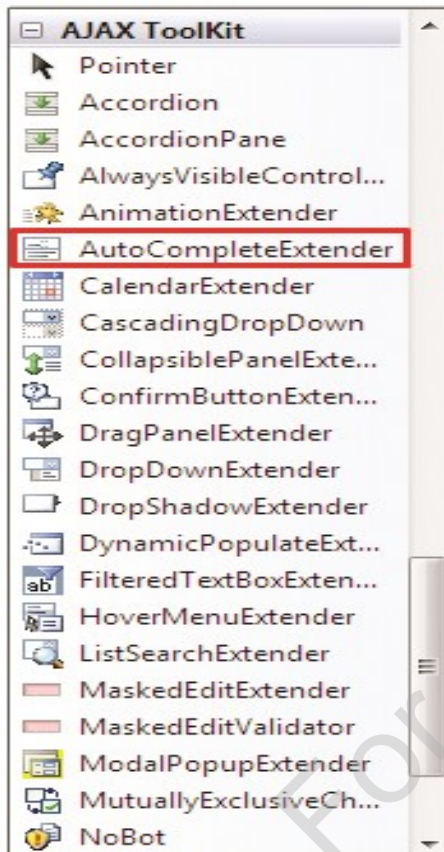
```
.maintext
{
font-style:italic;
font-family:Garamond;
font-size: 24;
}

.regulartext
{
font-style:normal;
font-family:Trebuchet MS;
font-size: 16;
}
```

Image 2

AutoCompleteExtender

- ◆ Enables word suggestion for a particular field, when used along with the `TextBox` control as shown in the figures.



AutoCompleteExtender

```
<cc1:AutoCompleteExtender
  ID="autocompexSearch"
  runat="server"
  MinimumPrefixLength="2"
  Enabled="True"
  ServicePath="~/AutocompleteService.asmx"
  ServiceMethod="Search"
  TargetControlID="txtSearch">
</cc1:AutoCompleteExtender>
```

Using Properties of AutoCompleteExtender

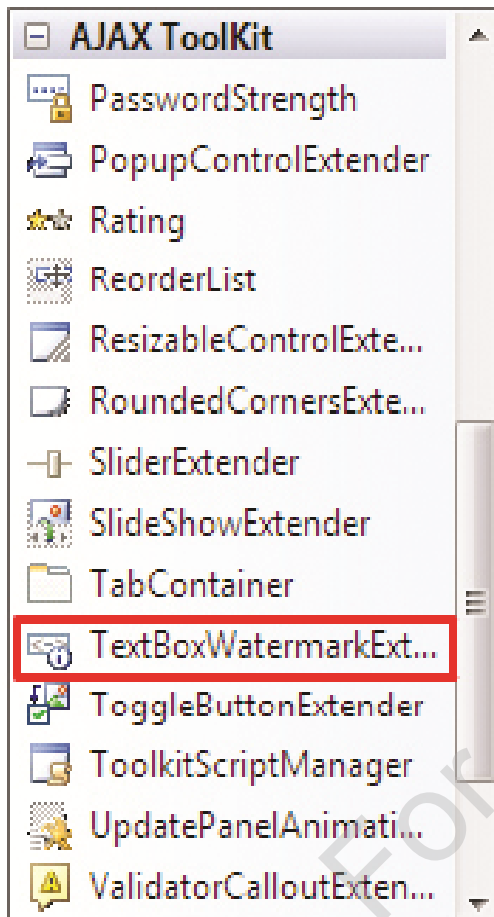
AutoCompleteExtender

- ◆ Following table lists the `AutoCompleteExtender` properties:

Property	Description
MinimumPrefixLength	Specifies the minimum number of characters required in the <code>TextBox</code> to get suggestions.
ServicePath	Specifies the path where the <code>AutoCompleteExtender</code> control will search for getting the word suggestions.
ServiceMethod	Specifies the method to be called for word suggestions.
TargetControlID	Specifies the control to which the <code>AutoCompleteExtender</code> is attached.

TextBoxWatermarkExtender

- ◆ Provides a `TextBox` with a watermark effect.

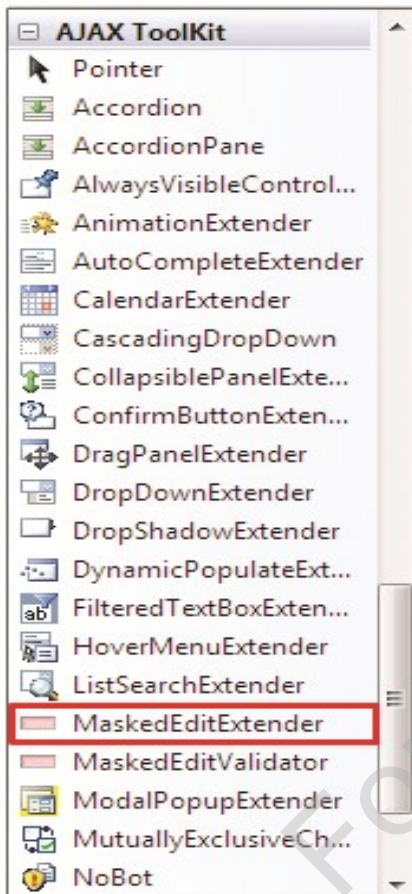


```
<cc1:TextBoxWatermarkExtender ID="txtwatmrkexName"
runat="server"
Enabled="True"
TargetControlID="txtName"
WatermarkText="Type Your Name Here"
WatermarkCssClass="watermarked">
</cc1:TextBoxWatermarkExtender>
```

- ◆ Following table lists the watermark properties:

Property	Description
WatermarkText	Enables the user to set the display text in the text box to prompt the user on what needs to be entered in the text box.
WatermarkCssClass	Enables you to associate a CSS class to the text box to provide styles to the watermarked text box.
TargetControlID	Specifies the ID of the control to which the watermark effect needs to be provided.

- ◆ A `MaskedEditValidator` control is used along with the `MaskedEditExtender` control to achieve desired validations. Following figures display the control and the code:



MaskedEditExtender

```
<cc1:MaskedEditExtender ID="mskedexNumber" runat="server"
    Enabled="True"
    Mask="9,999.99"
    MaskType="Number"
    InputDirection="LeftToRight"
    AcceptNegative="Left"
    ErrorTooltipEnabled="true"
    MessageValidatorTip="true"
    TargetControlID="txtNumber"/>
<cc1:MaskedEditValidator ID="mskvalNumber"
    runat="server"
    ControlExtender="mskedexNumber"
    ControlToValidate="txtNumber"
    IsValidEmpty="False"
    MaximumValue="1200"
    EmptyValueMessage="Number is required"
    TooltipMessage="Enter a valid number"
    Enabled="true"
    ErrorMessage="number not in proper format">
</cc1:MaskedEditValidator>
```

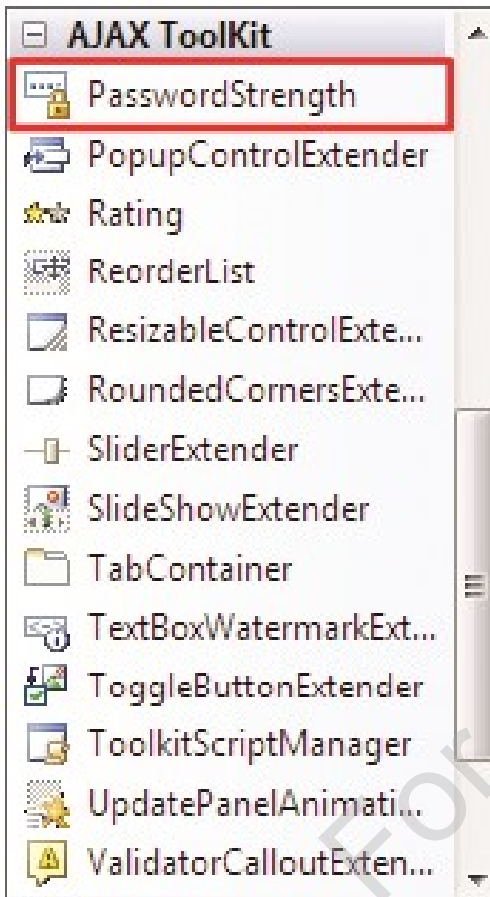
Code to Create a MaskedEditExtender Control

- ◆ Following table displays the text box properties:

Property	Description
Mask	Enables user to set the format of text to be inserted in the text box.
MaskType	Enables the type of text to be entered in the text box. This might be None, Number, Date, Time, or DateTime.
InputDirection	Specifies the way the text is inserted in the text box.
AcceptNegative	Specifies the direction of negative sign, which can be inserted in the text box.
ControlExtender	Used to attach a <code>MaskedEditExtender</code> to the <code>MaskedEditValidator</code> .
ControlToValidate	Specifies the control for which the validator is used.
MaximumValue	Specifies the maximum value that can be entered in the text box.

PasswordStrength

- ◆ Is attached to a text box, which accepts a password as its text as shown in the following figures:



PasswordStrength

```
<cc1:PasswordStrength ID="pwdexUserPwd" runat="server"
    Enabled="True"
    TargetControlID="txtPwd"
    DisplayPosition="RightSide"
    StrengthIndicatorType="Text"
    PreferredPasswordLength="5"
    PrefixText="Strength: "
    HelpStatusLabelID="txtHelp_HelpLabel"
    TextStrengthDescriptions="Weak; Average; Strong; Excellent">
</cc1:PasswordStrength>
```

Code to Create a PasswordStrength Control

- ◆ Following table displays the password strength properties:

Property	Description
<code>DisplayPosition</code>	Describes the position of strength indicator text with respect to the target control ID.
<code>StrengthIndicatorType</code>	Specifies the type of indicator to be used to display the strength of the password. This property can have any one of the two values, namely <code>Text</code> or <code>BarIndicator</code> .
<code>PreferredPasswordLength</code>	Enables you to specify a preferred length for the password.
<code>PrefixText</code>	Enables you prefix a text to a password indicator. Here, it is set to <code>Strength</code> .

DynamicPopulateExtender

- ◆ Is an extender that places content into a control by making a Web service call. The contents can also be fetched by making a page call as shown in the figure.

```
<cc1:DynamicPopulateExtender ID="dypopexSearch"
runat="server"
Enabled="True"
ClearContentsDuringUpdate="true"
ServiceMethod="GetTime"
ServicePath="Class1.asmx"
TargetControlID="lblTextMessage"
PopulateTriggerControlID="txtSearch">
</cc1:DynamicPopulateExtender>
```

Code to Create a DynamicPopulateExtender Control

- ◆ Following table displays the extender control properties:

Property	Description
TargetControlID	Enables user to set the ID of the control on which this extender is supposed to function.
ServiceMethod	Specifies the method inside the service, which enables you to populate the control.
PopulateTriggerControlID	Allows user to set the name of the control on which certain actions would populate the target.

FilteredTextBox

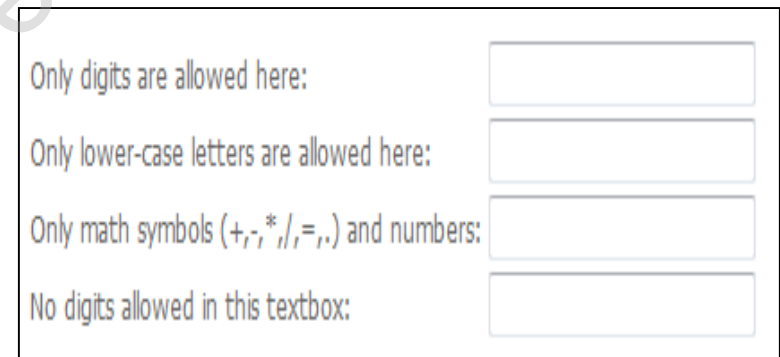
- ◆ Allows the user to define characters into a text box.
- ◆ Blocks users from entering invalid characters.

Code Snippet

```
<ajaxToolkit:FilteredTextBoxExtender ID="ftbe" runat="server"
    TargetControlID="TextBox3"      FilterType="Custom, Numbers"
    ValidChars="+-=/* () ." />
```

- ◆ Following figure and table displays the `FilteredTextBox` Control:

Property	Description
TargetControlID	Enables you to set the ID of the text box for the extender to operate on.
FilterType	Enables you to apply the type of filter desired.
ValidChars	Enables you to define the string which contains all characters considered valid for the text box.



Only digits are allowed here:

Only lower-case letters are allowed here:

Only math symbols (+, -, *, /, =, .) and numbers:

No digits allowed in this textbox:

Using the `FilteredTextBox` Control

ToggleButtonExtender Control

- Enables user to add custom images to show the state of a check box as shown in the figure and the table.

```
<cc1:ToggleButtonExtender ID="tgbtnexDirection"
runat="server"
Enabled="True"
TargetControlID="chkDirection"
UncheckedImageUrl=" ../App_Code/images/arrow_UP.gif"
CheckedImageUrl=" ../App_Code/images/arrow_DN.gif"
ImageWidth="60"
ImageHeight="60">
</cc1:ToggleButtonExtender>
```

Code to Create a ToggleButtonExtender Control

Property	Description
TargetControlID	Enables you to associate a control with the ToggleButtonExtender.
UncheckedImageUrl	Enables you to associate an image to represent uncheck mode of check box.
CheckedImageUrl	Enables you to associate an image to represent checked mode of check box.
ImageWidth	Specifies the width of the image.
ImageHeight	Specifies the height of the image.

Accordion Control

- ◆ Enables user to display multiple collapsible panes, one at a time.
- ◆ Enables user to club related information.
- ◆ Contains one or more `AccordionPane` controls, each of which contain `AccordionPane` controls consists of a `<Header>` and `<Content>`.
- ◆ Supports three `AutoSize` modes:

None



- Causes the Accordion to move without restriction.

Limit



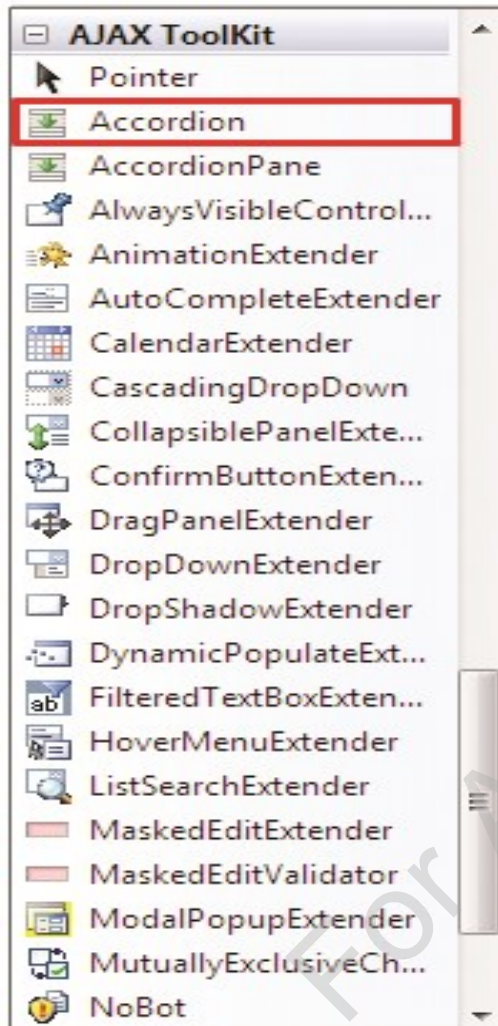
- Causes the Accordion to be fixed with a specific height, which can be set.

Fill



- Causes the Accordion to be fixed with a specific height.

- ◆ Following figure displays the Accordion control and the code:



Accordion Control

```
<cc1:Accordion ID="accrdAeroPlanes"
runat="server"
FadeTransitions="false"
FramesPerSecond="40"
SuppressHeaderPostBacks="true" TransitionDuration="250"
RequireOpenedPane="false"
AutoSize="Fill">
<Panes>
<cc1:AccordionPane ID="AccordionPane1" runat="server">
<Header><a href="">1. cars</a></Header>
<Content>
<p> Trying accordion control by opening pane1</p>
</Content>
</cc1:AccordionPane>
<cc1:AccordionPane ID="AccordionPane2" runat="server">
<Header><a href="">2. Aeroplanes </a></Header>
<Content>
<p>Trying accordion control by opening pane2</p>
</Content>
</cc1:AccordionPane>
</Panes>
</cc1:Accordion>
```

Code Showing Properties of Accordion

- ◆ Following figure displays the `Accordion` control properties:

Property	Description
<code>FadeTransitions</code>	Lets user specify a fading effect.
<code>FramesPerSecond</code>	Lets user set the number of frames displayed per second for a transition animation effect.
<code>TransitionDuration</code>	Specifies the number of milliseconds required for a transition animation.
<code>RequireOpenedPane</code>	Ensures that one pane is open always when the page is loaded.
<code><Panes></code>	Enables user to add <code>Accordion</code> panes.
<code><Header></code>	Enables user to specify a header to the pane.
<code><Content></code>	Enables user to specify the content that is to be included inside the pane.

- ◆ Allows the user to get automatic population of a set of DropDownList control as shown in the following figure and the table:

```
<cc1:CascadingDropDown
  ID="casddStates"
  runat="server"
  Category="Country"
  Enabled="True"
  ParentControlID="ddlCountry"
  ServiceMethod="GetStates"
  ServicePath="WebService.asmx"
  TargetControlID="ddlStates">
</cc1:CascadingDropDown>
```

Code Showing Properties of CascadingDropDown

Property	Description
Category	Represents the name of the category, which the DropDownList represents.
ParentControlID	Specifies the ID of Parent to which the current DropDownList is associated.
ServiceMethod	Specifies the service method to be called.
ServicePath	Specifies the service to be called.

CascadingDropDown Control

- ◆ Logic is included to retrieve countries and populate into the DropDownList of the code-behind file as shown in the following code:

Code Snippet:

```
protected void Page_Load(object sender, EventArgs e)
{
    //Code to retrieve countries from database
    ...
}
```

- ◆ Following code defined the Web method in the Web service:

```
[WebMethod]
    public AjaxControlToolkit.CascadingDropDownNameValue[] GetStates(string
knownCategoryValues, string category)
    {
        //This service is called to populate the other DropDownList
        // Depending on the value selected in the parent
        // DropDownList
        string[] _categoryValues = knownCategoryValues.Split(':', ';');
        string _country = _categoryValues[1];
        List<CascadingDropDownNameValue> _states = new
List<CascadingDropDownNameValue>();

        StatesTableAdapter _statesAdapter = new StatesTableAdapter();
        foreach (DataRow _row in _statesAdapter.GetStates(_country))
        {
            _states.Add(new
CascadingDropDownNameValue(_row["StateName"].ToString(),
_row["StateID"].ToString()));
        }
        return _states.ToArray();
    }
```

ConfirmButtonExtender Control

- ◆ Enables user to confirm a client interaction, like a click of a button, with some custom message as shown in the code snippet.

Code Snippet

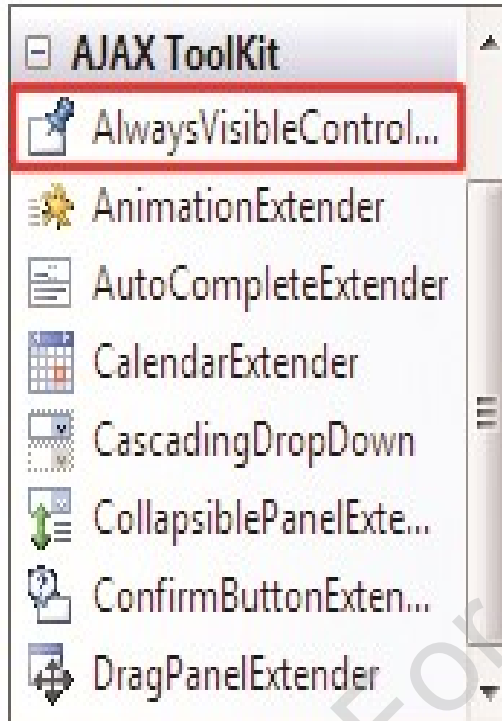
```
ccl:ConfirmButtonExtender ID="cnfbtnexSubmit" runat="server"  
ConfirmText="Are you sure you want to exit?"  
Enabled="True"  
TargetControlID="btnSubmit">
```

Following table lists the properties of the ConfirmButtonExtender:

Property	Description
TargetControlID	Specifies the control that is to be extended with ConfirmButtonExtender control.
Confirmtext	Specifies the confirmation text that needs to be displayed.

AlwaysVisibleControlExtender Control

- ◆ Enables user to set the controls on the page at a fixed position.
- ◆ Helps user keep the controls at fixed position from the vertical as well as horizontal edges.
- ◆ Following figure and table displays the AlwaysVisibleControlExtender control.



**AlwaysVisibleControl
Extender Control**

Property	Description
HorizontalOffset	This property enables you to set the position of the control with respect to the horizontal edge of the browser.
ScrollEffectDuration	This property specifies the duration of scrolling effect to last when the control is repositioned.
HorizontalSide	This property enables you anchor the control with respect to the horizontal edge of browser.
VerticalOffset	This property enables you to set the position of control with respect to the vertical edge of the browser.
VerticalSide	This property enables you anchor the control with respect to the vertical edge of browser.

AlwaysVisibleControlExtender Control

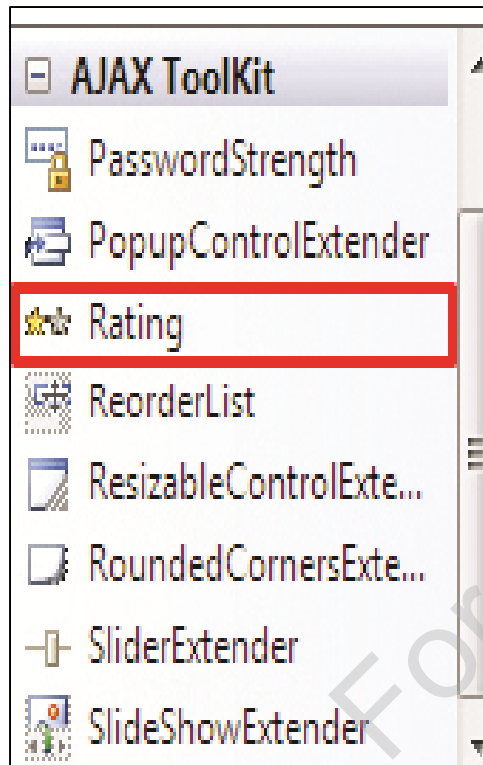
- ◆ Following code shows properties of the AlwaysVisibleControlExtender Control:

Code Snippet

```
<cc1:AlwaysVisibleControlExtender ID="alwvsblexHelp"
runat="server" Enabled="True"
TargetControlID="btnHelp"
HorizontalOffset="5"
ScrollEffectDuration=".10"
HorizontalSide="Center"
VerticalOffset="5"
VerticalSide="Top">
</cc1:AlwaysVisibleControlExtender>
```

Rating Control

- ◆ Enables user to specify ratings for elements on a Web page.
- ◆ Specifies the initial rating, maximum rating, and apply different styles to indicate state of stars.
- ◆ Supports custom scripts to be run when a user submits rating.
- ◆ Following figure and table displays the details of the Rating control:



Rating Control

Property	Description
CurrentRating	This property lets you set an initial value for rating.
MaxRating	This property lets you set a maximum value for rating.
StarCssClass	This property lets you associate a CSS class, which enables the styles for the stars to be displayed.
FilledStarCssClass	This property lets you associate a CSS class, which enables the styles for the stars, which are selected.

Rating Control

- ◆ Following code shows properties of the Rating control:

Code Snippet

```
<cc1:Rating ID="RatingA" runat="server"
CurrentRating="2"
MaxRating="5"
StarCssClass="star"
WaitingStarCssClass="waiting"
FilledStarCssClass="filledstar"
EmptyStarCssClass="emptystar"
style="float: left;"
BackColor="#FFFFCC"
Height="35px"
ReadOnly="True"
Width="177px" />
```

TabContainer Control

- ◆ Enables user to create a set of tabs that can help to organize Web page content.
- ◆ Contains one or more TabPanel controls.
- ◆ Following figure and table displays the properties of the TabContainer control.

```
<cc1:TabContainer ID="tbcLogin" runat="server"
ActiveTabIndex="1"
Height="181px" Width="601px">
<cc1:TabPanel runat="server"
HeaderText="Login" ID="tbpn1Login"
ScrollBars="Horizontal">
<ContentTemplate>
<asp:Label Text="Enter password"
ID="lblPwd" runat="server">
<asp:TextBox Text="" runat="server" ID="txtPwd">
</asp:TextBox>
<br />
<asp:Button runat="server" Text="Submit" />
</asp:Label>
</ContentTemplate>
</cc1:TabPanel>
</cc1:TabContainer>
```

Property	Description
HeaderText	Sets the header to be displayed for a tab.
ScrollBars	Determines whether to display a scroll bar in the tab.
<ContentTemplate>	Determines the content that is to be displayed inside the tab.

Properties of the TabContainer Control

ColorPicker Control

- ◆ Allows user to select colors from the color palette when the focus is changed to an input element.
- ◆ Following codes create the JavaScript function that is called in response to the `OnClientColorSelectionChanged` event:

Code Snippet

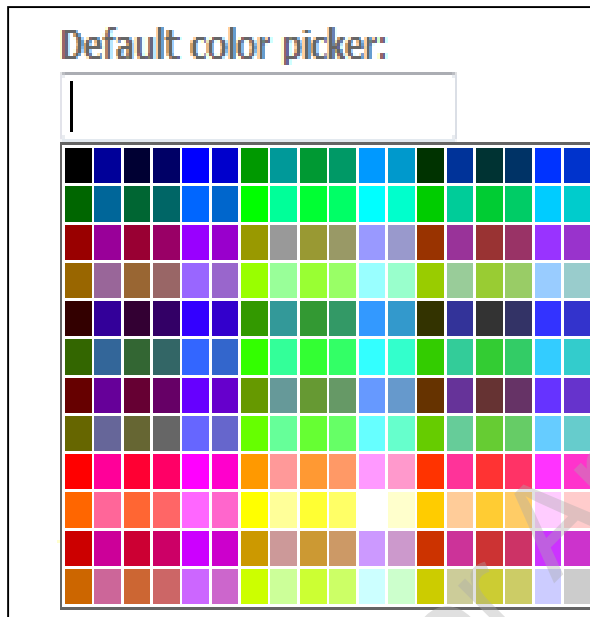
```
<ajaxToolkit:ColorPickerExtender runat="server" ID="CPE1"
TargetControlID="Clr1" OnClientColorSelectionChanged="colorChanged"
/>
```

```
function colorChanged(sender) {
    sender.get_element().style.color =
        "#" + sender.get_selectedColor();
}

<ajaxToolkit:ColorPickerExtender runat="server"
TargetControlID="Clr2"
PopupButtonID="Img1"
SampleControlID="Sample1"
SelectedColor="33ffcc" />
```

ColorPicker Control

- Following figure and table displays the `ColorPicker` control:



Default ColorPicker

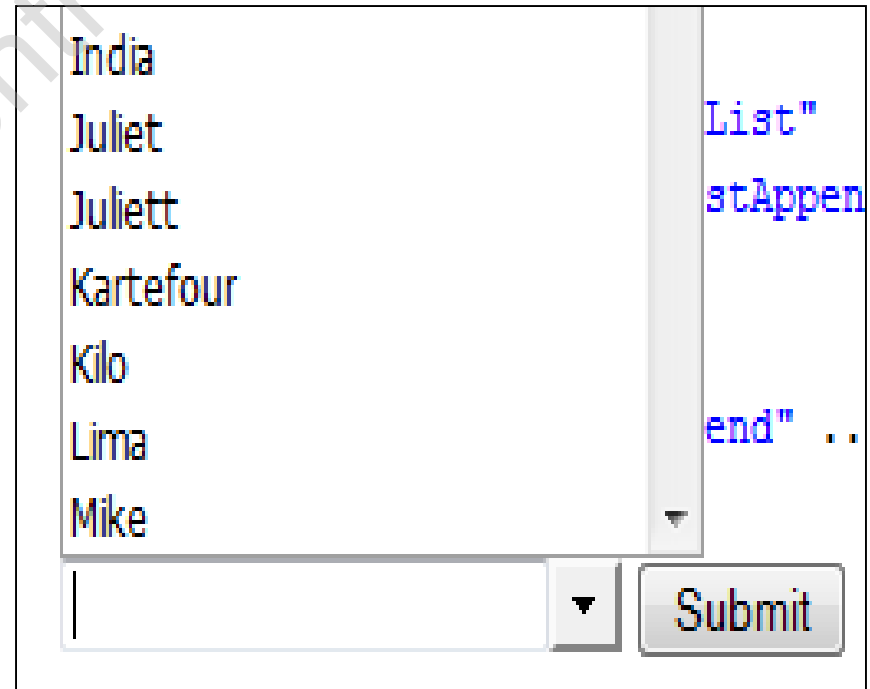
Property	Description
OnClientColorSelectionChanged	Is called when the <code>colorSelectionChanged</code> event is raised.
TargetControlID	Sets the ID of the text box to extend with the color picker.
PopupButtonID	Enables the display of the color picker popup.
SampleControlID	Helps to display the color selected.
Popup Position	Allows user to specify the place where the color picker pop-up must appear.
Selected Color	Indicates the value of the color that is initialized in the <code>ColorPicker</code> .

ComboBox Control

- ◆ Allows user to make use of the `TextBox` along with the `DropDownList` control. The user can select either from the existing list or can initialize with a new set of items.
- ◆ Following code and figure displays the details of the `ComboBox` control:

Code Snippet

```
<ajaxToolkit:ComboBox ID="Cmb1"
runat="server"
DropDownStyle="DropDown"
    AutoCompleteMode="None"
    CaseSensitive="false"
    RenderMode="Inline"
    ItemInsertLocation="Append"
ListItemHoverCssClass="ComboBoxLi
stItemHover"
<asp:ListItem>...</asp:ListIem>
```



ComboBox Control

- ◆ Following table displays the properties of the `ComboBox` control:

Property	Description
DropDownStyle	<p>This property determines two aspects. One, if the user is permitted to enter text that is not present in the list. The second is to determine if the list can be always displayed or not. There are three settings in this. They are as follows:</p> <ul style="list-style-type: none">• <code>DropDownList</code>: Unmatched text cannot be entered• <code>DropDown</code>: Any text can be entered• <code>Simple</code>: Any text can be entered and the list is always displayed
AutoCompleteMode	<p>This property determines how typed text is automatically completed in the <code>ComboBox</code>. There are three settings in this. They are as follows:</p> <ul style="list-style-type: none">• <code>Append</code>: Reminder of the first-matched item is appended to the user-typed text and the appended text is highlighted.• <code>SuggestAppend</code>: Both of the previous specified actions are applied.• <code>None</code>: The control's auto-complete behaviors are disabled.
CaseSensitive	<p>This property determines if the match between user-typed text and existing list items is done in a case-sensitive manner. The default setting for this property is 'false'.</p>

Property	Description
RenderMode	This property defines if the <code>ComboBox</code> is performed as an 'Inline' or 'Block' level HTML element. The default setting for this is 'Inline'.
ItemInsertLocation	This property determines if new items inserted into the list should be appended or prepended, or they should be inserted in an 'Ordinal' manner. The default setting is 'Append'.
ListItemHoverCssClass	This property helps to replace the default styles with a custom CSS class.
ListItem	This uses more than one child controls for declaring items that are added to the <code>ComboBox</code> list.

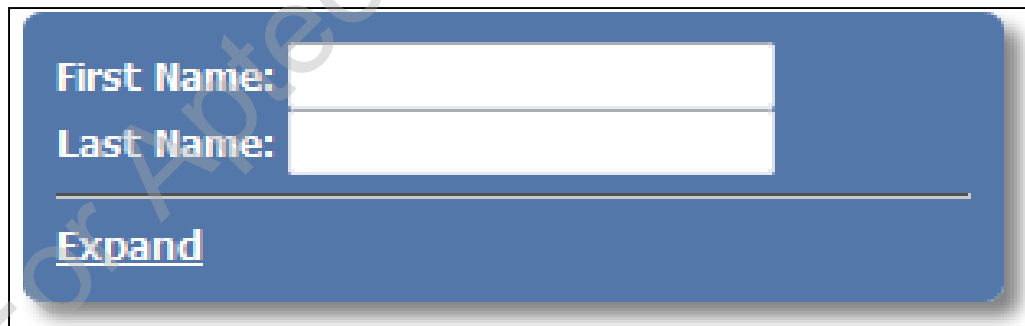
DropShadow Control

- ◆ Allows user to apply a drop shadow to the ASP.NET panel controls as shown in the code snippet.

Code Snippet

```
<ajaxToolkit:DropShadowExtender ID="dse" runat="server"  
TargetControlID="Pnl1" Opacity=".8" Rounded="true"  
TrackPosition="true" /> None"
```

- ◆ Following figure displays the details of the DropShadow control:



First Name:

Last Name:

Expand

DropShadow Control

- ◆ Following table displays the details of the DropShadow control:

Property	Description
TargetControlID	This property sets the ID of the control to be extended.
Width	This property specifies the width of the drop shadow. It is given in pixels and the default setting is 5.
Opacity	The opacity of the drop shadow can be specified using this property. It ranges from 0 to 1.0 (fully transparent to fully opaque). The default setting is 0.5.
TrackPosition	This property lets the user to attach a Boolean value to track the panel position to which the shadow is attached. The default setting is 'false'.
Rounded	This property lets the user to decide the requirement of rounded corners for the target and drop shadow and accordingly specify a Boolean value.

DynamicPopulate Control

- ◆ Allows user to replace the control contents with the results of Web service or a page method call as shown in the code snippet.

Code Snippet

```
<ajaxToolkit:DynamicPopulateExtender ID="dp" runat="server"
    TargetControlID="Pnl1"
    ClearContentsDuringUpdate="true"
    PopulateTriggerControlID="Lbl1"
    ServiceMethod="GetHtml"
    UpdatingCssClass="dynamicPopulate_Updating" />
```

Time at the server:

Choose a date/time format:

- ☐ Normal
☐ Short Date
☒ Long Date
☐ UTC Date/Time

This time is dynamically formatted and returned as HTML from the server:

Monday, January 27, 2014

DynamicPopulate Control

DynamicPopulate Control

- ◆ Following table displays the details of the `DynamicPopulate` control:

Property	Description
<code>TargetControlID</code>	This property sets the ID of the button or link of this extender.
<code>ClearContentsDuringUpdate</code>	On the start of an update, this property can be used to clear the target elements' HTML contents.
<code>ServicePath</code>	This property defines the URL of the service path that is to be called.
<code>ServiceMethod</code>	This is the name of the method which is to be called on the Web service or page.
<code>PopulateTriggerControlID</code>	This property defines the optional control name that will trigger the target population.
<code>UpdatingCssClass</code>	This is the CSS class applied to the target when asynchronous calls are made.

HoverMenu Control

- ◆ Correlates the controls with a popup panel that displays extra content.
- ◆ When the mouse cursor is moved over the main control, a popup panel becomes visible at the position that has been specified by the developer and a CSS style is used on the control which specifies it as hot.
- ◆ Following figure and code shows the details of the HoverMenu control:

Mouse over the grid below to see the options for each row.

Have product idea	Figure out opportunities	0
Talk to customers	Make sure they want it	1
Design product	Figure out the features and architecture	2
Build prototype	Work out the issues	3
Test features	Use TDD and automated testing	4
Build production version	Make it fast and robust	5
Fix bugs	Make sure it works	6
Ship	Ship it!	7

Code Snippet

```
<ajaxToolkit:HoverMenuExtender
ID="hme2" runat="Server"
TargetControlID="Pnl9"

    PopupControlID="PopupMenu"
    HoverCssClass="popupHover"
    PopupPosition="Left"

    OffsetX="0" OffsetY="0"
    PopDelay="50" />
```

HoverMenu Control

- ◆ Following table displays the properties of the `HoverMenu` control:

Property	Description
TargetControlID	This property sets the target that is being targeted by the extender. Placing the cursor over this control, displays the hover menu popup.
HoverCssClass	When the hover menu is displayed, this is the CSS class that is applied to the target.
PopupPostion	This specifies the place where the popup should be positioned. The various positions are Left (Default), Right, Top, Bottom, and Center.
OffsetX/OffsetY	This defines the number of pixels required to move the popup from its default position.
HoverDelay	This specifies the time duration after which the popup should be displayed after hovering over the target control. The default setting is 0 milliseconds.
PopDelay	This specifies the time duration for which the popup should remain visible after the mouse is moved away from the target control. The default setting is 100 milliseconds.
Animations	This property helps user specify generic animations for the <code>HoverMenu</code> extender.

HTMLEditor Control

- ◆ Simplifies the process of creating and editing HTML content.
- ◆ The generated HTML markup and the preview document too can be viewed by the user.
- ◆ Following code displays the use of HTMLEditor control:

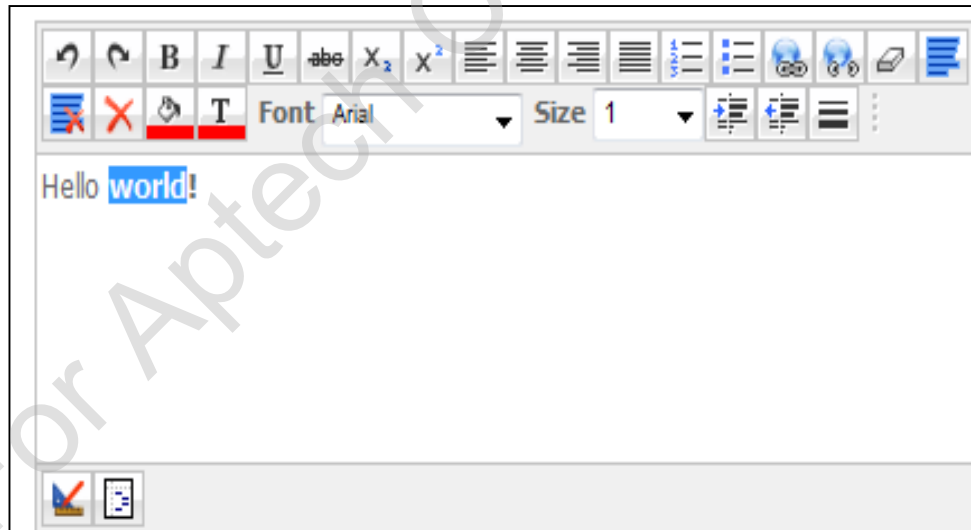
Code Snippet

```
<%@ Register
    Assembly="AjaxControlToolkit"
    Namespace="AjaxControlToolkit.HTMLEditor"
    TagPrefix="HTMLEditor" %>

...
<HTMLEditor:Editor runat="server"
    Height="400px"
    Width="80%"
    AutoFocus="true"/>
```

- ◆ Following table and figure displays the details of the `HTMLEditor` control:

Property	Description
<code>ActiveMode</code>	Allows you make active the editing panel control.
<code>AutoFocus</code>	Allows you focus the editing panel and set the cursor inside it.
<code>Content</code>	This gets or sets the <code>HTMLEditor</code> content.
<code>CssClass</code>	This is used to define and customize the look and feel of <code>HTMLEditor</code> .



HTMLEditor Control

ListSearch Control

- ◆ Allows the user to search for typed items in the `ListBox` or the `DropDownList`.
- ◆ Following code and figure displays the details of a `ListSearch` control.

Code Snippet

```
<ajaxToolkit:ListSearchExtender id="LSE" runat="server"
    TargetControlID="LstBx1" PromptText="Type to search"
    PromptCssClass="ListSearchExtenderPrompt"
    PromptPosition="Top" AutoResetTimeout="0" IsSorted="true"/>
```



ListSearch Control

ListSearch Control

- ◆ Following table and figure displays the details of the `ListSearch` control:

Property	Description
<code>PromptText</code>	This displays the message that is to be shown when the <code>ListBox</code> is focused upon.
<code>PromptCssClass</code>	This property lets the user to define the CSS class which is to be applied to the prompt message.
<code>PromptPosition</code>	This determines the location where the message should appear.
<code>QueryPattern</code>	This identifies the pattern in which the characters typed should be utilized in the search query.
<code>IsSorted</code>	This property determines if the items added in the list should be sorted or not.
<code>QueryTimeout</code>	This property determines the need to reset the search query after the timeout in case there is no match found.
<code>Animations</code>	This property lets the user define generic animations for this extender.

ModalPopup Control

- ◆ Allows the user to show content inside an element that resembles a modal dialog box.
- ◆ This prevents the user from interacting with the balance of the page.
- ◆ Any hierarchy of controls can be given to the modal content as shown in the code snippet.

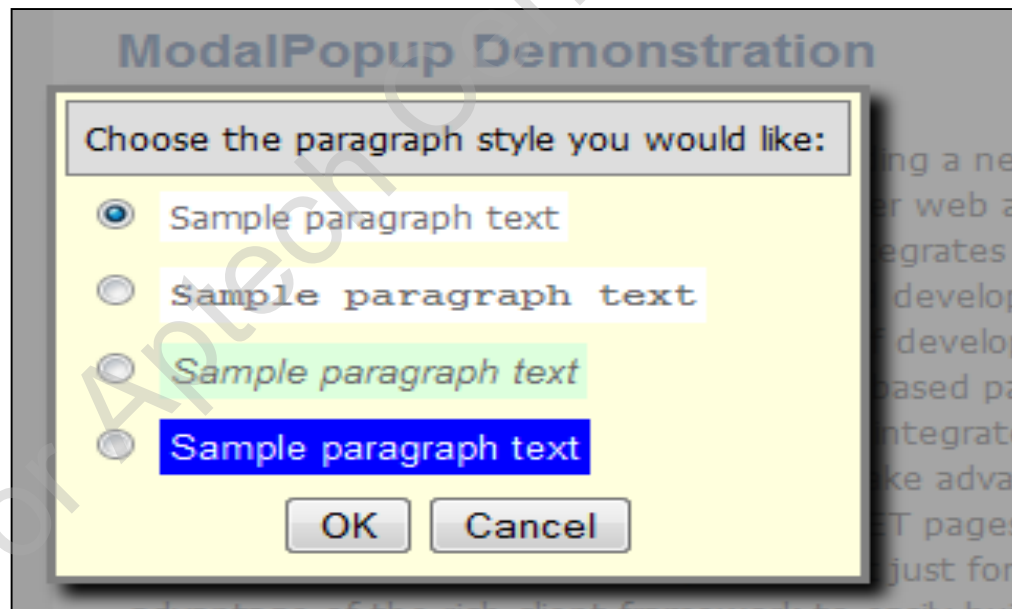
Code Snippet

```
<ajaxToolkit:ModalPopupExtender ID="MPE" runat="server"
    TargetControlID="LnkBtn1"
    PopupControlID="Pnl1"
    BackgroundCssClass="modalBackground"
    DropShadow="true"
    OkControlID="OkButton"
    OnOkScript="onOk()"
    CancelControlID="CancelButton"
    PopupDragHandleControlID="Pnl3" />
```

ModalPopup Control

- ◆ Following table and figure displays the details of the ModalPopup control:

Property	Description
TargetControlID	This property sets the element ID which is responsible for activating the modal popup.
PopupControlID	This property sets the element ID which is to be displayed as a modal popup.



ModalPopup Control

TabContainer Control

- ◆ Allows user to create a set of tabs that can be used to organize page content.
- ◆ Is a host for a number of `TabPanel` controls as shown in the code snippet.

Code Snippet

```
<ajaxToolkit:TabContainer runat="server"
    OnClientActiveTabChanged="ClientFunction"
    Height="150px">
    <ajaxToolkit:TabPanel runat="server"
        HeaderText="Signature and Bio"
        <ContentTemplate>
            ...
        </ContentTemplate>
    />
</ajaxToolkit:TabContainer>
```

TabContainer Control

- ◆ Following table displays the properties of the TabContainer control:

Property	Description
ActiveTabChanged (Event)	This is used on the server side whenever a tab gets changed after a postback.
OnClientActiveTabChanged	This is the name given to the JavaScript function that is attached to the client-side tabChanged event.
CssClass	This defines a customized look and feel for the tabs.
ActiveTabIndex	This determines the first and foremost tab to be shown.
Height	The height of the tabs' body is specified here.
Width	The width of the tabs' body is specified here.
ScrollBars	This determines if a scroll bar should be displayed in the Tab Container body or not.
TabStripPlacement	This specifies the location of the tabs.

- ◆ Following figure displays the details of the TabContainer control:

The image shows a web application window titled "Toolkit User Profile:". Inside, there is a tabbed interface with four tabs: "Signature and Bio", "Email", "Controls", and "Dynamic". The "Signature and Bio" tab is currently selected and highlighted with an orange border. Below the tabs, there are two text input fields. The first is labeled "Signature:" and the second is labeled "Bio:". Below these fields is a "Save" button. At the bottom of the window, there is a text instruction: "Hit Save to cause a postback from an update panel inside the tab panel." A large, diagonal watermark reading "For Apteck Centre Use Only" is overlaid across the entire screenshot.

TabContainer Control

- The AJAX Control Toolkit is a collection of server-side controls that provides powerful features and rich functionality to Web applications.
- Associating the client component that contains new functionality with the server control is done by an extender control.
- AutoCompleteExtender is a control extender that helps in achieving this functionality.
- FilteredTextBox is an extender that allows users to define only valid characters in a text box.
- DynamicPopulate helps replace control contents with the results of a Web service or page method call.
- The ListSearchExtender allows the user to search for items in a ListBox or DropDownList through typing.
- Using the ModalPopup extender, the user can display content inside an element that resembles a modal dialog box.