

Working with ASP.NET Server Controls

Asst. Prof. Dr. Özgü Can

Server Controls

- Mixing plain **HTML** and **server-side control** in classic ASP or PHP
 - To create a text box with a message and current date and time:

```
<input type="text" value="Hello World, the time is <%=Time()%>" />
```



Difficult to write and manage pages.

Server Controls

- In ASP.NET, controls “live” on the server inside an **ASPX page**.

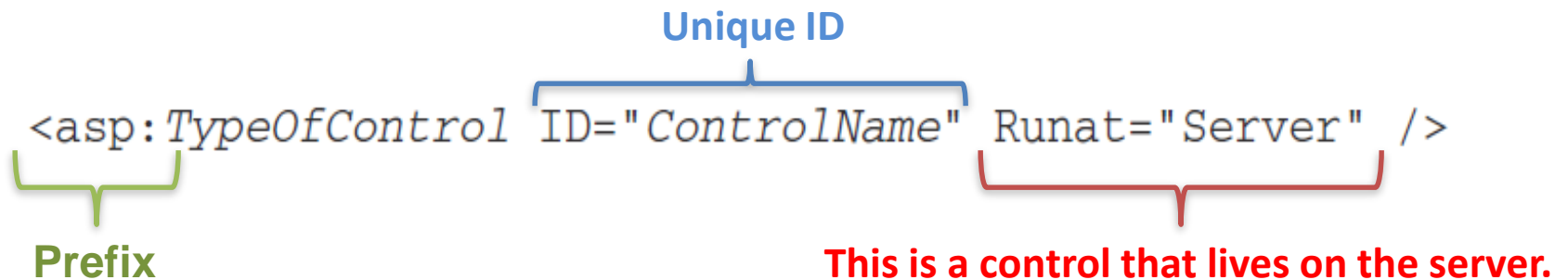
1. Browser requests the page
2. The server-side controls are processed by the ASP.NET engine

responsible for receiving and processing requests for ASPX pages

1. Controls emit client-side HTML code
2. HTML code ends up in the browser

Server Controls

- Instead of defining HTML controls in your pages directly, you define an ASP.NET server control:

The diagram shows the tag `<asp:TypeOfControl ID="ControlName" Runat="Server" />`. A green bracket under `asp:` is labeled "Prefix". A blue bracket under `ID="ControlName"` is labeled "Unique ID". A red bracket under `Runat="Server"` is labeled "This is a control that lives on the server."

```
<asp:TypeOfControl ID="ControlName" Runat="Server" />
```

Prefix

Unique ID

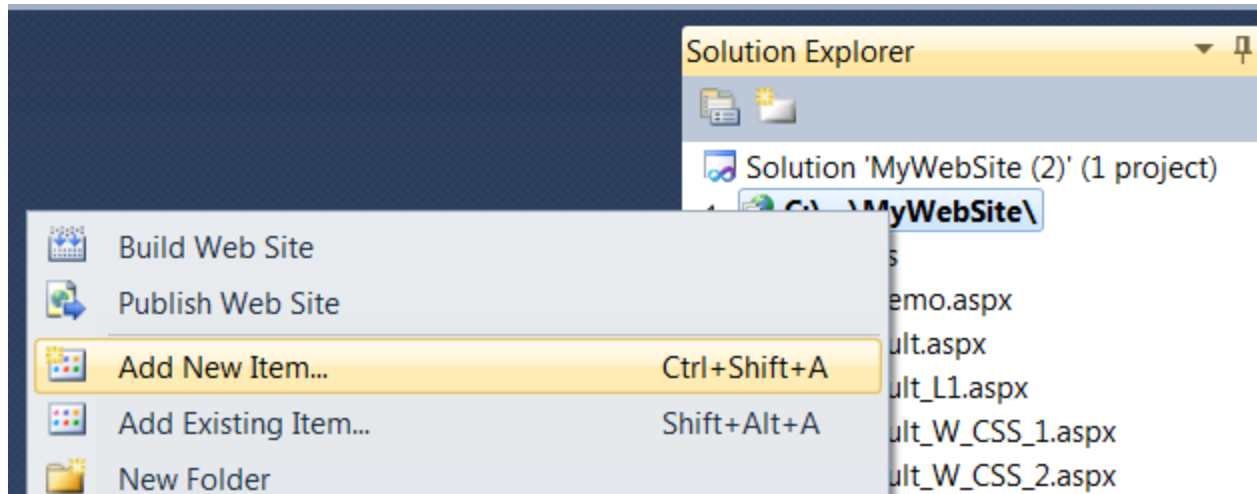
This is a control that lives on the server.

```
<asp:TextBox ID="Message" Runat="Server" />
```

```
<asp:TextBox ID="Message" Runat="Server"></asp:TextBox>
```

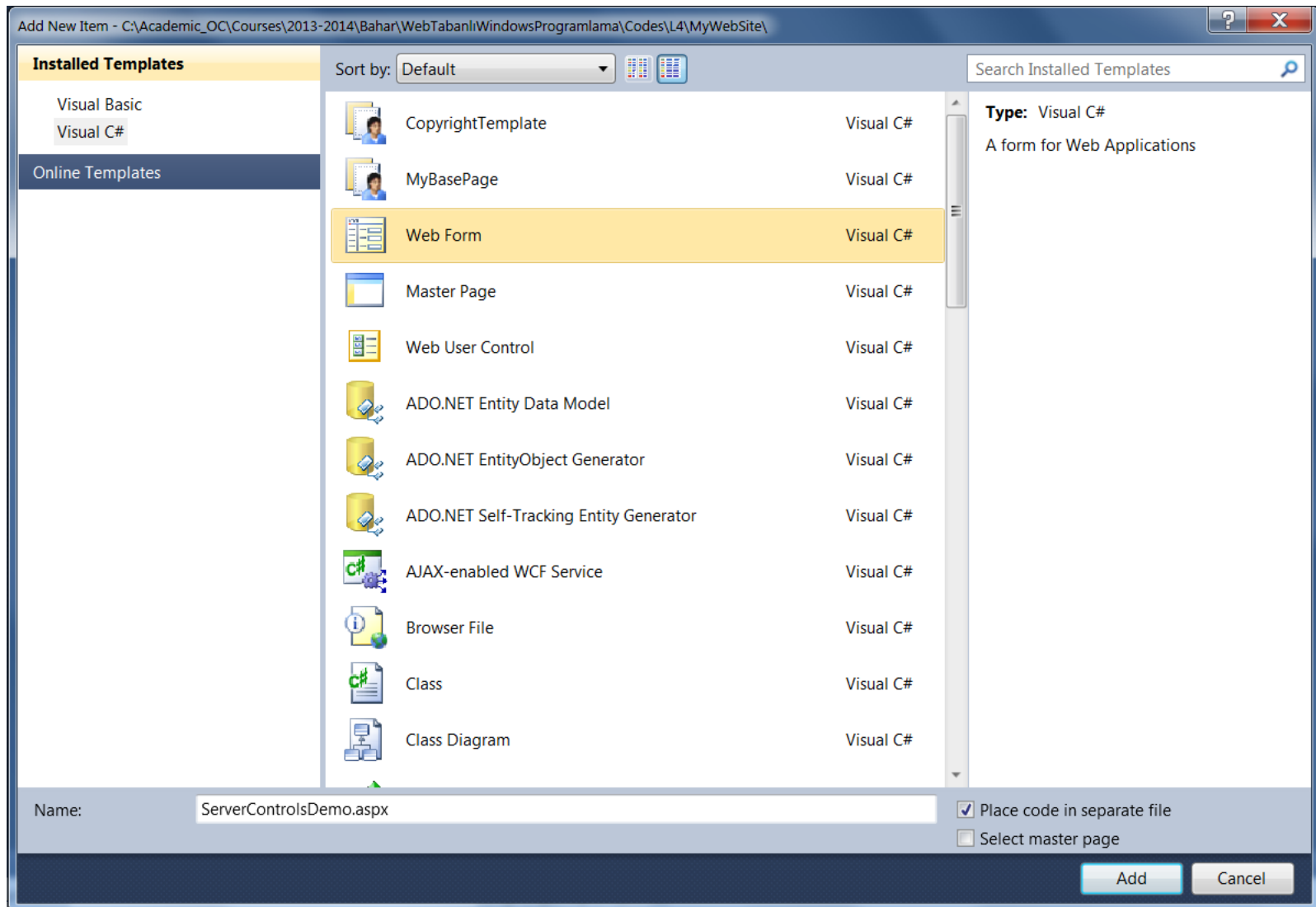
Example

Working with Server Controls



Example

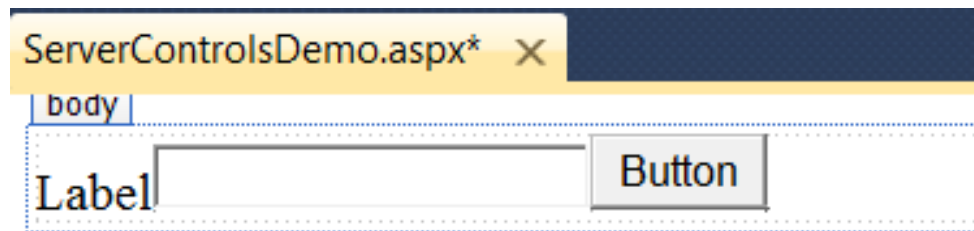
Working with Server Controls



Example

Working with Server Controls

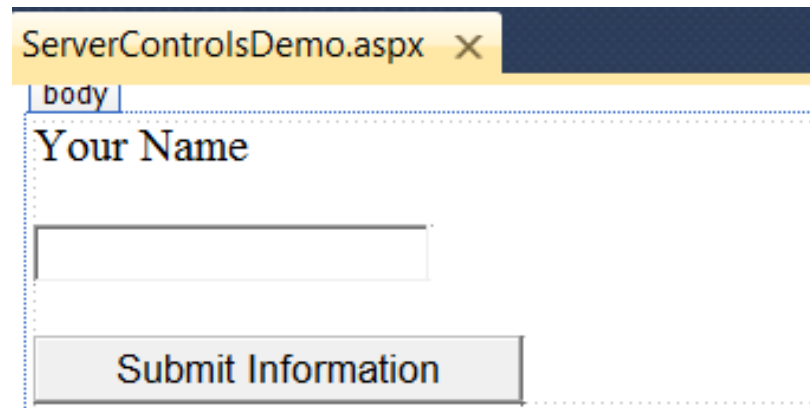
- In Design View:
 - Add **Label**, **TextBox** and a **Button**



Example

Working with Server Controls

- Properties:
 - Label → **Text : Your Name**
 - Button → **Text: Submit Information**
ID: submitButton
 - TextBox → **ID: yourNameTextBox**



The screenshot shows a web browser window with the title 'ServerControlsDemo.aspx'. The page content includes a label 'Your Name' followed by a text input field. Below the input field is a button labeled 'Submit Information'. The entire form is enclosed in a dashed blue border, and the label 'body' is visible in the top left corner of the page content area.

Example

Working with Server Controls

Double-click `submitButton`

```
protected void submitButton_Click(object sender, EventArgs e)
{
    Label1.Text = "Your name is " + yourNameTextBox.Text;
}
```

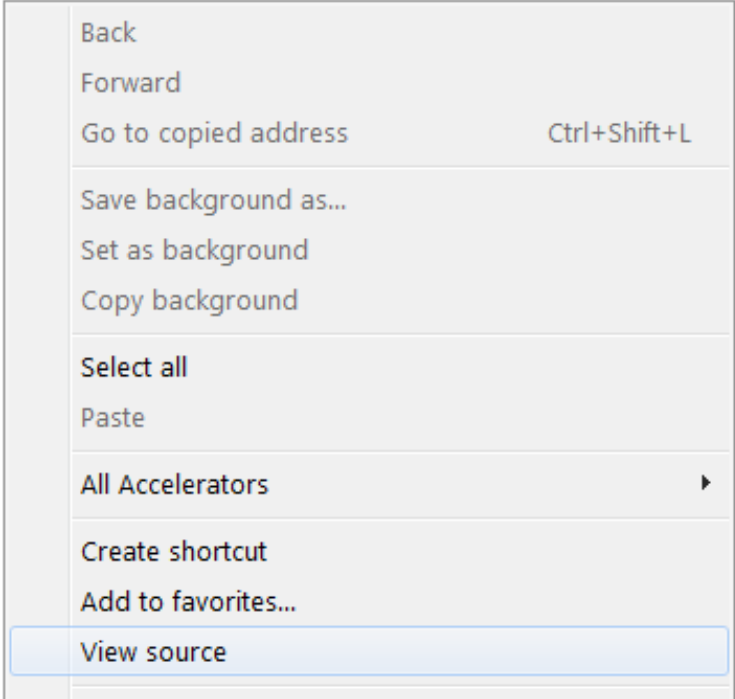
Example

Working with Server Controls

- View in browser.
- Don't click button!
- View Source

Your Name

Submit Information



A screenshot of a web browser's context menu. The menu is open, showing various options. The 'View source' option at the bottom is highlighted with a blue border. Other visible options include 'Back', 'Forward', 'Go to copied address' (with a keyboard shortcut 'Ctrl+Shift+L'), 'Save background as...', 'Set as background', 'Copy background', 'Select all', 'Paste', 'All Accelerators' (with a right-pointing arrow), 'Create shortcut', and 'Add to favorites...'. The background of the browser window is a light gray.

- Back
- Forward
- Go to copied address Ctrl+Shift+L
- Save background as...
- Set as background
- Copy background
- Select all
- Paste
- All Accelerators ▶
- Create shortcut
- Add to favorites...
- View source**

Example

Working with Server Controls

```
<div>

    <span id="Label1">Your Name</span>
    <br />
    <br />
    <input name="yourNameTextBox" type="text" id="yourNameTextBox" />
    <br />
    <br />
    <input type="submit" name="submitButton" value="Submit Information" id="submitButton" />
</div>
```

Example

Working with Server Controls

Your Name



Your name is Özgü

- View Source

```
<div>
```

```
  <span id="Label1">Your name is Özgü</span>
```

```
  <br />
```

```
  <br />
```

```
  <input name="yourNameTextBox" type="text" value="Özgü" id="yourNameTextBox" />
```

```
  <br />
```

```
  <br />
```

```
  <input type="submit" name="submitButton" value="Submit Information" id="submitButton" />
```

```
</div>
```

Common Properties for All Controls

PROPERTY	DESCRIPTION
<code>AccessKey</code>	Enables you to set a key with which a control can be accessed at the client by pressing the associated letter.
<code>BackColor</code> <code>ForeColor</code>	Enables you to change the color of the background (<code>BackColor</code>) and text (<code>ForeColor</code>) of the control.
<code>BorderColor</code> <code>BorderStyle</code> <code>BorderWidth</code>	Changes the border of the control in the browser. Each of these three ASP.NET properties maps directly to its CSS counterpart.
<code>CssClass</code>	Lets you define the HTML <code>class</code> attribute for the control in the browser. This class name then points to a CSS class you defined in the page or an external CSS file.
<code>Enabled</code>	Determines whether the user can interact with the control in the browser. For example, with a disabled text box (<code>Enabled="False"</code>) you cannot change its text.
<code>Font</code>	Enables you to define different font-related settings, such as <code>Font-Size</code> , <code>Font-Names</code> , and <code>Font-Bold</code> .
<code>Height</code> <code>Width</code>	Determines the height and width of the control in the browser.
<code>TabIndex</code>	Sets the client-side HTML <code>tabindex</code> attribute that determines the order in which users can move through the controls in the page by pressing the Tab key.
<code>ToolTip</code>	Enables you to set a tooltip for the control in the browser. This tooltip, rendered as a <code>title</code> attribute in the HTML, is shown when the user hovers the mouse over the relevant HTML element.
<code>Visible</code>	Determines whether or not the control is sent to the browser. You should really see this as a server-side visibility setting because an invisible control is never sent to the browser at all. This means it's quite different from the CSS <code>display</code> and <code>visibility</code> properties you saw in the previous chapter that hide the element at the client.

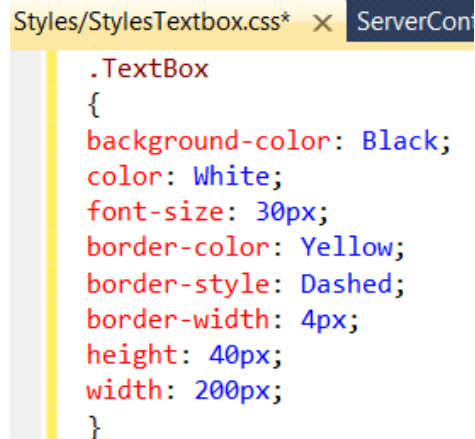
Common Properties for All Controls

```
<form id="form1" runat="server">
<div>
  <asp:TextBox ID="TextBox1" AccessKey="a" BackColor="Black" ForeColor="White" Font-Size="30px"
  BorderColor="Yellow" BorderStyle="Dashed" BorderWidth="4" CssClass="TextBox"
  Enabled="True" Height="40" Width="200" TabIndex="1" ToolTip="Hover text here"
  Visible="True" runat="server" Text="Hello World" ></asp:TextBox>
</div>
```

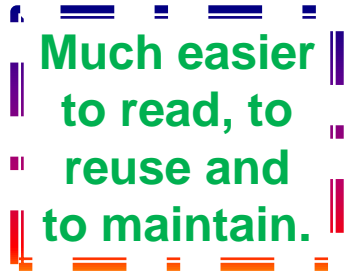


Hello World

```
<div>
  <asp:TextBox ID="TextBox1" runat="server" AccessKey="a" CssClass="TextBox" TabIndex="1"
  ToolTip="Hover text here" Text="Hello World"></asp:TextBox>
</div>
```

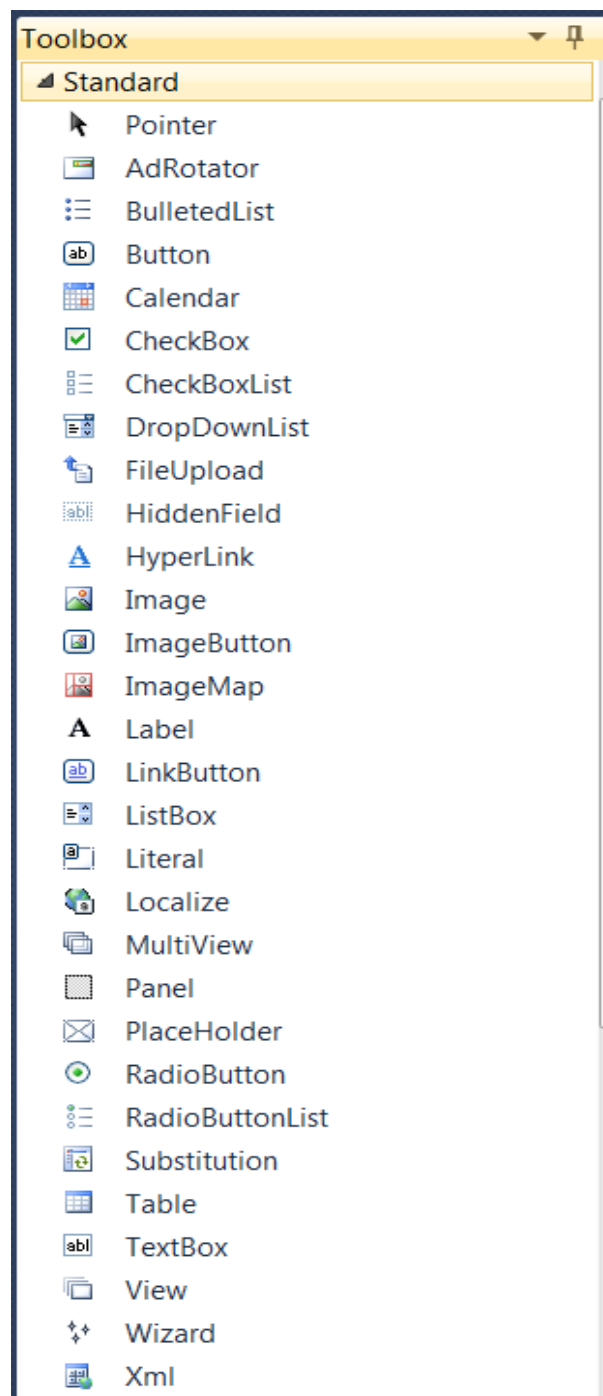


```
Styles/StylesTextbox.css* × ServerCon
{
background-color: Black;
color: White;
font-size: 30px;
border-color: Yellow;
border-style: Dashed;
border-width: 4px;
height: 40px;
width: 200px;
}
```



Much easier
to read, to
reuse and
to maintain.

Standard Controls



Simple Controls

- The Toolbox contains a number of **simple and straightforward controls**, including:
 - `TextBox`,
 - `Button`,
 - `Label`,
 - `HyperLink`,
 - `RadioButton`, and
 - `CheckBox`.

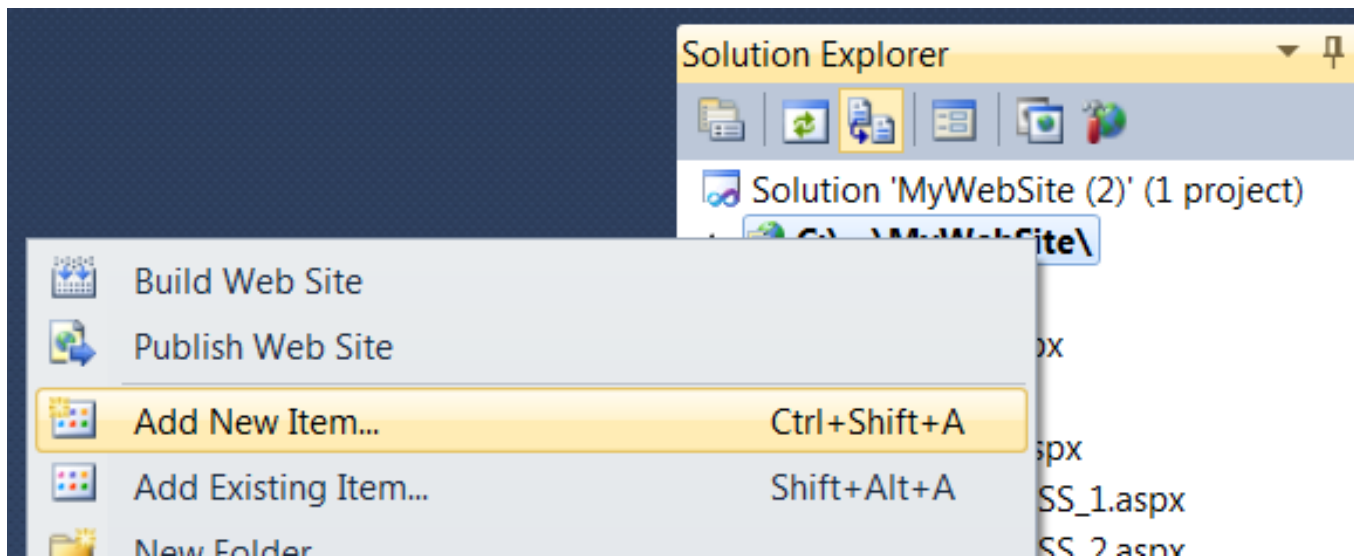
List Controls

- The standard category also contains a number of controls that **present themselves as lists in the browser.**
- These controls include:
 - `ListBox`,
 - `DropDownList`,
 - `CheckBoxList`,
 - `RadioButtonList`, and
 - `BulletedList`.

Example

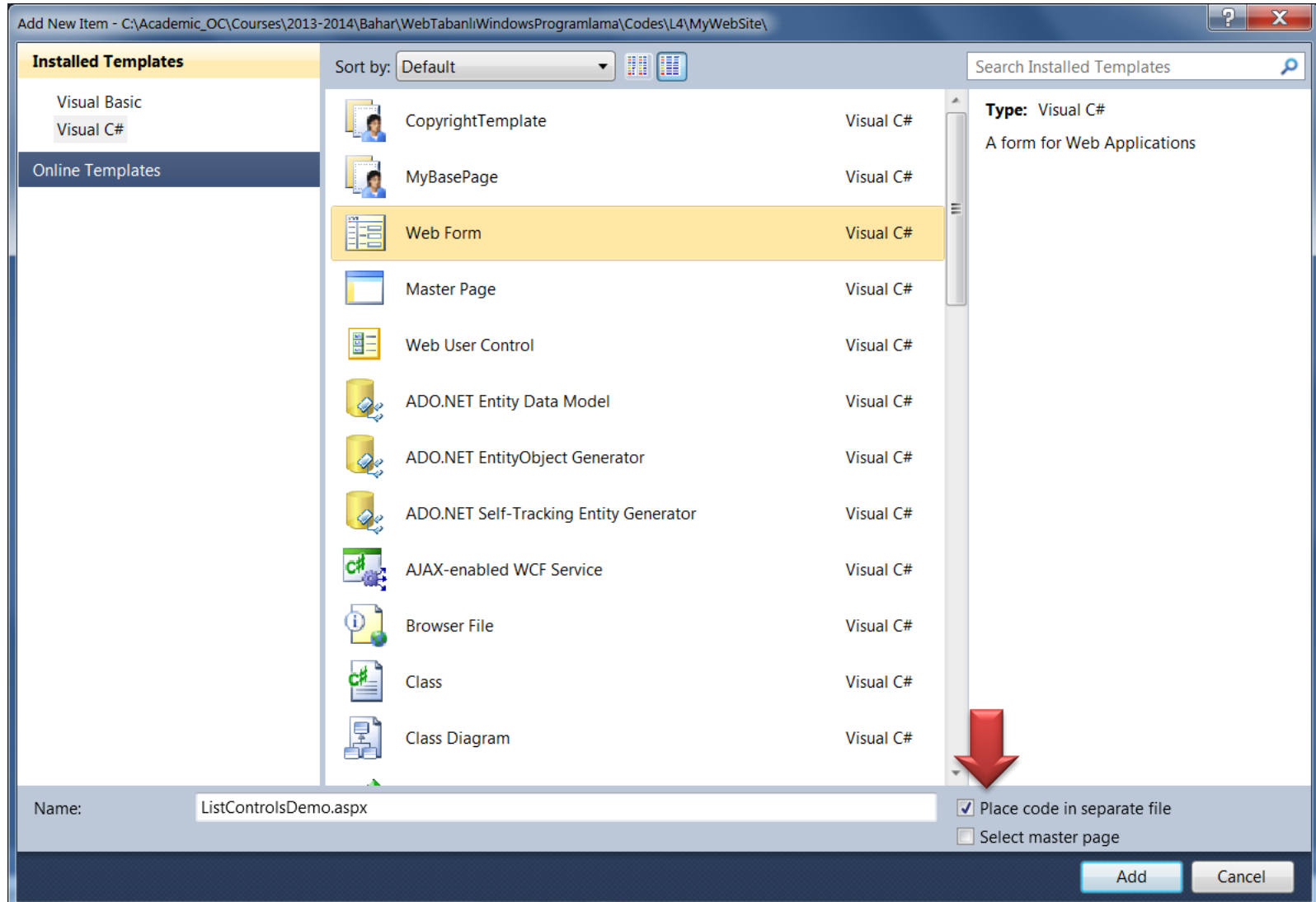
Working with List Controls

Add New Item in the Solution Explorer



Example

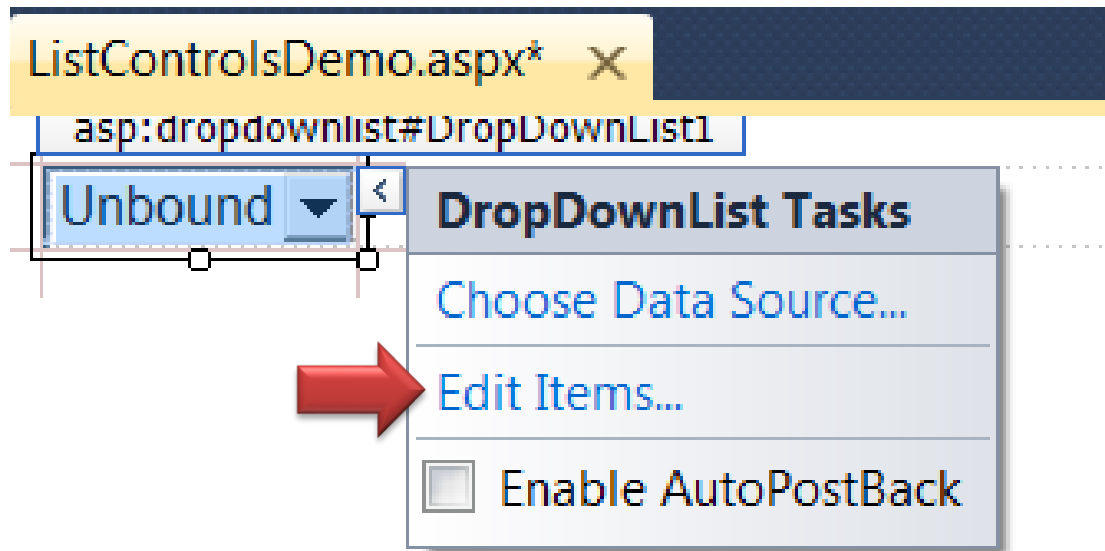
Working with List Controls



Example

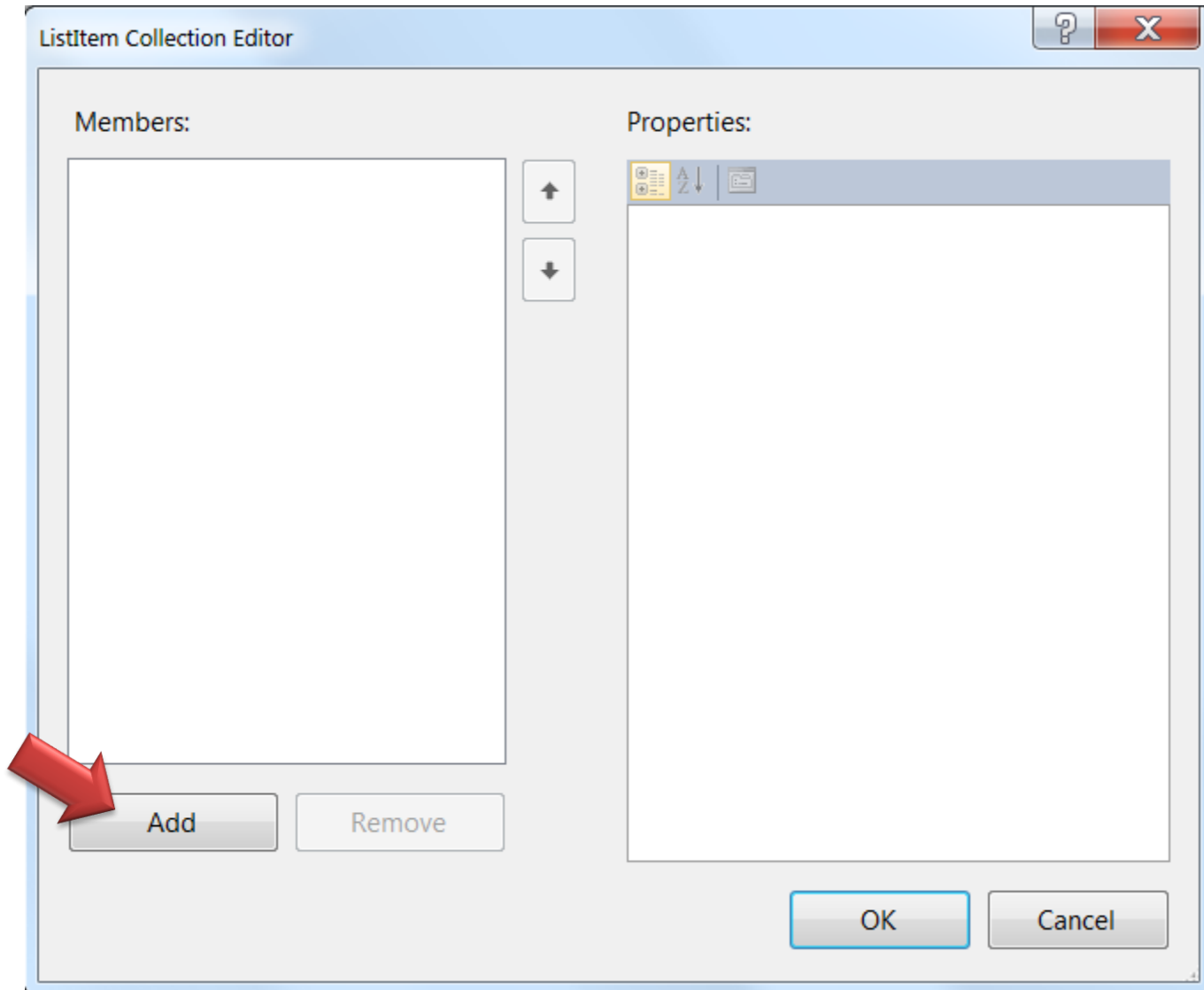
Working with List Controls

- In Design View:
 - Drag a **DropDownList** control



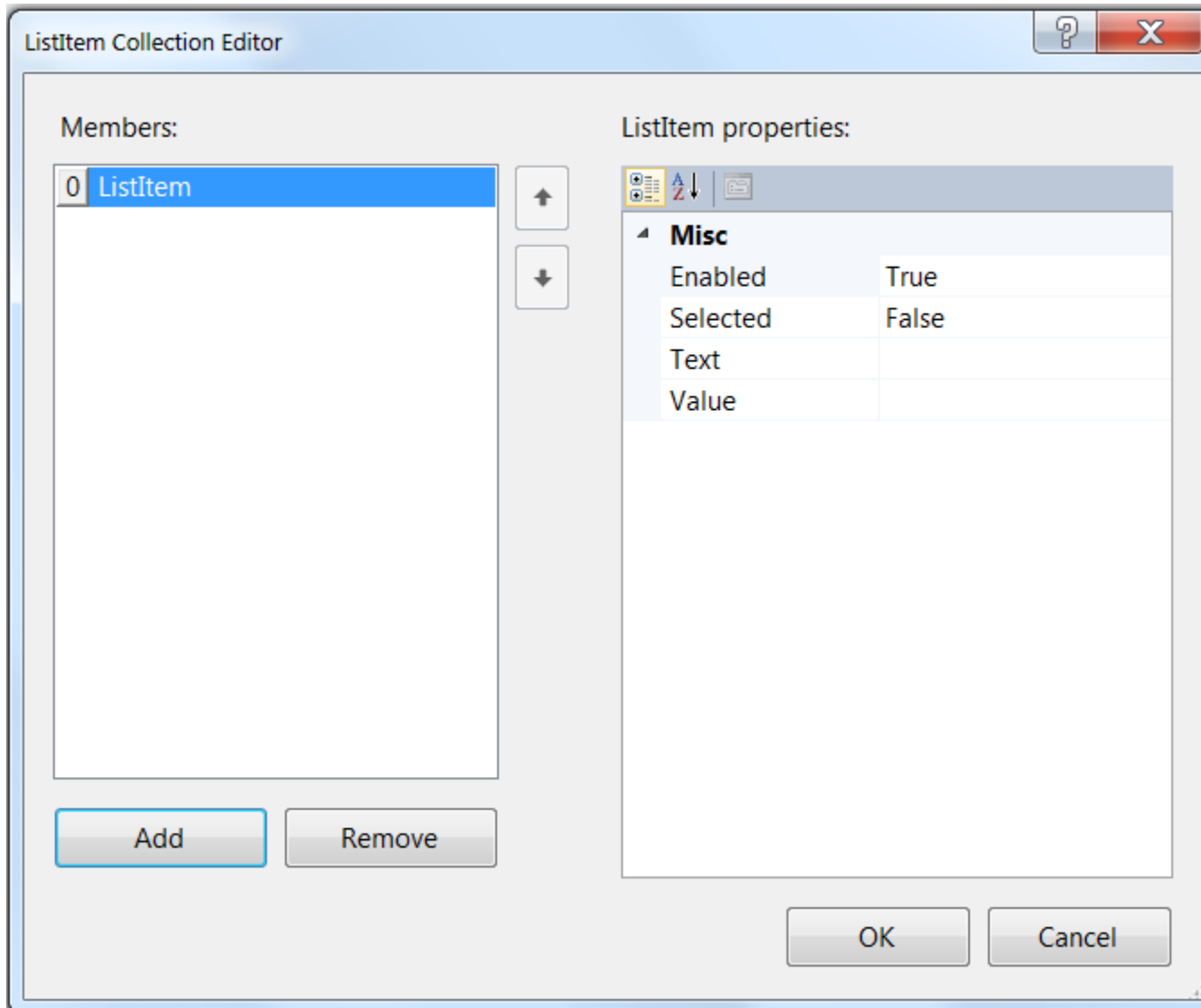
Example

Working with List Controls



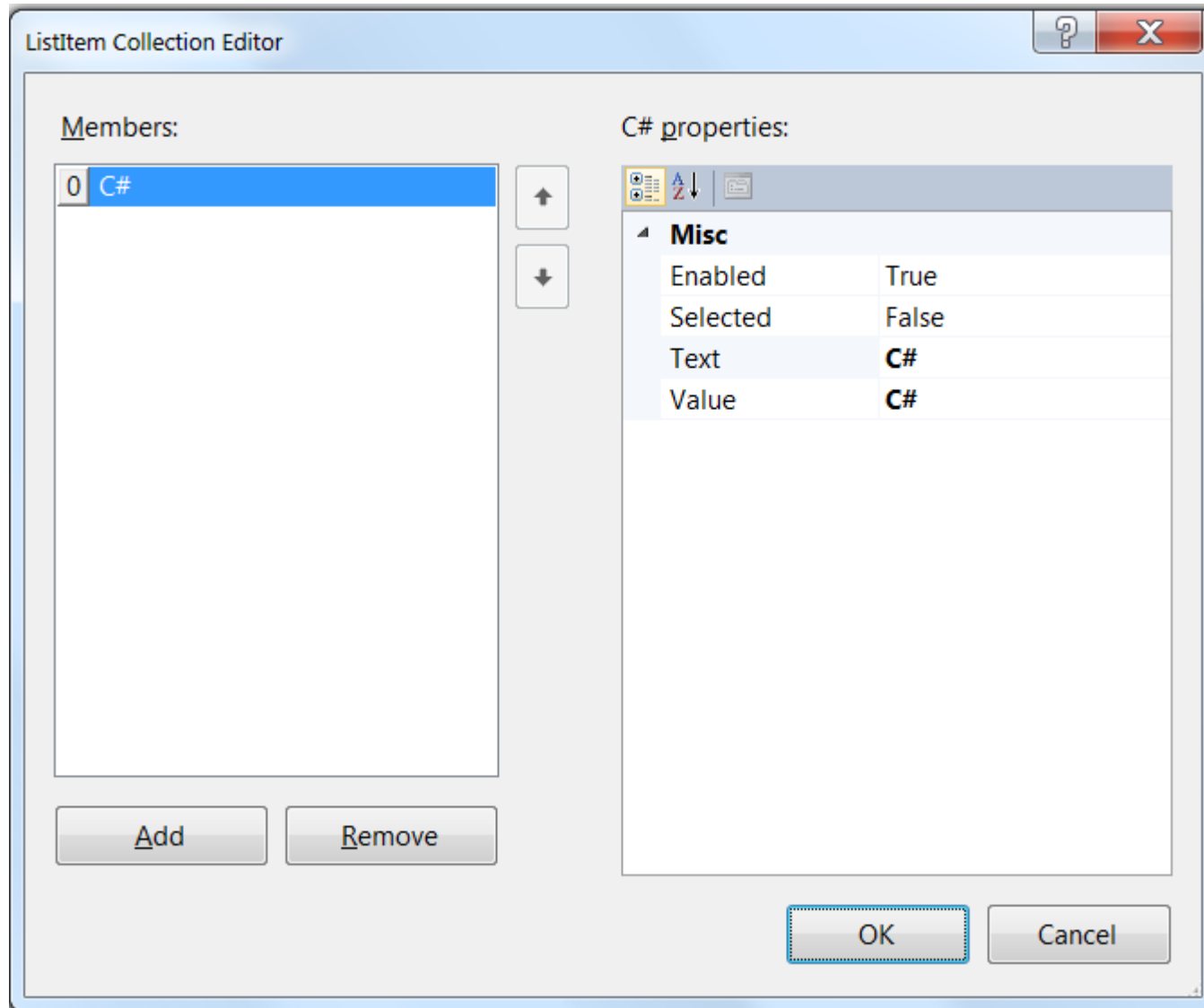
Example

Working with List Controls

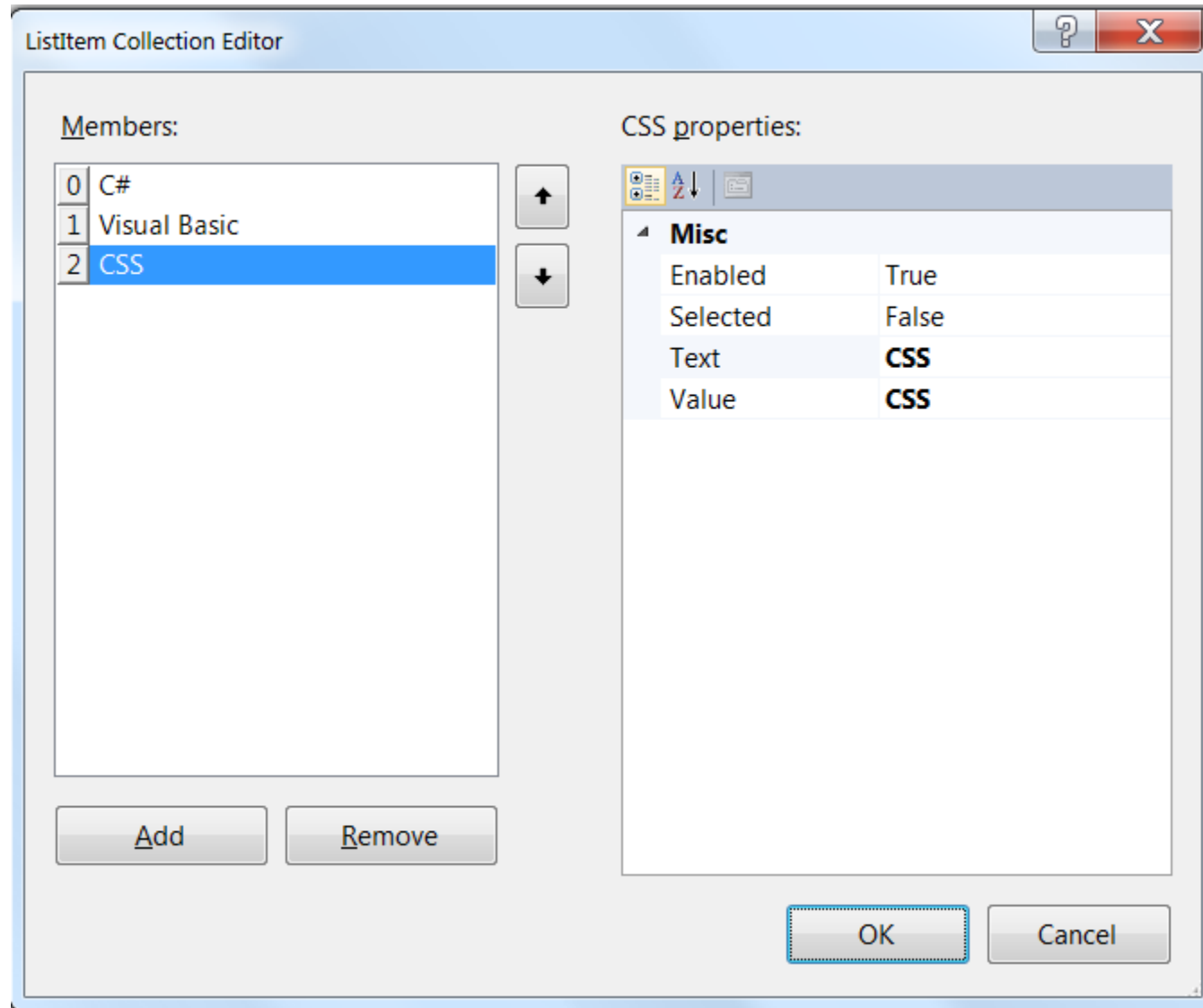


Example

Working with List Controls



Repeat creating list items for **Visual Basic** and **CSS**



Example

Working with List Controls

- In Source View:

```
<asp:DropDownList ID="DropDownList1" runat="server">  
    <asp:ListItem>C#</asp:ListItem>  
    <asp:ListItem>Visual Basic</asp:ListItem>  
    <asp:ListItem>CSS</asp:ListItem>  
</asp:DropDownList>
```

- Drag a **CheckBoxList** control after the **DropDownList** control.

```
<asp:DropDownList ID="DropDownList1" runat="server">  
    <asp:ListItem>C#</asp:ListItem>  
    <asp:ListItem>Visual Basic</asp:ListItem>  
    <asp:ListItem>CSS</asp:ListItem>  
</asp:DropDownList>  
<asp:CheckBoxList ID="CheckBoxList1" runat="server">  
</asp:CheckBoxList>
```

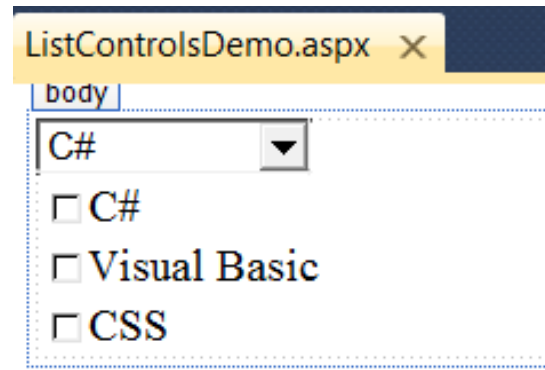
Example

Working with List Controls

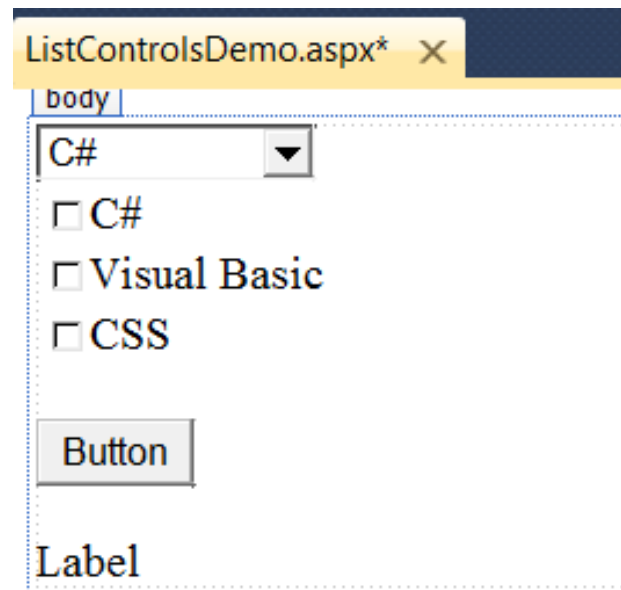
- Copy the three `<asp:ListItem>` elements from the `DropDownList` you created and paste them between the opening and closing tags of the `CheckBoxList`.

```
<asp:CheckBoxList ID="CheckBoxList1" runat="server">  
    <asp:ListItem>C#</asp:ListItem>  
    <asp:ListItem>Visual Basic</asp:ListItem>  
    <asp:ListItem>CSS</asp:ListItem>  
</asp:CheckBoxList>
```

- In Design View:



- Drag a **Button** and a **Label** after the **CheckBoxList** control



Example

Working with List Controls

- Clear **Label1** text
- Double-click the **Button**

```
protected void Button1_Click(object sender, EventArgs e)
{
    Label1.Text = "In the DropDownList you selected " + DropDownList1.SelectedValue + "<br />";
    foreach (ListItem item in CheckBoxList1.Items)
    {
        if (item.Selected == true)
        {
            Label1.Text += "In the CheckBoxList you selected " + item.Value + "<br />";
        }
    }
}
```

Example

Working with List Controls

- View in Browser

C# ▼

☐ C#

☐ Visual Basic

☐ CSS

Button

Visual Basic ▼

☒ C#

☐ Visual Basic

☒ CSS

Button

In the DropDownList you selected Visual Basic
In the CheckBoxList you selected C#
In the CheckBoxList you selected CSS

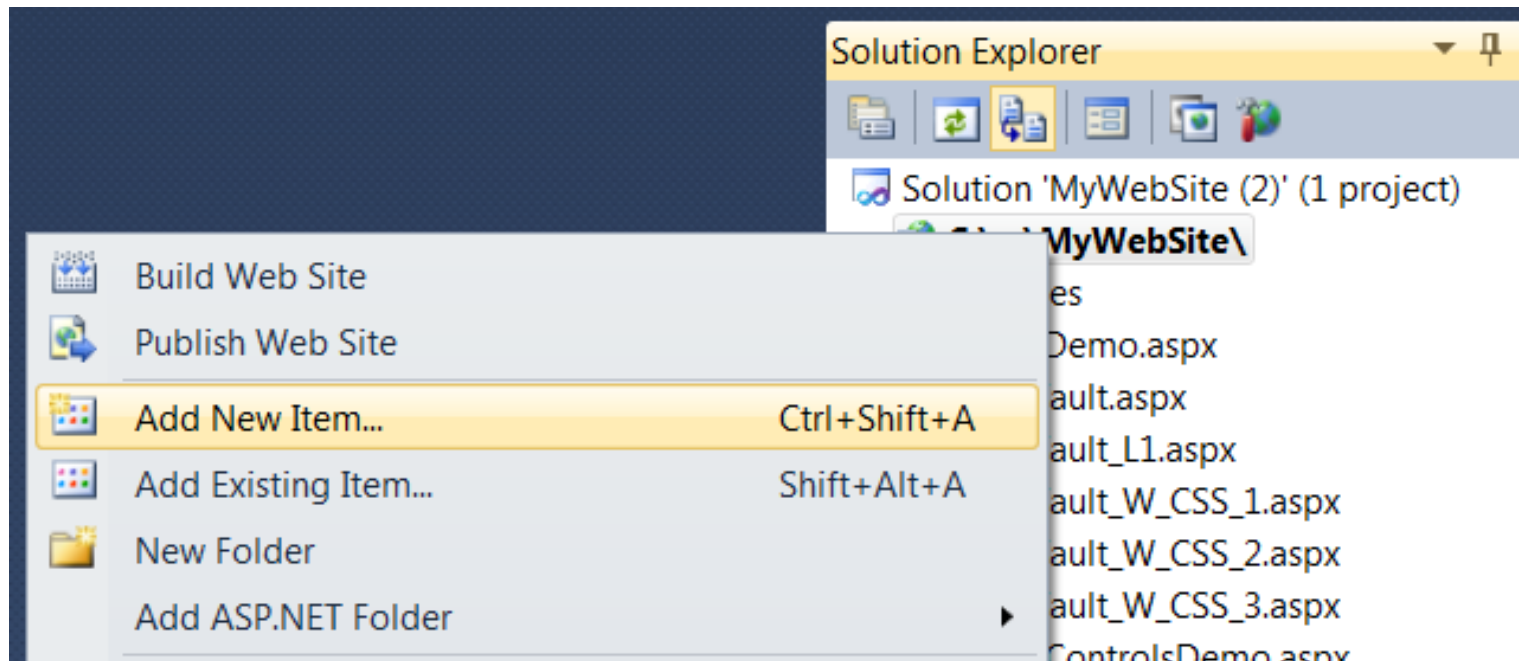
Container Controls

- It's desirable to have the ability to **group related content and controls**:
 - Panel
 - Placeholder
 - MultiView
 - Wizard

Example

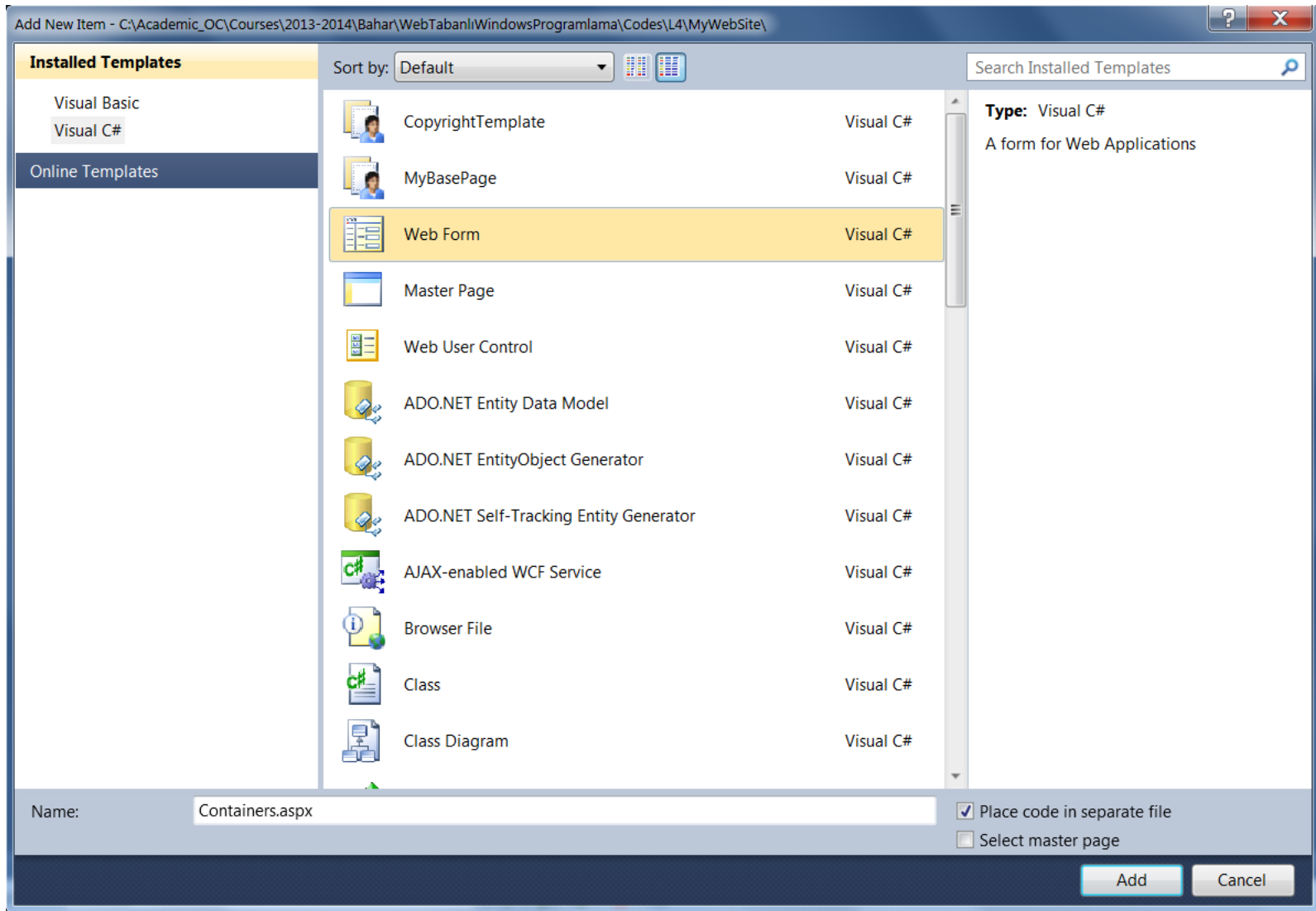
Panel Control

- Add New Item in the Solution Explorer



Example

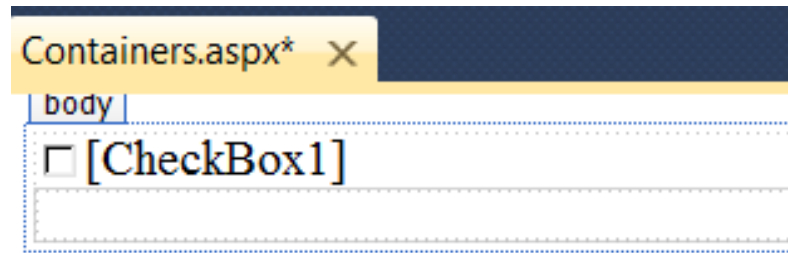
Panel Control



Example

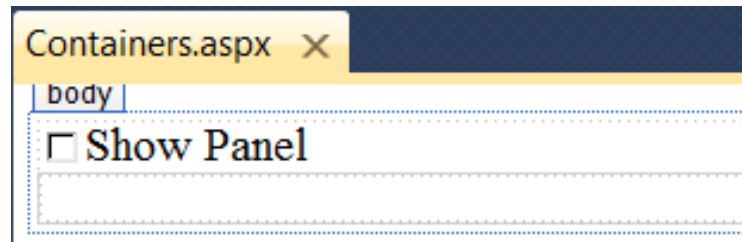
Panel Control

- Drag a **CheckBox** and a **Panel**



- **Properties**
 - **CheckBox** → **Text = Show Panel**
AutoPostBack = True
 - **Panel** → **Visible = False**

Example Panel Control



Type some text inside **Panel** → **I am visible now!**



```
<asp:CheckBox ID="CheckBox1" runat="server" AutoPostBack="True"
    Text="Show Panel" />
<asp:Panel ID="Panel1" runat="server" Visible="False">
    I am visible now!</asp:Panel>
```

Example

Panel Control

Double-click the **CheckBox** control.

```
protected void CheckBox1_CheckedChanged(object sender, EventArgs e)
{
    Panel1.Visible = CheckBox1.Checked;
}
```

Example

Panel Control

- View in browser & source

☐ Show Panel

```
<div>  
  
    <asp:CheckBox ID="CheckBox1" runat="server" AutoPostBack="True"  
        oncheckedchanged="CheckBox1_CheckedChanged" Text="Show Panel" />  
    <asp:Panel ID="Panel1" runat="server" Visible="False">  
        I am visible now!</asp:Panel>  
  
</div>
```

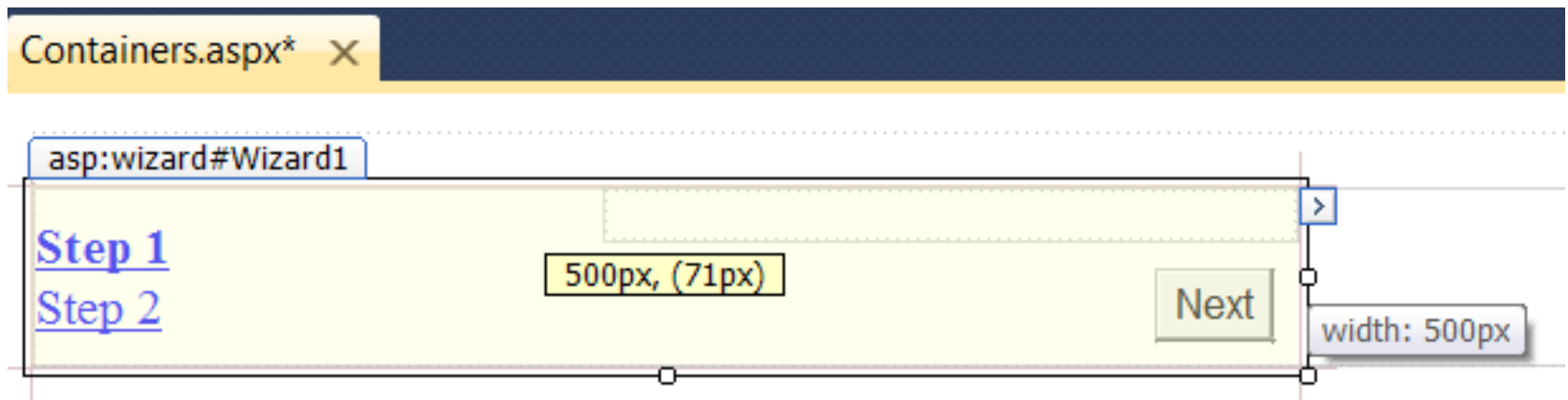
☒ Show Panel
I am visible now!

```
    <div id="Panel1">  
        I am visible now!  
    </div>
```

Example

Wizard Control

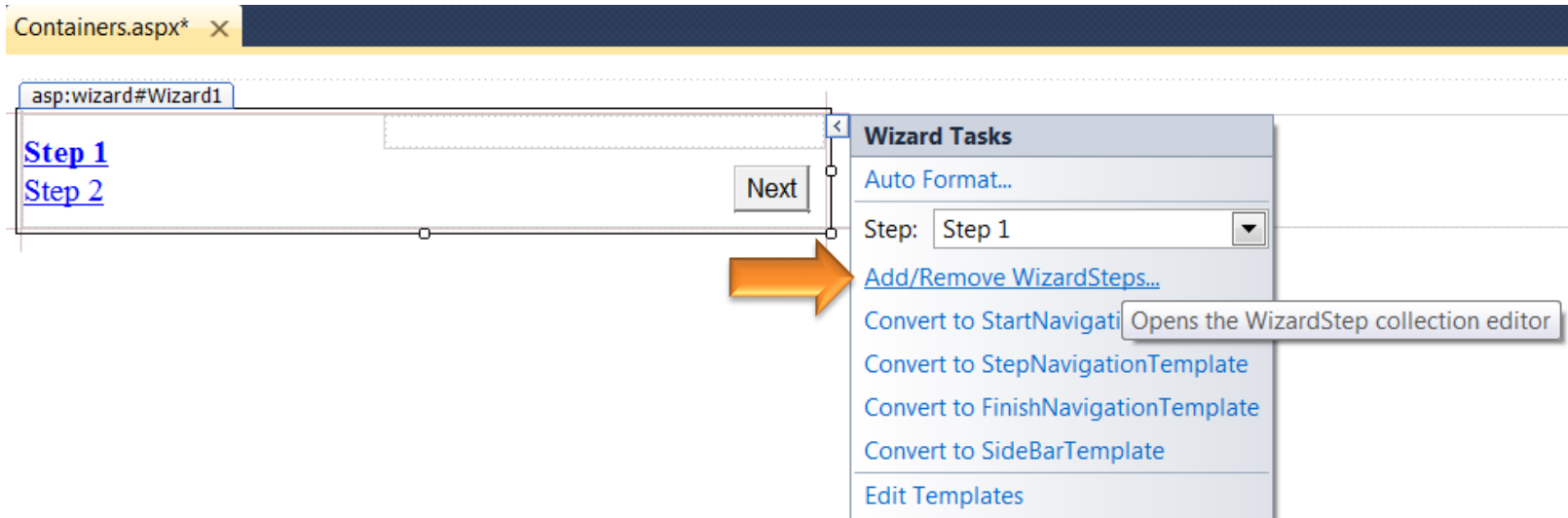
- In `Containers.aspx`:
 - Remove the text “**I am visible now**”
 - Drag a **Wizard** control inside the **Panel**
 - Drag its right edge further to the right, increasing the total width of the control to **500px**.



Example

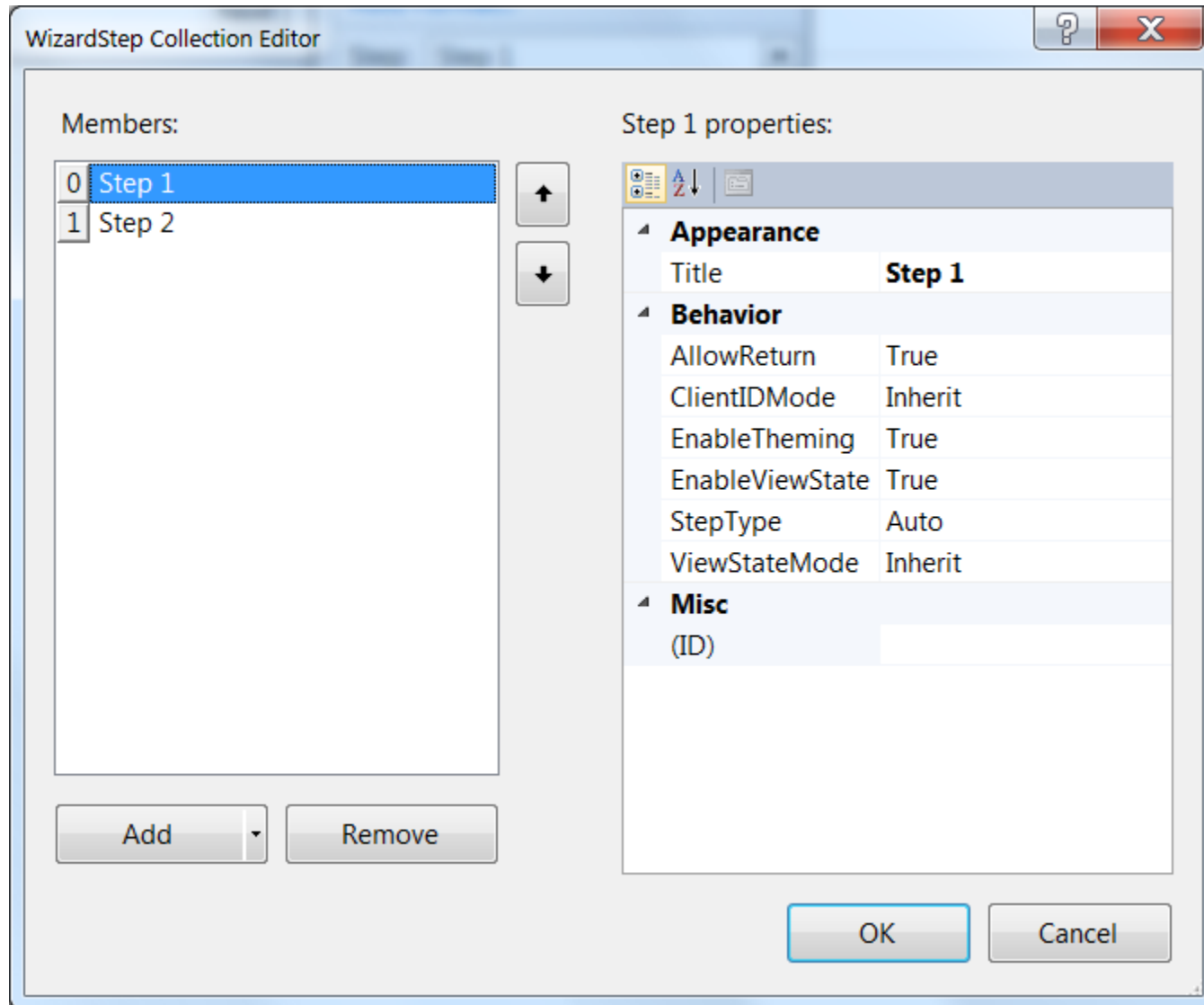
Wizard Control

- Open the Wizard's Smart Tasks panel

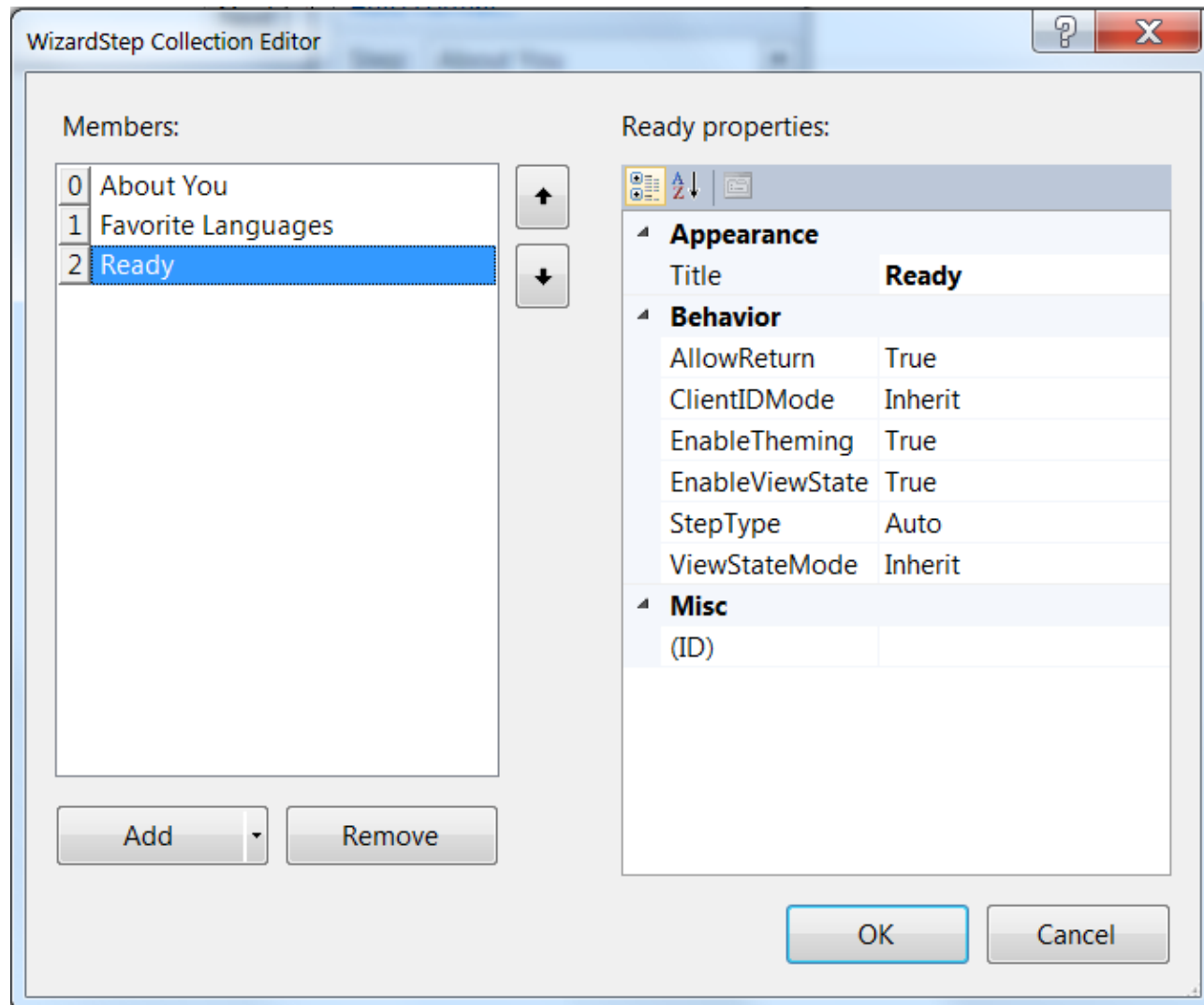


Example

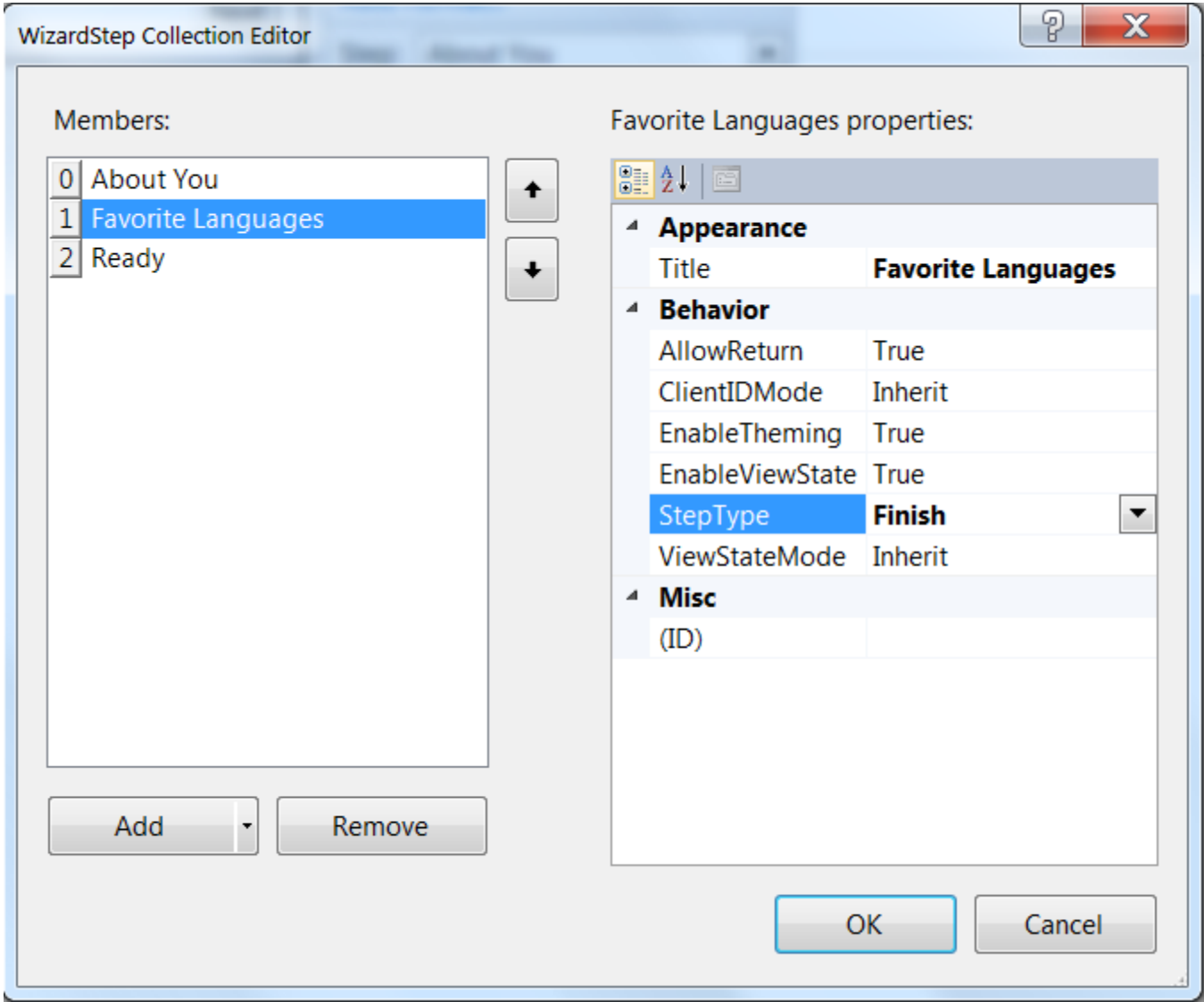
Wizard Control



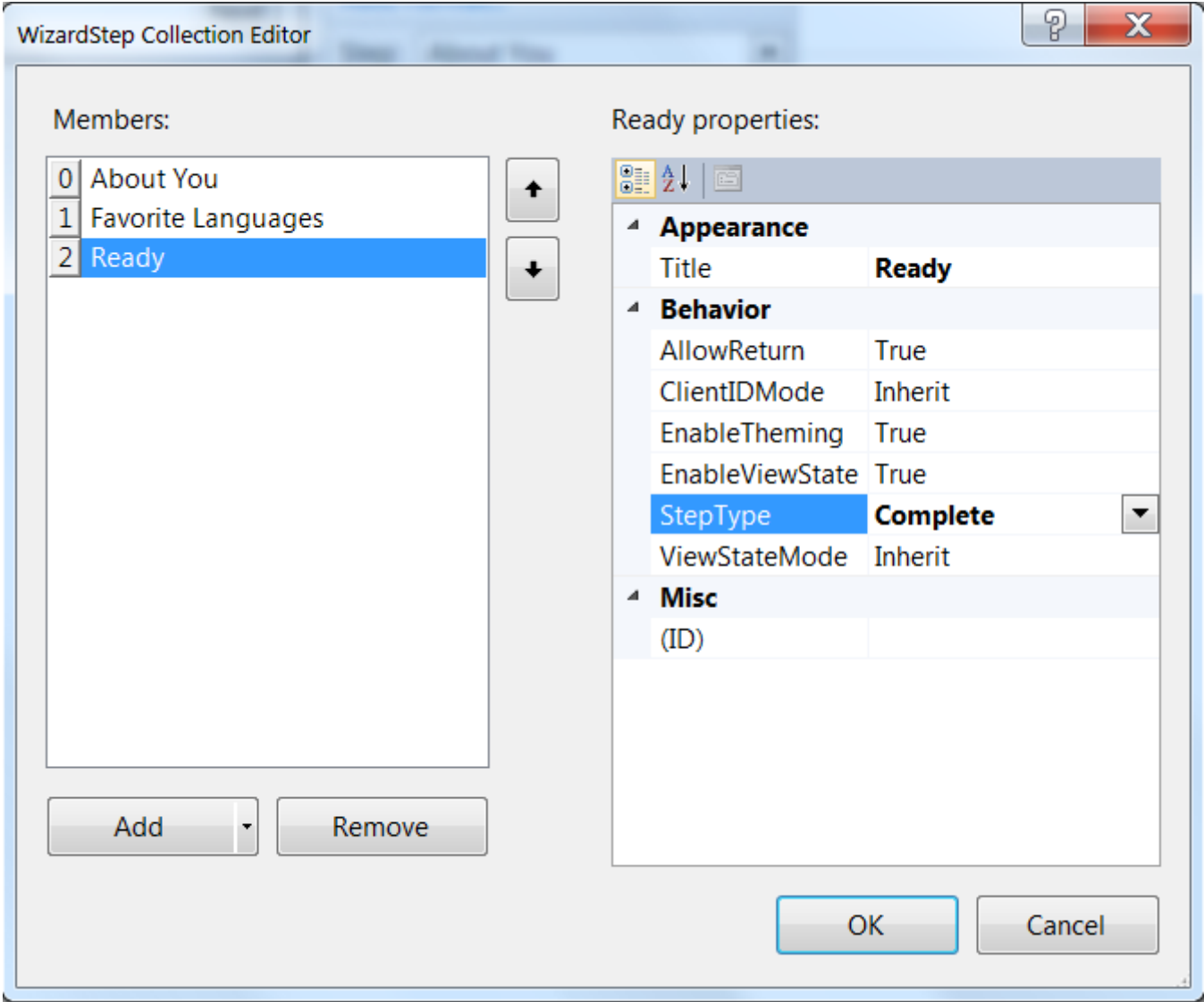
- Click the **Add** button to insert a third wizard step.
- Change the **Title** of **Members** list: **About You**, **Favorite Language** and **Ready**, respectively.



Change the **StepType** of the **Favorite Language** to **Finish**.



Change the **StepType** of the **Ready** to **Complete**.



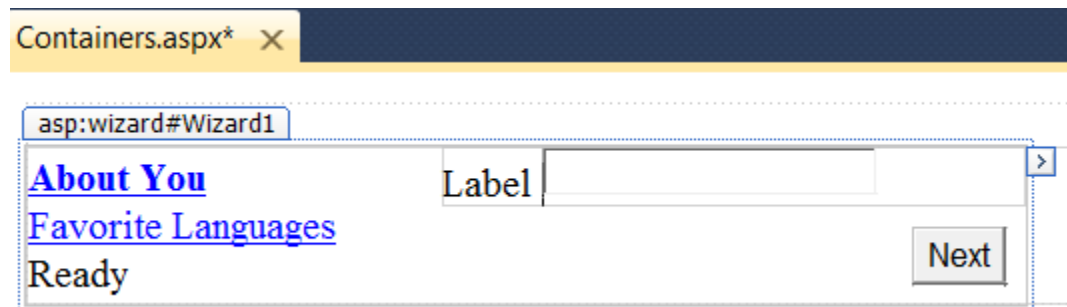
Example

Wizard Control

- Click **About You** to make it the active step

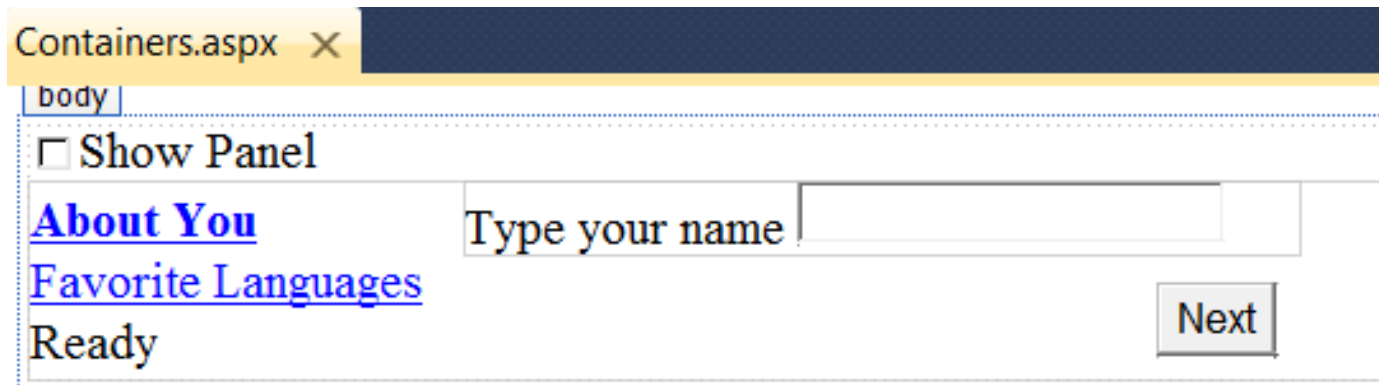


- Add a **Label** and a **TextBox** inside the grey rectangle.



Example Wizard Control

- Properties:
 - Label → Text = Type your name
 - TextBox → ID = yourNameTextBox

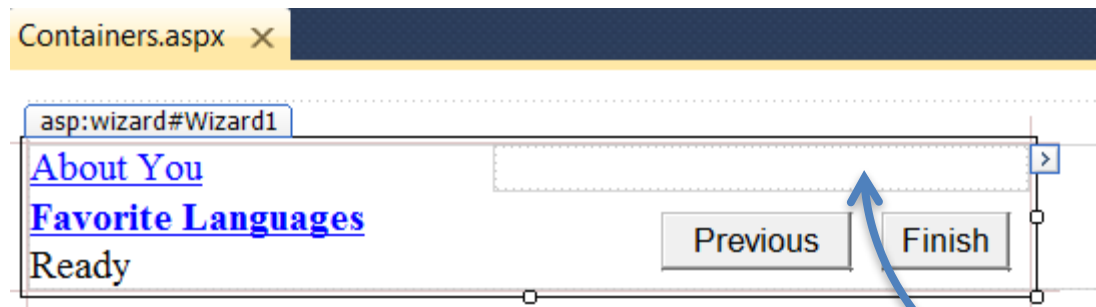


The screenshot shows a web browser window with the title bar 'Containers.aspx'. The main content area displays a wizard control. At the top, there is a checkbox labeled 'Show Panel'. Below this, the wizard has two steps: 'About You' and 'Favorite Languages'. The 'About You' step is currently active, showing a text input field with the placeholder text 'Type your name'. To the left of the input field are the step labels 'About You' and 'Favorite Languages', both underlined. Below the input field, the word 'Ready' is displayed. To the right of the input field, there is a 'Next' button.

Example

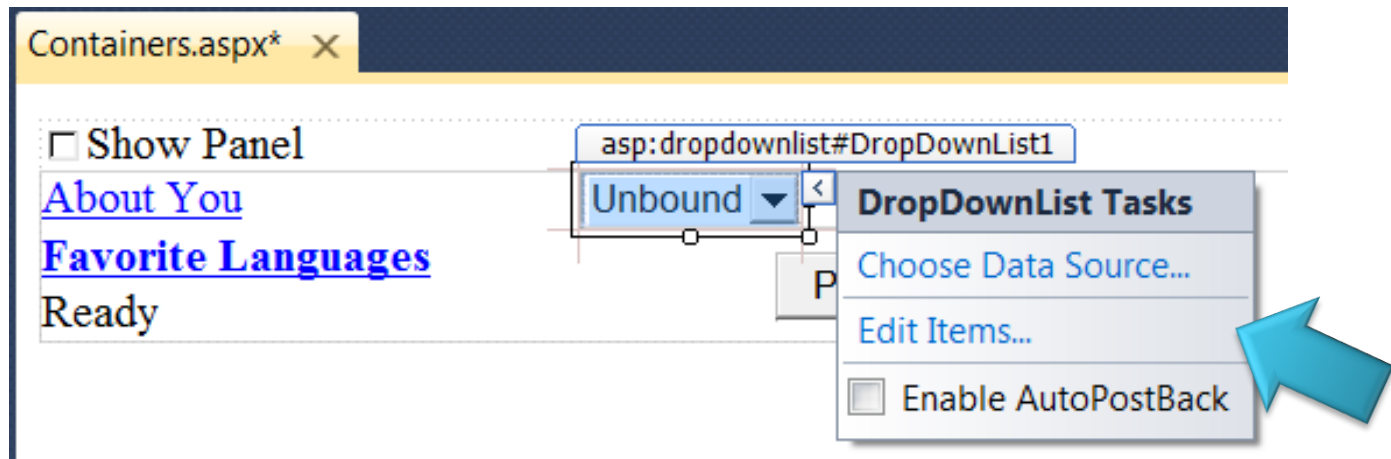
Wizard Control

- Click **Favorite Language** to make it the active step



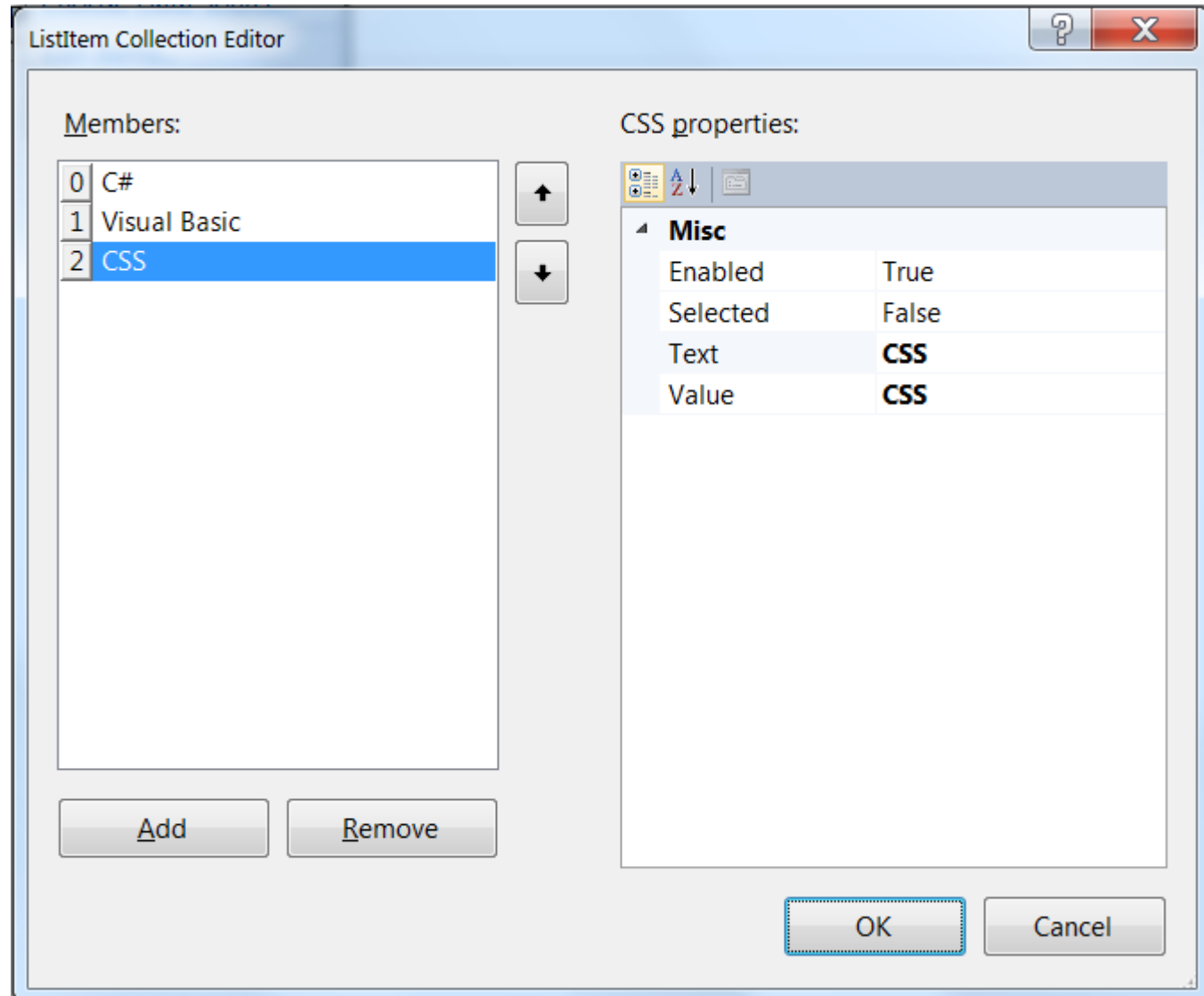
- Properties:
 - **ActiveStepIndex = 1**
- Add a **DropDownList** inside the grey rectangle.

Example Wizard Control



- Properties:
 - ID = favoriteLanguageDropDownList
- Choose Edit Items

Add 3 items: **C#**, **Visual Basic** and **CSS**



Example

Wizard Control

- In Source View:

```
<asp:WizardStep runat="server" StepType="Complete" Title="Ready">  
</asp:WizardStep>
```

- Drag a **Label** inside the last WizardStep labeled **Ready**.

– Change **Label** → **ID = Result**

```
<asp:WizardStep runat="server" StepType="Complete" Title="Ready">  
    <asp:Label ID="Result" runat="server" Text="Label"></asp:Label>  
</asp:WizardStep>
```


Example

Wizard Control

Double-click the Wizard **Finish** button

```
protected void Wizard1_FinishButtonClick(object sender, WizardNavigationEventArgs e)
{
    Result.Text = "Your name is " + yourNameTextBox.Text;
    Result.Text += "<br />Your favorite language is " + favoriteLanguageDropDownList.SelectedValue;
}
```

Example

Wizard Control

- View in browser

☐ Show Panel

☒ Show Panel

[About You](#)

[Favorite Languages](#)

Ready

Type your name

Next

☒ Show Panel

[About You](#)

[Favorite Languages](#)

Ready

C# ▼

Previous

Finish

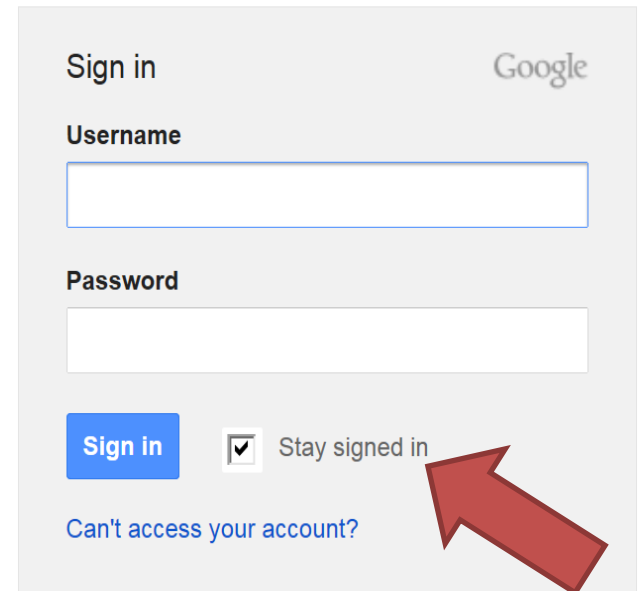
☒ Show Panel

Your name is Özgü

Your favorite language is Visual Basic

ASP.NET State Engine

- HTTP is *stateless*.
 - Web server **does not keep track of requests** that have been made from a specific browser.
- It's useful **if controls are able to maintain their own state**.
- The **state engine** in ASP.NET is capable of storing state for many controls.



Sign in Google

Username

Password

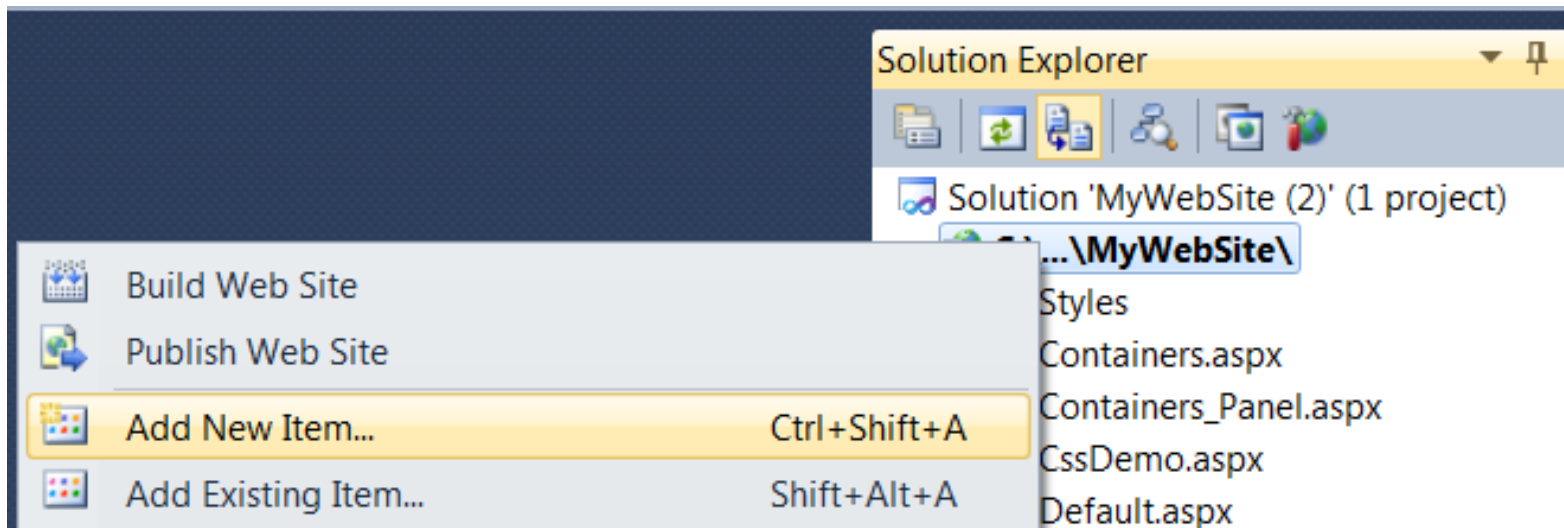
☒ Stay signed in

[Can't access your account?](#)

Example

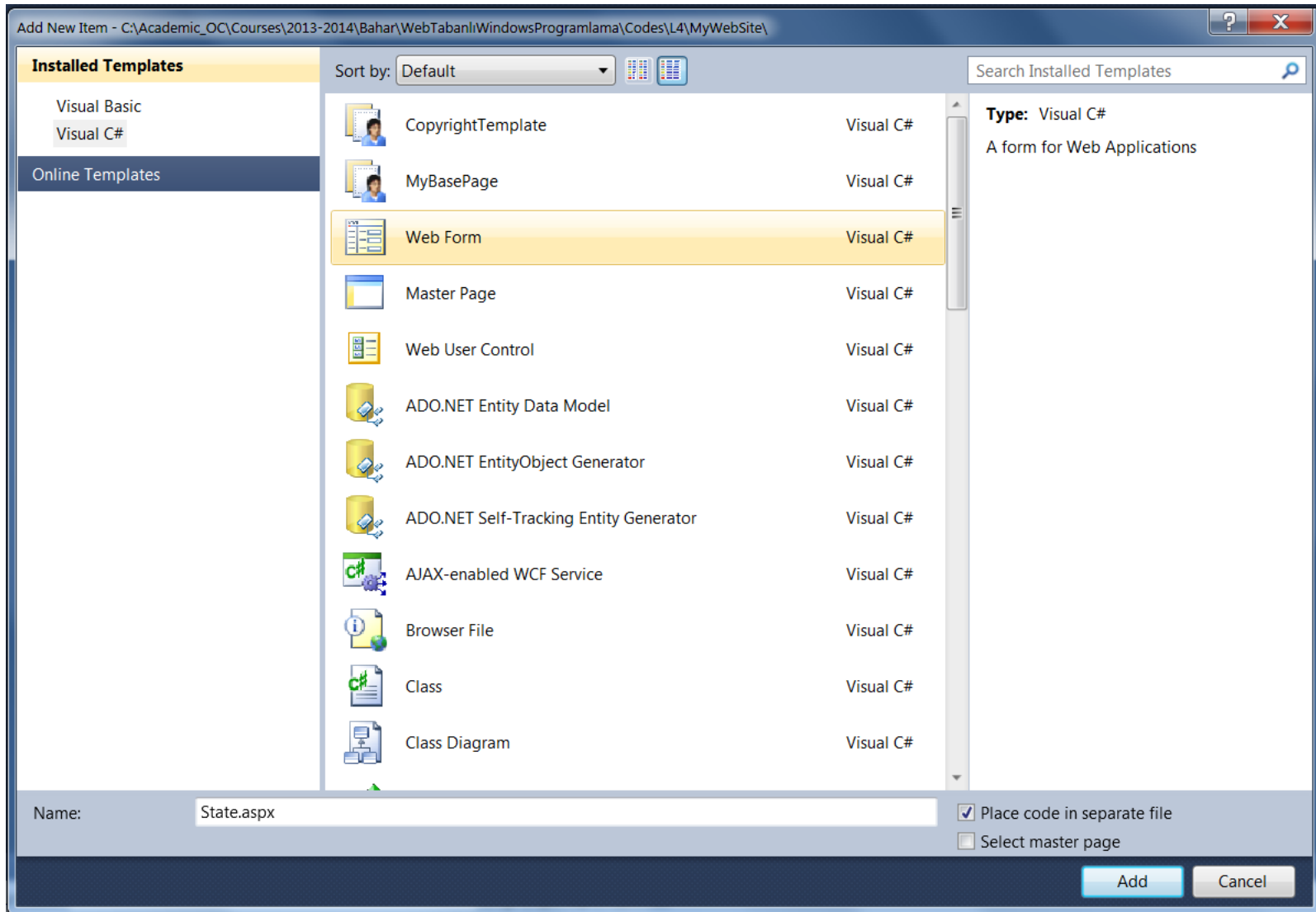
ASP.NET State Engine

- Add New Item in the Solution Explorer



Example

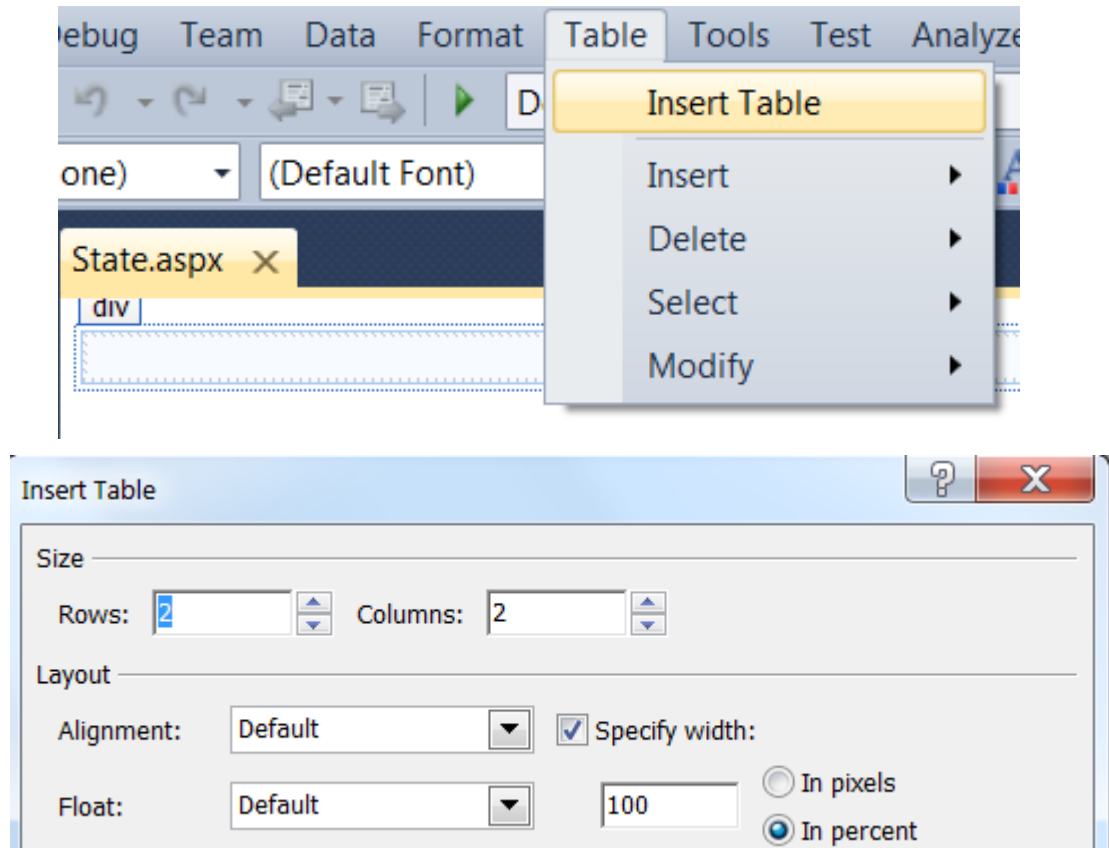
ASP.NET State Engine



Example

ASP.NET State Engine

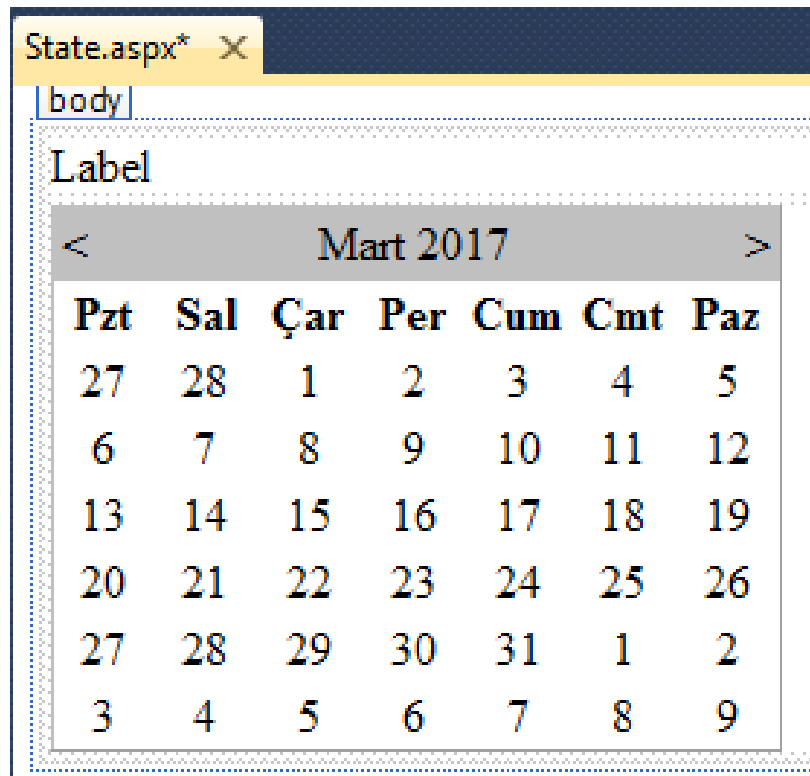
In Design View insert a table with 2 rows & 2 columns



Example

ASP.NET State Engine

- Drag a **Label** → First cell first row
- Drag a **Calendar** → First cell second row



The screenshot shows a web browser window titled "State.aspx*" with a "body" tab. Inside the browser, there is a "Label" control. Below the label is a calendar control for "Mart 2017". The calendar has a header with "<" and ">" navigation buttons. The main body of the calendar is a table with 7 columns: "Pzt", "Sal", "Çar", "Per", "Cum", "Cmt", and "Paz". The table contains 6 rows of dates. The first row shows dates 27, 28, 1, 2, 3, 4, 5. The second row shows 6, 7, 8, 9, 10, 11, 12. The third row shows 13, 14, 15, 16, 17, 18, 19. The fourth row shows 20, 21, 22, 23, 24, 25, 26. The fifth row shows 27, 28, 29, 30, 31, 1, 2. The sixth row shows 3, 4, 5, 6, 7, 8, 9.

Pzt	Sal	Çar	Per	Cum	Cmt	Paz
27	28	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	9

Example

ASP.NET State Engine

- Change the appearance of the calendar

The screenshot shows a web browser window with a tab titled "State.aspx* X". The main content area displays a calendar control with the title "Mart 2017". The calendar is a table with 7 columns and 6 rows. The columns are labeled "Pzt", "Sal", "Çar", "Per", "Cum", "Cmt", and "Paz". The rows contain dates from 27 to 9. To the right of the calendar is a "Calendar Tasks" panel with a link "Auto Format...". A tooltip "Set control formatting properties" is visible near the bottom right of the calendar.

State.aspx* X

asp:calendar#Calendar1

Mart 2017						
Pzt	Sal	Çar	Per	Cum	Cmt	Paz
27	28	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	9

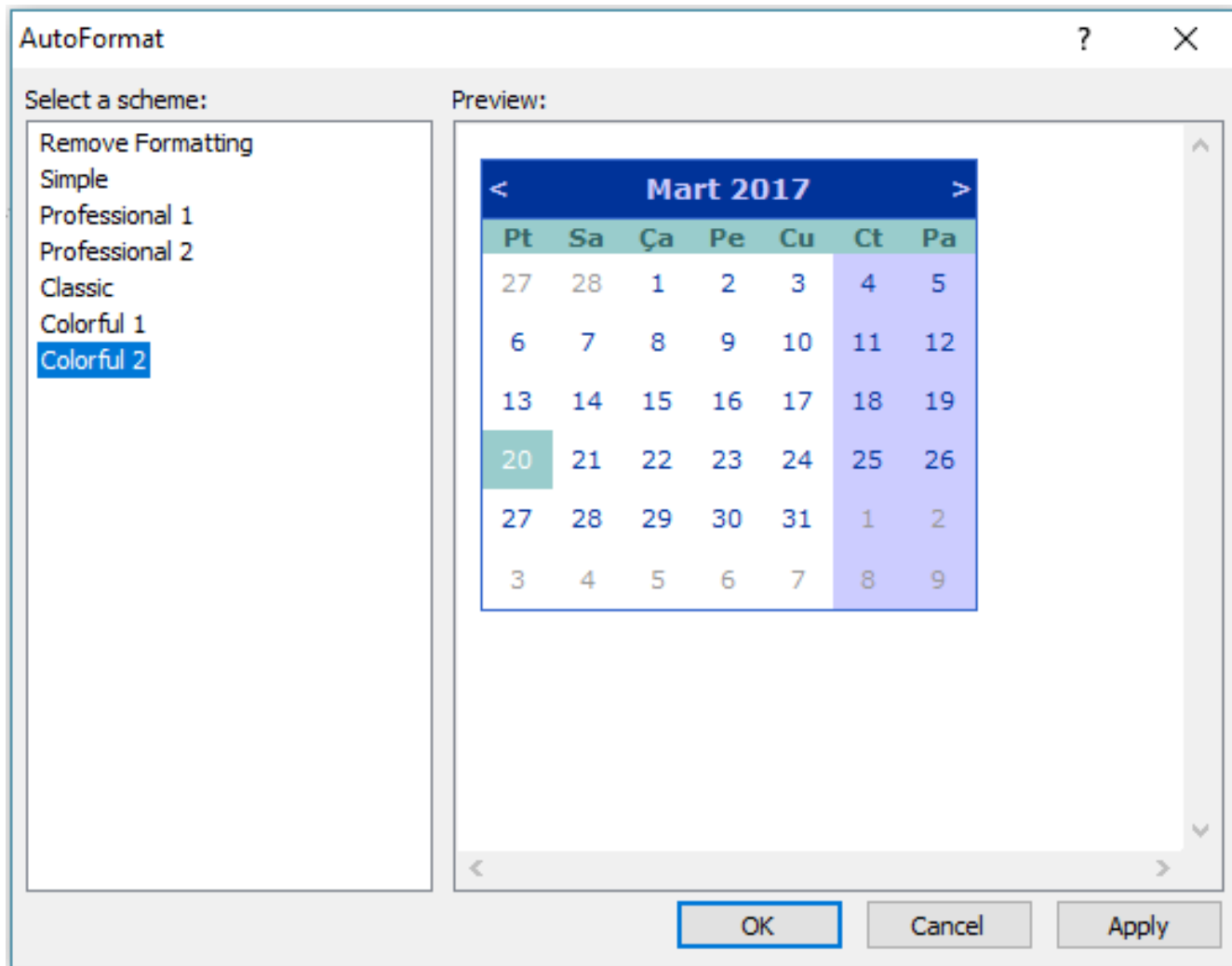
Calendar Tasks

[Auto Format...](#)

Set control formatting properties

Example

ASP.NET State Engine



Example

ASP.NET State Engine

- Drag a **Button** control to each of the 2 cells

State.aspx* X

body

Label

Mart 2017						
Pt	Sa	Ça	Pe	Cu	Ct	Pa
27	28	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	9

Button

Button

Example

ASP.NET State Engine

- Properties for the button in the 1st row:
 - ID = setDateButton
 - Text = Set Date
- Properties for the button in the 2nd row:
 - ID = plainPostBackButton
 - Text = Plain Postback

Example

ASP.NET State Engine

State.aspx X

body

Label

Mart 2017						
Pt	Sa	Ça	Pe	Cu	Ct	Pa
27	28	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	9

Set Date

Plain Postback

Example

ASP.NET State Engine

- Double-click **Set Date** button

```
protected void setDateButton_Click(object sender, EventArgs e)
{
    Label11.Text = DateTime.Now.ToString();
}
```

- View in browser

Label

< Mart 2017 >						
Pt	Sa	Ça	Pe	Cu	Ct	Pa
27	28	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	9

Set Date

Plain Postback

- Click **Set Date**

20.03.2017 11:49:38

≤ Mart 2017 ≥						
Pt	Sa	Ça	Pe	Cu	Ct	Pa
<u>27</u>	<u>28</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>	<u>9</u>

Set Date

Plain Postback

Notice that **Label** does not change when you click **Plain Postback**.

Example

ASP.NET State Engine

- **Label** Properties:
 - **EnableViewState = False**
 - Asp.NET runtime **does not track** the **Label** control anymore.

EnableViewState

- Gets or sets a value indicating **whether the server control persists its view state**, and the view state of any child controls it contains, to the requesting client.

- View in browser & Click **Set Date**

20.03.2017 11:50:35

≤ Mart 2017 ≥						
Pt	Sa	Ça	Pe	Cu	Ct	Pa
<u>27</u>	<u>28</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>	<u>9</u>

Set Date

Plain Postback

- Click **Plain Postback**

Label

≤ Mart 2017 ≥						
Pt	Sa	Ça	Pe	Cu	Ct	Pa
<u>27</u>	<u>28</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>	<u>9</u>

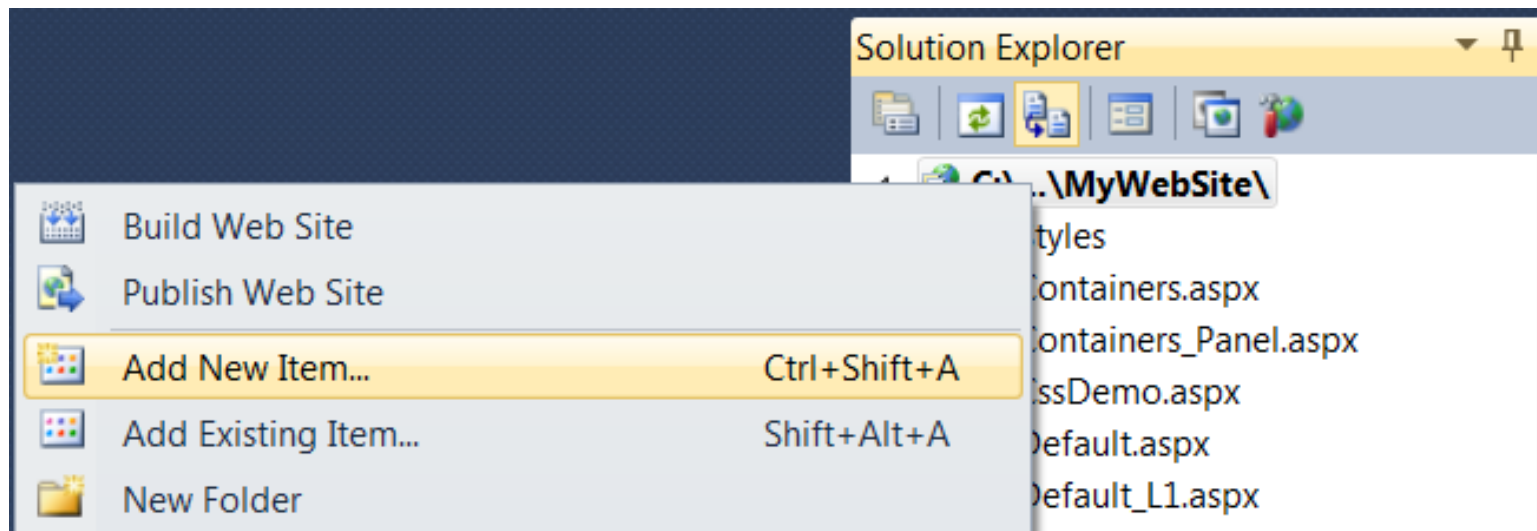
Set Date

Plain Postback

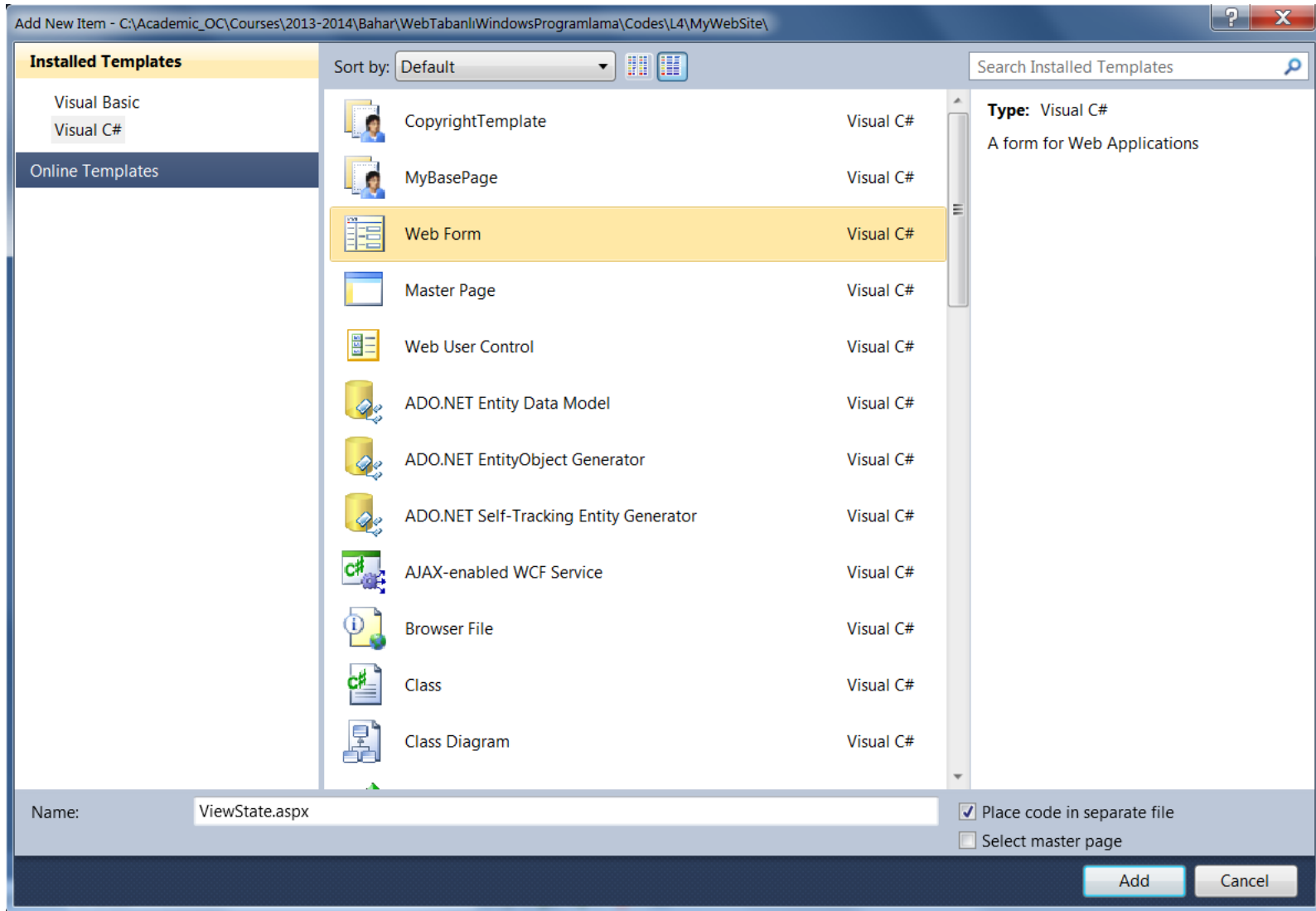
Example

ViewState

- Add New Item in the Solution Explorer



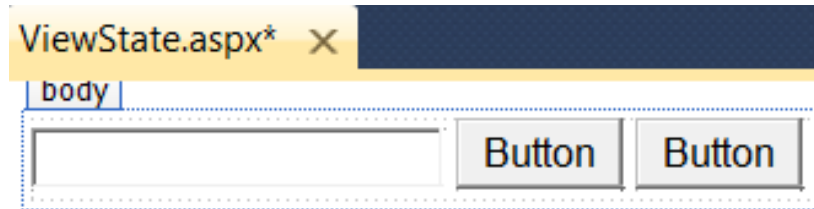
Example ViewState



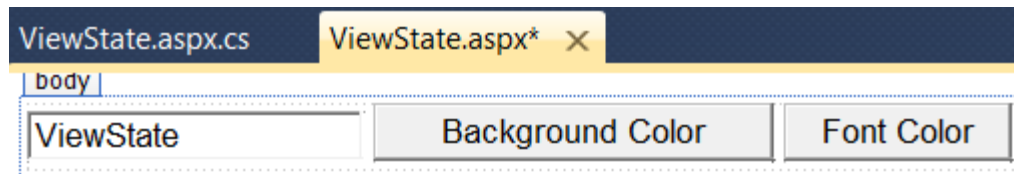
Example

ViewState

- Drag a **TextBox** and 2 **Button** control



- Properties:
 - **TextBox** → **Text = ViewState**
 - **Button1** → **ID = backgroundColorButton**
Text = Background Color
 - **Button2** → **ID = fontColorButton**
Text = Font Color



Example ViewState

- Double-click **Background Color** button

```
protected void backgroundColorButton_Click(object sender, EventArgs e)
{
    TextBox1.BackColor = System.Drawing.Color.Orange;
}
```

- Double-click **Font Color** button

```
protected void fontColorButton_Click(object sender, EventArgs e)
{
    TextBox1.ForeColor = System.Drawing.Color.Green;
}
```

Example ViewState

- Properties:
 - `TextBox` → `EnableStateView = True`
- View in browser

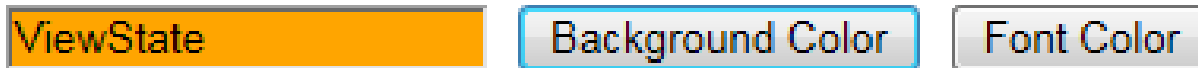
ViewState

Background Color

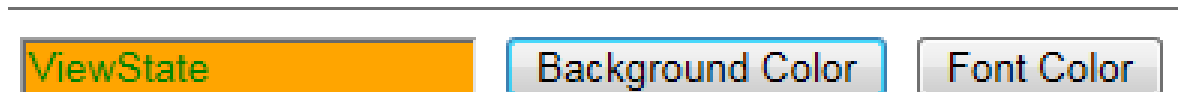
Font Color

Example ViewState

- Click **Background Color**



- Click **Font Color**



Example ViewState

- View source code

```
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE" value="/wEPDwUKLTc5NDc1NDA2MGRkgXj+tKVBdE9dbSzvduVa8muIjTPSxQB5rWi6nsi7QrU=" />
```

- After you click:

```
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE" value="/wEPDwUKLTc5NDc1NDA2MA9kFgICAw9kFgICAQ8PFgYeCUJhY2tDb2xvcgp/HgRfIVNCAgweCUZvcnVDb2xvcgpPZGRktFa/eiATMUwNUFY97X81nu1mdlKYHb3j7kQuADdTCA0=" />
```

Example ViewState

- Properties:
 - `TextBox` → `EnableStateView = False`
- View in browser

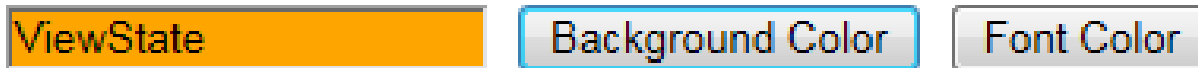
ViewState

Background Color

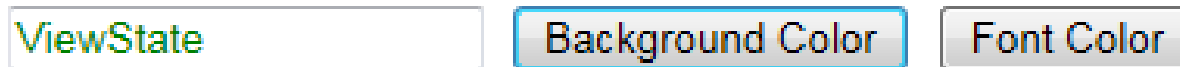
Font Color

Example ViewState

- Click **Background Color**



- Click **Font Color**



ViewState

- View State engine adds a **considerable amount of information** to the page.
 - Turn it off when you don't need it!
- This way, you can minimize **the size of the hidden __VIEWSTATE** field.
 - The page becomes **smaller** → The page is **loaded faster** in the browser.

ViewState

- **Turning off** View State:
 - At the **web site level**
 - At the **page level**
 - At the **control level**

ViewState

At the web site level

- You can do this in the **web.config** file in the root of the site by modifying the **<pages>** element under **<system.web>**, setting the **enableViewState** attribute to **false**:

```
<pages enableViewState="false">  
  ...  
</pages>
```

ViewState

At the page level

- In the page directive you can set **EnableViewState** to **False**:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="State.aspx.cs" Inherits="State" EnableViewState="false" %>
```

ViewState

At the control level

- Each ASP.NET Server Control enables you to set **EnableViewState** individually, giving you the option to turn it off for some controls, while leaving it on for others.



View State

- Once you've turned off View State at a higher level (**web.config** or **page level**) you can't turn it on again at a lower level (**the page** or a **specific control**).
- Do not turn off View State in the **web.config** file.

View State

- At the page level, set **EnableViewState** to **True** and **ViewStateMode** to **Disabled**:

```
<%@ Page Language="C#" ... EnableViewState="True" ViewStateMode="Disabled" %>
```

- This turns off **View State** for all controls in the page *except for those that explicitly enable it* again by setting the **ViewStateMode** to **Enabled**.

View State

- For the controls you want to give **View State** support, set the **ViewStateMode** to **Enabled**:

```
<asp:Label ID="Label1" runat="server" Text="Label" ViewStateMode="Enabled" />
```

Example

ASP.NET State Engine (Cont'd)

- Modify the page directive in **State.aspx**

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="State.aspx.cs" Inherits="State" %>
```

- **EnableViewState = True**
- **ViewStateMode = Disabled**



```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="State.aspx.cs" Inherits="State" EnableViewState="true" ViewStateMode="Disabled" %>
```

Example

ASP.NET State Engine (Cont'd)

- In Desing View → Drag a second **Label**

State.aspx* X

Label

asp:label#Label2

Label

Set Date

Plain Postback

≤	Mart 2017						≥
Pt	Sa	Ça	Pe	Cu	Ct	Pa	
<u>27</u>	<u>28</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>	<u>9</u>	

Example

ASP.NET State Engine (Cont'd)

- Enable the ViewStateMode for Label1

```
<asp:Label ID="Label1" runat="server" ViewStateMode="Enabled" Text="Label"></asp:Label>  
<br />  
<br />  
<asp:Label ID="Label2" runat="server" Text="Label"></asp:Label>
```

- Update setDateButton_Click

```
protected void setDateButton_Click(object sender, EventArgs e)  
{  
    Label1.Text = DateTime.Now.ToString();  
    Label2.Text = DateTime.Now.ToString();  
}
```

Example

ASP.NET State Engine (Cont'd)

- View in browser

Label

Label

Set Date

≤ Mart 2017 ≥						
Pt	Sa	Ça	Pe	Cu	Ct	Pa
<u>27</u>	<u>28</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>	<u>9</u>

Plain Postback

20.03.2017 11:52:15

20.03.2017 11:52:15

≤ Mart 2017 ≥						
Pt	Sa	Ça	Pe	Cu	Ct	Pa
<u>27</u>	<u>28</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>	<u>9</u>

Set Date

Plain Postback

Click **Plain Postback**

20.03.2017 11:52:15

Label

≤ Mart 2017 ≥						
Pt	Sa	Ça	Pe	Cu	Ct	Pa
<u>27</u>	<u>28</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>	<u>9</u>

Set Date

Plain Postback