



C#  
Programming



DR 3URIANI

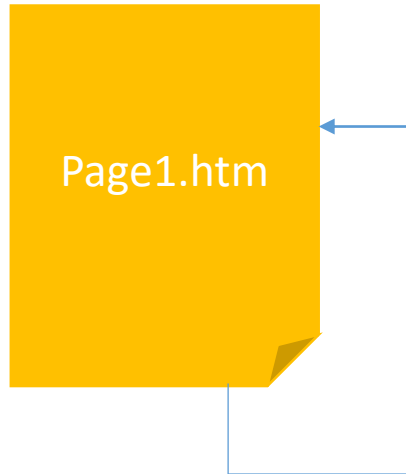
MUSTAFFA

AL - FATIHAH

CH 3 | WEB CONTROLS

- Displaying text using Literal and Label web controls
- Collecting Input using Text Boxes, Drop-Down Lists, Radio Buttons and Checkboxes

# POSTBACK VS. REDIRECT WEB FORM



The ACTION property is set to the URL of the page the form appears on (Page1.htm)



The ACTION property is set to Page2.htm

- Web controls can be accessed programmatically from the page's code.
- In this manner, Web controls serve as an intermediary between the source code and HTML portions of an ASP.NET page
- Categories of Web controls:
  - ✓ Web controls that are used to display text
  - ✓ Web controls that are used to collect user input
  - ✓ Web controls that are used to display data from database

Three types:

- Simple

Textbox, Button, Checkbox, DropDownList, Label, etc.

- Validation Controls

Validate user inputs

- Rich Controls

Calendar, navigation, ad-rotator ...

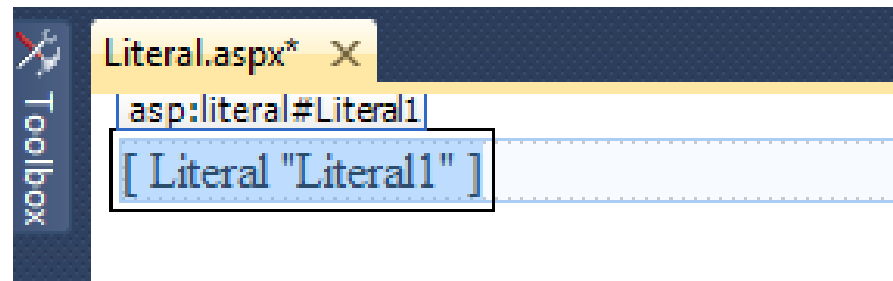
- Simplest
- Eight properties
- Does not contain any properties to specify the format of its output
- The HTML rendered by the Literal web control is precisely the value of its Text property
- Example 1: Text

“Welcome to ump!”

- Set property Text through property window or set it programmatically.
- Example 2:

```
CurrentTime.Text = DateTime.Now
```

- If you need to format the Literal control's output, you have to insert the appropriate HTML in the control's Text property.
- Example: `Text = "<i> Welcome to UMP</i>"`
- Set Mode attribute to "Encode", text not being converted to italic font, `<i>` tag are displayed instead.
- This is ideal if you want to display code in your application.



When the Text Property is not set, the Literal Web Control is shown in designer as [Literal "ID"]



# WEB CONTROLS FOR DISPLAYING TEXT | LITERAL

EG CH3 - Microsoft Visual Web Developer 2010 Express

File Edit View Website Build Debug Format Tools Window Help

Debug Any CPU

(New Inline Style) (None) (Default Font) (Default) B I U

Literal.aspx\*  
asp:literal#Literal1  
Hello BCS 2203

Solution Explorer

Solution 'EG CH3' (1 project)  
D:\... \EG CH3\  
Literal.aspx  
web.config

Properties

Literal1 System.Web.UI.WebControls.Literal

Appearance

Text Hello BCS 2203

Behavior

ClientIDMode Inherit  
EnableViewState True  
Mode Transform

Text

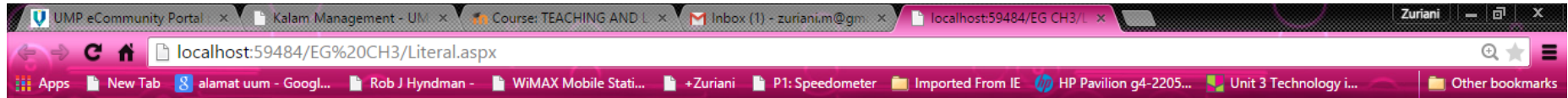
The text to be shown for the literal.

Design Split Source <html> <body> <form#form1> <div> <asp:Literal#Literal1>

Drag margin handles to resize margins. Press SHIFT or CTRL for more options.

12:47 PM 5/18/2015

# WEB CONTROLS FOR DISPLAYING TEXT | LITERAL

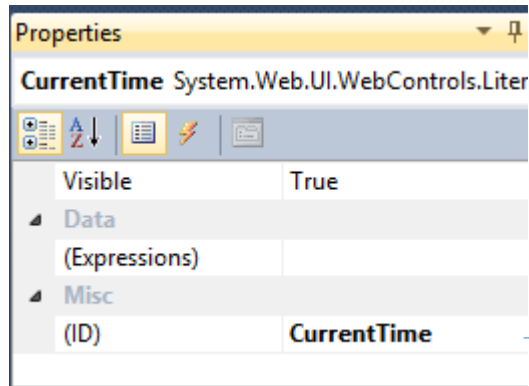


Hello BCS 2203

- How about if the value of the Text Property is dynamic?
- To set the Literal Control's Text Property programmatically, use the following syntax:

```
LiteralControlID.Text = Value;
```

Example: Display the Current Date and Time



1) Rename the Literal Web Control's ID property

- Enter the following code into the Page Load Event Handler:

```
CurrentTime.Text = DateTime.Now.ToString("T");
```

Literal Web Control's ID property

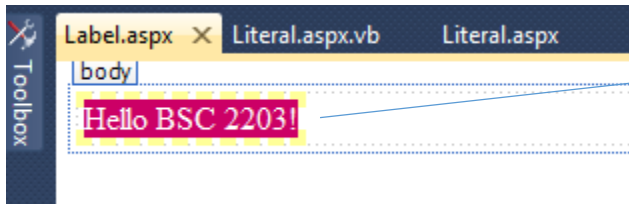


5/18/2015 1:05:40 PM

- Text property
- Used to display text in a set location on a web page
- Contains a number of formatting properties that, when set, specify how its Text property should be displayed in the user's browser.
- Formatting properties can be divided into the following classes:
  - ✓ Font properties, Color properties, Border properties, Miscellaneous properties.

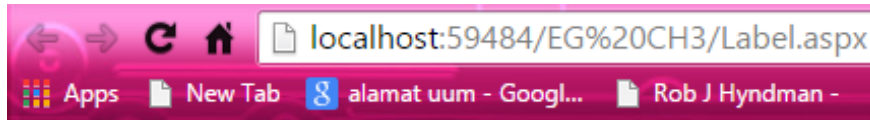
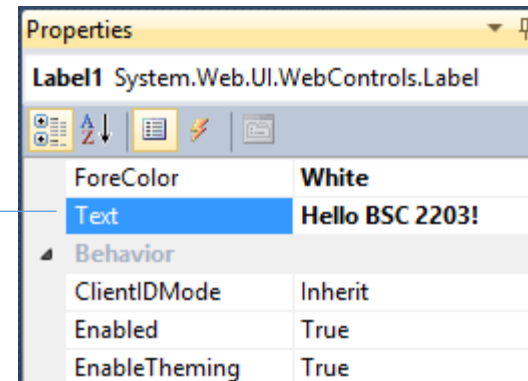
```
Label1.Text = "Welcome to ASP.net";
```

# WEB CONTROLS FOR DISPLAYING TEXT | LABEL



Play around with the formatting properties!  
ForeColor, border style, border color, background color

Set the Label Web Control's Text Property

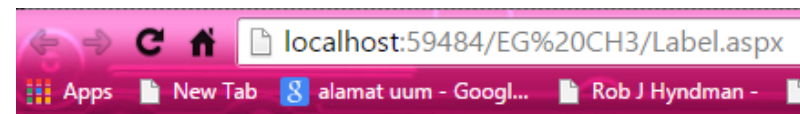


Hello BSC 2203!

- Set the Label Control's Text Property programmatically, type the following code into the [Page Load Event Handler](#):



```
Label1.Text =  
DateTime.Now.ToString("T");
```



5/18/2015 1:47:02 PM

## Literal

- To display plain text without any styling (we can't apply CSS styles for a Literal Control as it does not use any HTML tag).
- To display HTML content

## Label

- To set labels for the form fields
- Can change the fontcolor, background color, etc (CSS styles can be applied to Label control)- can easily be changed

# LITERAL vs. LABEL

LiteralVSLabel.aspx\* X

body

## Literal

*This is a Literal*  
*This is another Literal*

## Label

**This is a Label**  
This is another Label

```
<td style="text-align: center; font-style: italic; color: #0033CC; background-color: #FFFF00">  
  <asp:Literal ID="Literal1" runat="server" Text="This is a Literal"></asp:Literal>  
  <br />  
  <asp:Literal ID="Literal2" runat="server" Text="This is another Literal"></asp:Literal>  
</td>
```

```
<td style="text-align: center">  
  <asp:Label ID="Label1" runat="server"  
    style="color: #0000FF; background-color: #FFFF00" Text="This is a Label"></asp:Label>  
  <br />  
  <asp:Label ID="Label2" runat="server" Text="This is another Label"></asp:Label>  
</td>
```



# LITERAL vs. LABEL

Label		Literal
<i>This is a Label with White font color</i> This is another Label with Yellow font color		<i>This is a Literal</i> This is another <i>Literal</i>

```
<td>  
  <asp:Label ID="Label1" runat="server" ForeColor="White"  
    style="font-style: italic"></asp:Label>  
  <asp:Label ID="Label2" runat="server" ForeColor="Yellow"></asp:Label>  
</td>
```

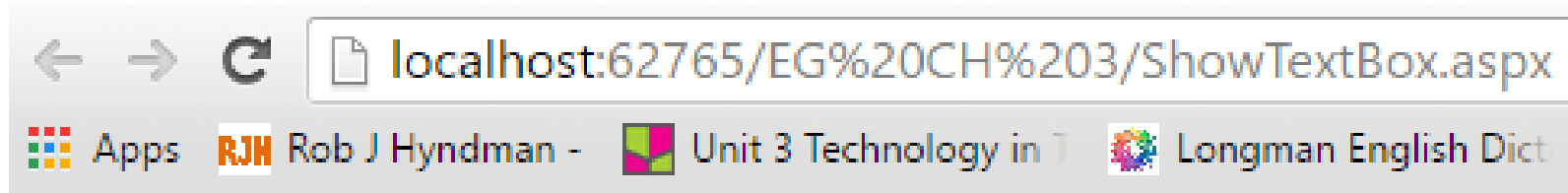
```
<td style="color: #FF0000; font-style: italic">  
  <asp:Literal ID="Literal1" runat="server"></asp:Literal>  
  <asp:Literal ID="Literal2" runat="server"></asp:Literal>  
</td>
```

- To collect input
- Can be used to accept user input
- By default Textbox comes with a single line of text
- You can also use the Textbox control to display a **multiline textbox** that masks user input by changing the value of the TextMode property to **TextBoxMode.Multiline**

SingleLine – Displays a single-line input field

MultiLine – Displays a multi-line input field

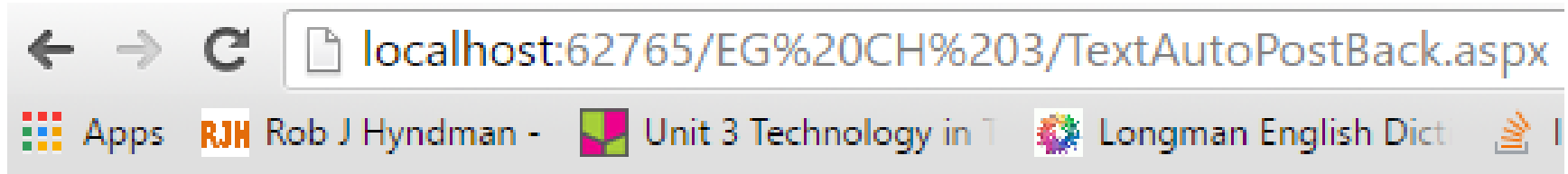
Password – Displays a single-line input field in which the text is hidden



Hello World!

.....

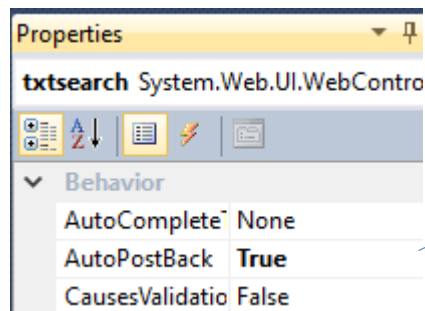
Hello World!



Search :

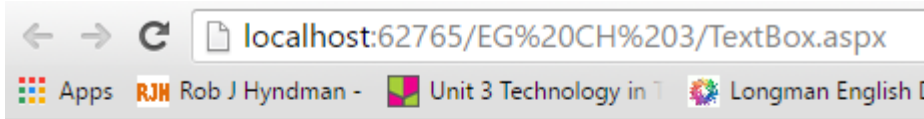
Search for : c#

```
protected void txtsearch_TextChanged(object sender, EventArgs e)
{
    lblresult.Text = "Search for : " + txtsearch.Text;
}
```

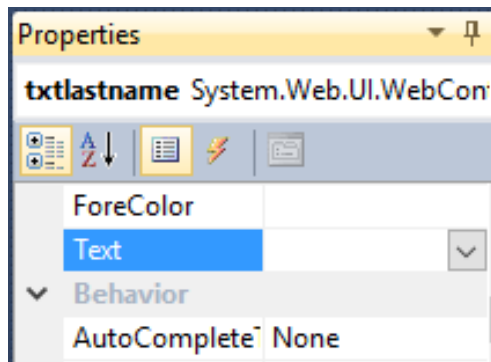


When the AutoPostBack property has the value True, the form containing the TextBox is automatically posted back to the server when the contents of the TextBox changes.

# WEB CONTROLS TO COLLECT USER INPUTS | TEXTBOX

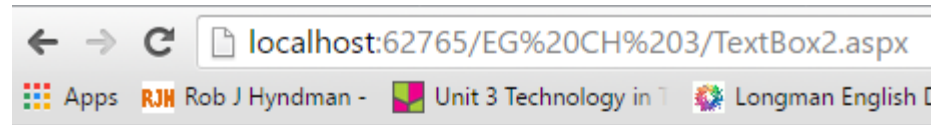


First Name	<input type="text" value="Muhammad"/>
Last Name	<input type="text" value="Abdullah"/>
	<input type="button" value="Submit"/>



When Autocomplete is enabled, the user does not need to re-enter common information, such as first name etc in a form field. If the user does not disabled AutoComplete in his browser, then his browser prompts him to enter the same value that he entered previously for the form field (even if the user entered the value for a form field at a different website)

```
protected void Button1_Click(object sender, EventArgs e)
{
    lblname.Text = "Welcome, " + txtfirstname.Text + txtlastname.Text;
}
```



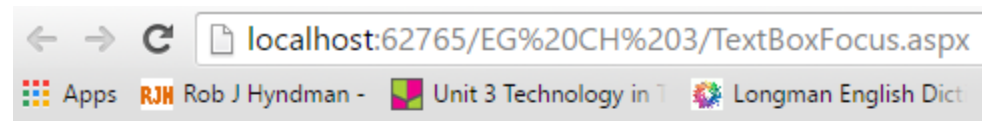
Name :

Comments :

Name : Nina Abdullah  
Comments : So far so good.

```
lblname.Text = " Name : ";  
lblname0.Text = txtname.Text;  
lblcomments.Text = " Comments : ";  
lblcomments0.Text = txtcomments.Text;
```

- TextBox control supports the Focus() method.
- You can use the Focus() method to shift the initial form focus to particular TextBox control.
- By default, no form field has focus when a page first opens.

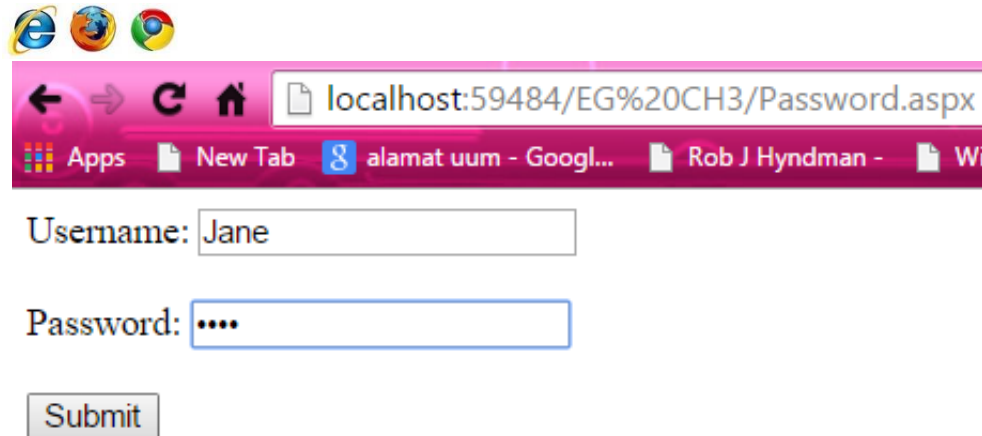


Name

Username

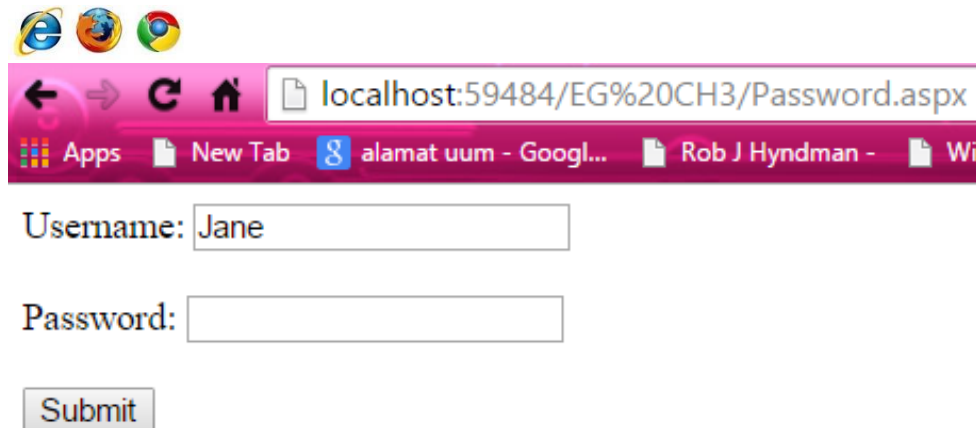
```
protected void Page_Load(object sender, EventArgs e)
{
    txtname.Focus();
}
```

After entered the required information, click the Submit button.



A screenshot of a web browser window. The address bar shows 'localhost:59484/EG%20CH3/Password.aspx'. The browser has several tabs open, including 'alamat uum - Googl...' and 'Rob J Hyndman - Wi'. The page content includes a 'Username:' label followed by a text box containing 'Jane', a 'Password:' label followed by a password box with four dots, and a 'Submit' button below them.

Password Text Box values are not remembered across postbacks

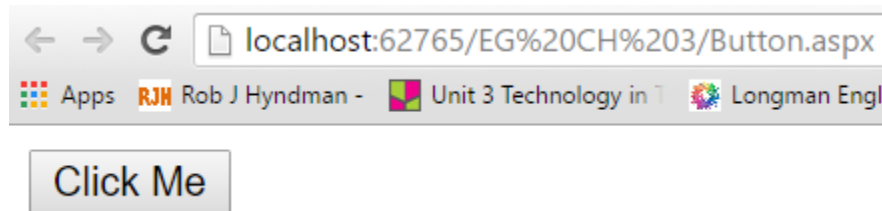


A screenshot of the same web browser window after a postback. The address bar and tabs remain the same. However, the 'Password:' text box is now empty, while the 'Username:' text box still contains 'Jane'. The 'Submit' button is still present below the fields.

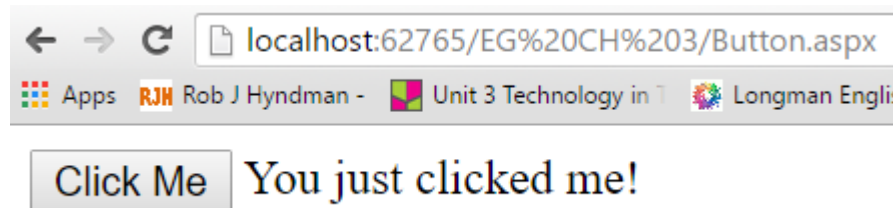


- Columns (Specifying the width)
- MaxLength (Limit the number of characters that can be entered into a text box)
  - ✓ Formatting Properties
  - ✓ Back color
  - ✓ BorderColor
  - ✓ BorderStyle
  - ✓ BorderWidth
  - ✓ Font
  - ✓ ForeColor

- Usually used to submit forms.
- OnClick Event

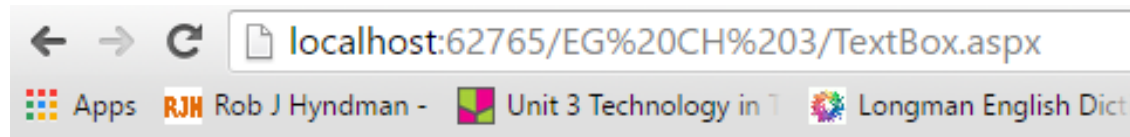


Pre



Post

- A variation of Button control
- It is the same except that the LinkButton control takes the form of a hyperlink (displays a hyperlink-style button control on a web page).
- By default, a LinkButton is a Submit button



First Name

Muhammad

Last Name

Abdullah

Submit

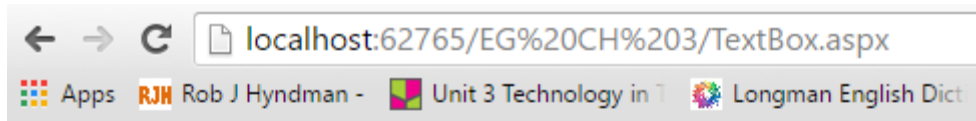
Button control

[Submit](#)

LinkButton control

Welcome, Muhammad Abdullah

- Another a variation of Button control
- Enable you to use a custom image as the form's button.
- **ImageUrl** property specify the picture's location.



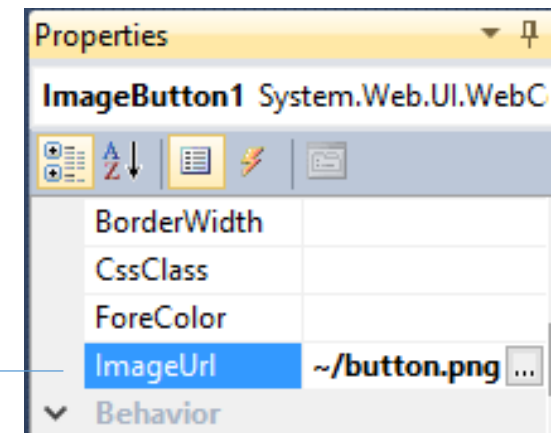
First Name

Last Name

[Submit](#)



Welcome, Muhammad Abdullah



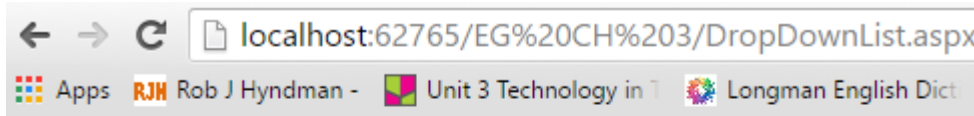
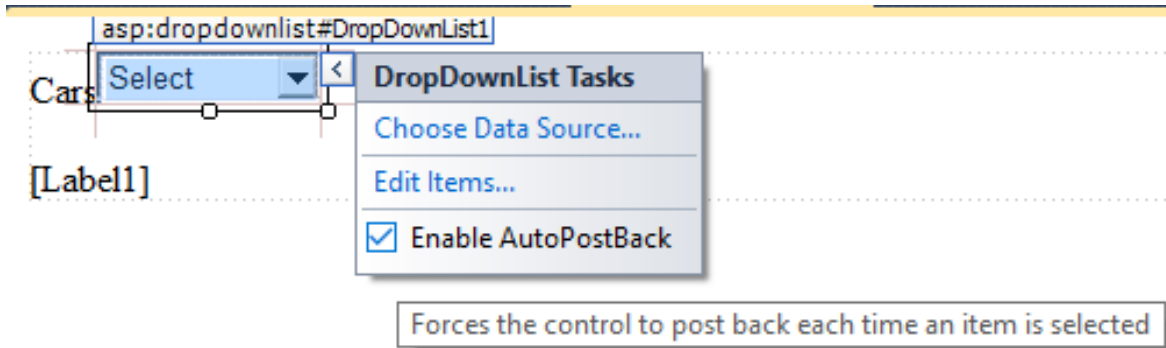
- Can contain any number of items
- Allows the user to select single item from the drop-down list
- Type in one after another or come from the database
- Items property. (Add)
- List items have four properties: Enabled, Selected, Text, and Value.
- Selected indicates whether the list item is the item that is selected by default when the web page loads.
- The syntax for the selected item:

```
DropDownListID.SelectedItem
```

- Example: `Result.Text = "You like" & Flavors.SelectedItem.Text`
- A default ListItem can be added to a DropDownList **programmatically** with the following syntax after binding data to the DropDownList:

```
DropDownList1.Items.Add("Pepperoni")
```

## AutoPostBack



Cars

You selected : Subaru

```
protected void Page_Load(object sender, EventArgs e)
{
    Label1.Text = "You selected : " + DropDownList1.Text;
}
```

DropDownList2.aspx\* × DropDownList.aspx.vb\* DropDownList.aspx

asp:dropdownlist#DropDownList1

Unbound < DropDownList Tasks

- Choose Data Source...
- Edit Items...
- ☐ Enable AutoPostBack

ListItem Collection Editor

Members:

0	Pepperoni
1	Onion
2	Beef

↑

↓

Add Remove

Beef properties:

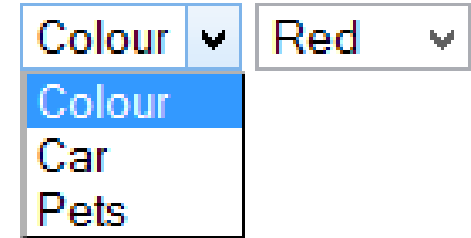
Misc	
Enabled	True
Selected	False
Text	Beef
Value	Beef

OK Cancel

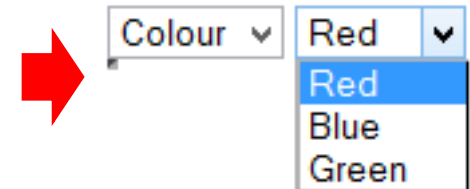
## 2 DROPDOWNLIST TO COLLECT INPUT

Create a web page containing two Drop-Down List

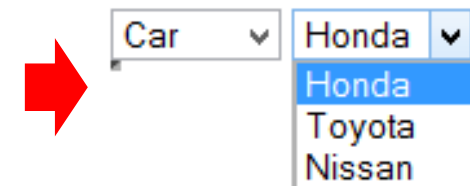
- Add 2 Drop-Down List.
- For DropDownList1, add the following items: Colour, Car and Pets.
- For DropDownList2, it will give the following selections.
- Example, when Colour (from DropDownList1) is selected, the DropDownList2 will load/give the following selections
- When Car (from DropDownList1) is selected, the DropDownList2 will load/give the following selections
- When Pets (from DropDownList1) is selected, the DropDownList2 will load/give the following selections



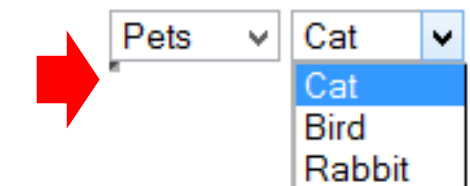
Initial state of the two dropdown lists. The first dropdown list is labeled 'Colour' and has a dropdown arrow. The second dropdown list is labeled 'Red' and has a dropdown arrow.



When 'Colour' is selected in the first dropdown list, the second dropdown list shows the following selections: Red, Blue, and Green. A red arrow points from the initial state to this state.



When 'Car' is selected in the first dropdown list, the second dropdown list shows the following selections: Honda, Toyota, and Nissan. A red arrow points from the initial state to this state.



When 'Pets' is selected in the first dropdown list, the second dropdown list shows the following selections: Cat, Bird, and Rabbit. A red arrow points from the initial state to this state.



- Radio button allows user to choose only one of a predefined set of options
- When a user clicks on a radio button, it becomes checked and all other radio buttons with same group become unchecked.
- The buttons are grouped logically if they all share the same GroupName property
- Set Text Property
- Set GroupName Property
- Determining which RadioButton was selected.
- **checked** property
- Example:

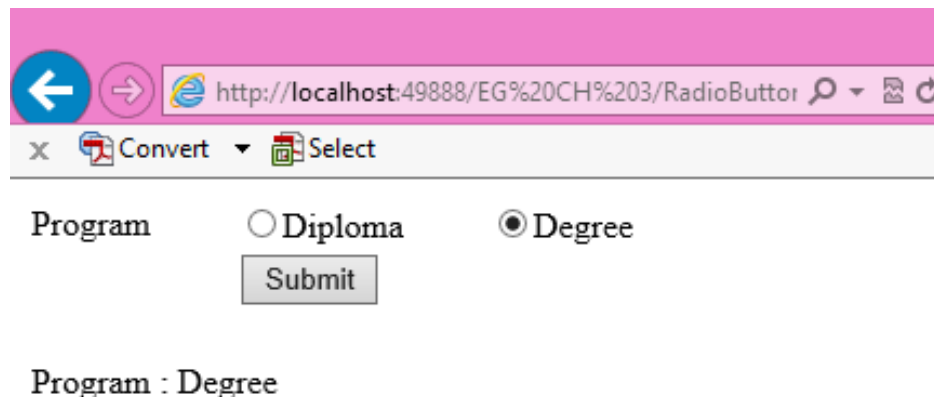
```
if vanilla.checked then
{
    Results.Text = "You like Vanilla";
}
```

**C#**

```
RadioButton1.Checked = True
```

```
if (RBProgDiploma.Checked == true)
{
    LblProgram.Text = "Program : " + RBProgDiploma.Text;
}
else
{
    LblProgram.Text = "Program : " + RBProgDegree.Text;
}
```

RadioButton.aspx.cs



Program ☐ Diploma ☒ Degree

Submit

Program : Degree

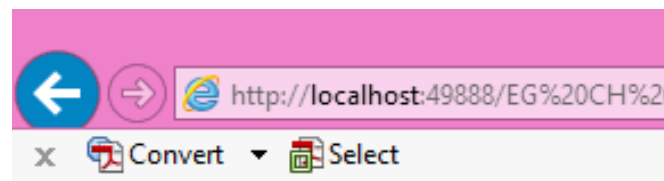
RadioButton.aspx

\*\*\*Don't forget to set the GroupName property!

- Used to create a group of radio buttons
- Each selectable item in a RadioButtonList control is defined by ListItem element

```
protected void Button1_Click(object sender, EventArgs e)
{
    Label1.Text = "You have chosen " + RadioButtonList1.SelectedItem.Text;
}
```

RadioButtonList.aspx.cs



You have chosen Sony

RadioButtonList.aspx

- Allow the user to make multiple selections from a number of options
- Checkboxes comes with a caption, which you can set in the Text Property

```
CheckBox1.Text = "CheckBox";
```

- CheckBoxID.checked
- Example:

```
if vanilla.checked then  
{  
    Results.Text + "You like Vanilla";  
}
```

```
protected void Button1_Click(object sender, EventArgs e)
{
    if (CBOption1.Checked == true)
    {
        Label1.Text = "You have chosen " + CBOption1.Text;
        Label1.ForeColor = System.Drawing.Color.Green;
    }
    if (CBOption2.Checked == true)
    {
        Label1.Text = "You have chosen " + CBOption2.Text;
        Label1.ForeColor = System.Drawing.Color.Blue;
    }
    if (CBOption1.Checked == true && CBOption2.Checked == true)
    {
        Label1.Text = "You have chosen " + CBOption1.Text + " and " + CBOption2.Text;
        Label1.ForeColor = System.Drawing.Color.Crimson;
    }
}
```

CheckBox.aspx.cs

☒ Option 1 ☐ Option 2

Submit

You have chosen Option 1

☒ Option 1 ☒ Option 2

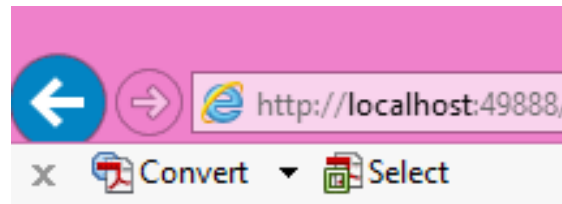
Submit

You have chosen Option 1 and Option 2

- Used to create a multi-selection check box group.
- Steps:
  - ✓ Place a CheckboxList control
  - ✓ Add the items with the ListItem template into the CheckBoxLayout
  - ✓ Place a Button control
  - ✓ Add a Literal and ListBox control below the Button control
  - ✓ Setting the “Visible” attribute to false allows the text invisible until the button is clicked.

```
protected void Button1_Click(object sender, EventArgs e)
{
    foreach (ListItem mylist in CheckBoxList1.Items)
    {
        if (mylist.Selected == true)
        {
            Label1.Text += mylist.Text + "<br/>";
        }
    }
}
```

CheckBoxList.aspx.cs



Choose your favourite color(s):

☐ Black

☒ Red

☒ White

Submit

Red  
White



CheckBoxList:

- ☒ Red
- ☒ Blue
- ☐ Green

RadioButtonList:

- ☐ Red
- ☐ Blue
- ☐ Green

Submit

Red  
Blue

```
protected void Button1_Click(object sender, EventArgs e)
{
    int i;
    String myString="";

    for (i=0; i<CheckBoxList1.Items.Count; i++)
    {
        if (CheckBoxList1.Items[i].Selected)
        {
            myString = myString + CheckBoxList1.Items[i].Text + "</br>";
        }
    }
    Label1.Text = myString;
}
```

CBL RBL.aspx.cs





TO BE  
CONTINUED