

Data Binding

Today you will learn

- Introduction to Data Binding
- Types of ASP.NET Data Binding
- Single-value Data Binding
- Repeated-Value Data Binding

Introduction to Data Binding

- **Data binding** involves creating a direct connection between a data source and a control in an application window.
- ASP.NET data binding works in one direction only.
- Information moves from a data object into a control. Then the data objects are thrown away, and the page is sent to the client.
- If the user modifies the data in a data-bound control, your program can update the corresponding record in the database, but nothing happens automatically.

Types of ASP.NET Data Binding

- Two types of ASP.NET data binding exist:
 - single-value binding
 - repeated-value binding.

Single-Value, or “Simple,” Data Binding

- Add special data binding expressions into your .aspx files.
`<%# expression_goes_here %>`
 - Example: `<%# Country %>`
`<%# Request.Browser.Browser %>`
- Add `DataBind()` method in .aspx.cs file under any class events:
`this.DataBind()`

Single-Value Data Binding

Count number of transaction and display browser name

Step 01: add expression in .aspx file

```
<asp:Label id="lblDynamic" runat="server" Font-Size="X-Large" >  
There were <%# TransactionCount %> transactions today.  
I see that you are using <%# Request.Browser.Browser %>.  
</asp:Label>
```

Step 02: declare variable as TransactionCount in .aspx.cs file

```
protected int TransactionCount;
```

Step 03: Assign the value to TransactionCount = 10

```
TransactionCount = 10;
```

Step 04: Convert all data bind expressions

```
this.DataBind();
```

Simple Data Binding

Display URL= Images/picture.jpg for label, checkbox, hyperlink and Image controllers using simple data binding.

Step 01: add expression in .aspx file

```
<asp:Label id="lblDynamic" runat="server"><%# URL %></asp:Label>
<asp:CheckBox id="chkDynamic" Text="<%# URL %>" runat="server" />
<asp:Hyperlink id="lnkDynamic" Text="Click here!" NavigateUrl="<%# URL %>"
  runat="server" />
<asp:Image id="imgDynamic" ImageUrl="<%# URL %>" runat="server" />
```

Step 02: declare variable as URL in .aspx.cs file

```
protected string URL;
```

Step 03: Assign the value to URL = Images/picture.jpg

```
URL = "Images/picture.jpg"
```

Step 04: Convert all data bind expressions

```
this.DataBind();
```

Problems with Single-Value Data Binding

- Putting code into a page's user interface
- Fragmenting code

Using Code Instead of Simple Data Binding

```
protected void Page_Load(Object sender, EventArgs e)
{
    TransactionCount = 10;
    lblDynamic.Text = "There were " + TransactionCount.ToString();
    lblDynamic.Text += " transactions today. ";
    lblDynamic.Text += "I see that you are using " + Request.Browser.Browser;
}
```

Repeated-Value Data Binding

- Repeated-value data binding works with the ASP.NET list controls.
- To use repeated-value binding, you link one of these controls to a data source (such as a field in a data table).
- When you call `DataBind()`, the control automatically creates a full list using all the corresponding values.
- Some of the list controllers are:
 - `ListBox`, `DropDownList`, `CheckBoxList`, and `RadioButtonList`
 - `HtmlSelect` (`Select`)
 - `GridView`, `DetailsView`, `FormView`, and `Listview`

Data Binding with Simple List Controls

- Data Binding with simple list control follows only three steps:
 - Create and fill some kind of data object.

```
List<string> fruit = new List<string>();  
fruit.Add("Kiwi");  
fruit.Add("Pear");  
fruit.Add("Mango");  
fruit.Add("Blueberry");  
fruit.Add("Apricot");  
fruit.Add("Banana");  
fruit.Add("Peach");  
fruit.Add("Plum");
```

- Link the object to the appropriate control.

```
lstItems.DataSource = fruit
```

- Activate the binding.

```
this.DataBind();
```


Multiple Binding

- You can bind the same data list object to multiple different controls.

- Step 01: Create and fill collection

```
List<string> fruit = new List<string>();  
fruit.Add("Kiwi");  
fruit.Add("Pear");  
fruit.Add("Mango");  
fruit.Add("Blueberry");  
fruit.Add("Apricot");  
fruit.Add("Banana");  
fruit.Add("Peach");  
fruit.Add("Plum");
```

- Step 02: Define the binding for list controllers

```
MyListBox.DataSource = fruit  
MyDropDownListBox.DataSource = fruit  
MyHtmlSelect.DataSource = fruit  
MyCheckBoxList.DataSource = fruit  
MyRadioButtonList.DataSource = fruit
```

Multiple Binding

- Step 03: Activate Binding `this.DataBind();`

Data Binding with a Dictionary Collection

- A **dictionary** collection is a special kind of collection in which every item (or definition, to use the dictionary analogy) is indexed with a specific key (or dictionary word).
- There are two basic dictionary-style collections in .NET:
 - The Hashtable collection (in the System.Collections namespace)
 - The Dictionary collection (in the System.Collections.Generic namespace)

Data Binding with a Dictionary Collection

Step 01: `// Use integers to index each item. Each item is a string.
Dictionary<int, string> fruit = new Dictionary<int, string>();`

```
fruit.Add(1, "Kiwi");  
fruit.Add(2, "Pear");  
fruit.Add(3, "Mango");  
fruit.Add(4, "Blueberry");  
fruit.Add(5, "Apricot");  
fruit.Add(6, "Banana");  
fruit.Add(7, "Peach");  
fruit.Add(8, "Plum");
```

Step 02: `MyListBox.DataSource = fruit`

Step 03: `MyListBox.DataTextField = "Value"`

Step 04: `this.DataBind();`

- Each item in a dictionary-style collection has both a key and a value associated with it.

Data Binding with a Dictionary Collection

- If you don't specify which property you want to display, ASP.NET simply calls the ToString() method on each collection item.
- To retrieve the key from dictionary collection use property called DataValueField
- Example:

```
MyListBox.DataTextField = "Value"  
MyListBox.DataValueField = "Key"
```
- To retrieve the key from <select> tag use property called: SelectedItem.Value
- Example:

```
MyListBox.SelectedItem.Value
```

END OF LECTURE