Data Binding

These notes introduce data binding. We will consider this further when we cover database access.

Data Binding

1. Many server controls have a feature that allows the data to be displayed in the control (*e.g.* a DropDownList) to be *bound* directly to a *data source,* like a database. *Data-bound* controls have this property:

   DataSource – Gets or sets the object from which a data-bound control retrieves its list of data items. This property can only be assigned at run-time.

   and this method:

   DataBind() – Binds a data source to a control

   Note: there is also a *DataSourceID* property that is the *id* of a data source component. We will use this in the next topic (database), but not here.[1]

2. We will consider binding to a database later in the semester. For now, we consider a simple example of binding an array of ListItems to a ListBox.

```
ListItem[] lis = new ListItem[11];
lis[0] = new ListItem("Jones, Ren", "23");
lis[1] = new ListItem("Maida, Stew", "4");
lis[2] = new ListItem("Milton, Frank", "27");
...
lis[10] = new ListItem("Norton, Vivian", "22");

lstLeft.DataSource = lis;
lstLeft.DataBind();
```

3. The second example considers binding an ArrayList of custom objects to a DropDownList. But, how does the DropDownList know what to use for the *Text* and *Value*? List controls have two additional properties:

   DataTextField – A string is used to set the field of the data source that provides the text content of the list items.

   DataValueField – A string is used to set the field of the data source that provides the value content of the list items.

   I believe that both of these must be the name of a full-fledged C# property, not just the name of an instance variable.

[1] https://stackoverflow.com/questions/7088017/asp-net-what-is-the-difference-of-datasourceid-and-datasource

4. Example – Note, this is a poor example in practice because the *Value* should be unique. Below, I have used the *NumPlayers* as the *Value* which probably would not be unique. However, the idea of binding an ArrayList of custom objects still works.

   Code to build ArrayList and bind:

```
ArrayList teams = new ArrayList();

teams.Add(new Team("Blazers", "Valdosta, GA", 14));
teams.Add(new Team("Terrapins", "College Park, MD", 13));
teams.Add(new Team("Tar Heels", "Chapel Hill, NC", 15));

ddlTeams.DataSource = teams;
ddlTeams.DataTextField = "Name";
ddlTeams.DataValueField = "NumPlayers";
ddlTeams.DataBind();
```

   The Team Class:

```
public class Team
{
	private string name;
	private string location;
	private int numPlayers;

	public string Name
	{
		get{ return name; }
		set{ name = value; }
	}

	public string Location
	{
		get{ return location; }
	}

	public int NumPlayers
	{
		get{ return numPlayers; }
	}

	public Team(string name, string location, int numPlayers)
	{
		this.name = name;
		this.location = location;
		this.numPlayers = numPlayers;
	}
}
```

5. Note that we only have to do the data binding once because once they are bound, then the data will persist across post-backs in the Form's post data (not ViewState) as is the case for any HTML form element.

6. We will consider data binding in more detail when we get to database.