Advanced Programming Language (630501)
Fall 2011/2012 – Lectures Notes # 17

Data Binding

Outline of the Lecture

- **Code- Behind**
- **Handling Events**
- **Data Binding (Using Custom Class and ArrayList class)**

Code-Behind

1. ***Code-behind*** is a new feature in ASP.NET that allows developers to truly separate the HTML and tag-based UI elements from the code that provides user interaction, validation, and so on. Code-behind modules offer developers a number of advantages:
   - Clean separation of HTML and code
   - Easier reuse.
   - Simpler maintenance.

Handling Events

2. Controls on mobile Web Forms, just like Web Forms controls, can raise various events; for example, a ***Command*** control can raise a ***Click*** event (this is similar to the Button control in a standard Web Forms control). Controls provide ***default events*** and ***nondefault*** events.

To create a default event handler for a control

1. After placing the control on the form, ***double-click*** the control.
   - The Mobile Forms ***designer*** opens the ***code-behind*** file for the current page and creates skeleton methods for handling the control's default events. For the Command control, the code looks like the following.

```
private void Command1_Click(object sender, System.EventArgs e)
{
}
```

2. Write code in the control's event handler methods that the application can call when the events occur. For the Command control, your code might resemble the following.

```
private void Command1_Click(Object sender, System.EventArgs e)
{
   Command1.Text = "Hello, Web Forms!";
}
```

Data Binding (Using Custom Class and ArrayList class)

1. Data Binding Properties:

A data-bound control may contain the following properties:

- ***DataSource*** specifies the ***data source*** for the data to populate the list. This may specify the table name.
- ***DataMember*** contains the name of the ***table*** to use in the DataSource property that you selected.
- ***DataTextField*** specifies the name of a ***field*** (for example, a column in the table) to associate with the ***Text*** property for each item in the control.
- ***DataValueField*** contains the name of a ***field*** to associate with the ***Value*** property for each item in the control. A typical use is to specify the primary field of a data record so that, at run time, you can identify the data row displayed in the list.

2. Resizable Arrays

- The FCL's ***System*** namespace contains a class named ***Array*** that models the behavior of ***static arrays***. ***System.Collections.ArrayList*** encapsulates ***dynamic arrays***—arrays that can be sized and resized as needed. ArrayLists are useful when you want to store data in an array but don't know up front how many items you'll be storing.
- Creating an ArrayList and adding items to it is simplicity itself:

```
ArrayList list = new ArrayList ();
list.Add ("John");
list.Add ("Paul");
list.Add ("George");
list.Add ("Ringo");
```

- ***Add*** adds an item to the end of the array and grows the array's memory allocation if necessary to accommodate the new item.
- The related ***Insert*** method inserts an item into an ArrayList at a specified position and moves higher-numbered items upward in the array. Insert also grows the array if that's necessary.

Example 17.1

```
<%@ Page language="c#" Codebehind="MobileWebForm1.aspx.cs"
            Inherits="MobileWebApplication5.MobileWebForm1"
            AutoEventWireup="false" %>
<%@ Register TagPrefix="mobile"
               Namespace="System.Web.UI.MobileControls"
               Assembly="System.Web.Mobile" %>
<script language="C#" runat="server" ID="Script1">
//private class declaration
//expose traffic related properties
private class Traffic
{
//private members of the class
string strHighWayName,strTravelTime;
//Default Constructor
public Traffic (string hwn, string tt)
{
strHighWayName = hwn;
strTravelTime = tt;
}
// HighWayName read only property
public string HighWayName
{
      get
      {
      return strHighWayName;
      }
}
// TravelTime read only property
public string TravelTime
{
        get
        {
        return strTravelTime;
        }
}
}
void Bind (object sender, EventArgs e)
{
//Declare a new ArrayList object
ArrayList TrafficList = new ArrayList();
TrafficList.Add(new Traffic("Street1","Time1"));
TrafficList.Add(new Traffic("Street2","Time2"));
TrafficList.Add(new Traffic("Street3","Time3"));
//Data Binding Mechanism
List1.DataTextField = "HighWayName";
List1.DataValueField = "TravelTime";
List1.DataSource = TrafficList;
List1.DataBind();
}
</script>
<mobile:Form id="Form1" runat="server">
<mobile:Label id="Label1" runat="server">Traffic Info</mobile:Label>
<mobile:List id="List1" runat="server"></mobile:List>
<mobile:Command id="Command1" runat="server"
                    OnClick = "Bind">Bind</mobile:Command>
</mobile:Form>
```

Example 17.2 a

```
<%@ Page language="c#" Codebehind="MobileWebForm1.aspx.cs"
Inherits="MobileWebApplication8.MobileWebForm1"
```

```
AutoEventWireup="false" %>
<%@ Register TagPrefix="mobile" Namespace="System.Web.UI.MobileControls"
Assembly="System.Web.Mobile" %>
<HEAD>
        <meta name="GENERATOR" content="Microsoft Visual Studio .NET 7.1">
        <meta name="CODE_LANGUAGE" content="C#">
        <meta name="vs_targetSchema"
content="http://schemas.microsoft.com/Mobile/Page">
</HEAD>
<body Xmlns:mobile="http://schemas.microsoft.com/Mobile/WebForm">
        <mobile:Form id="Form1" runat="server">
                <mobile:List id="List1" runat="server" Decoration="Bulleted"
DataTextField="HighWayName" DataValueField="TravelTime"></mobile:List>
        </mobile:Form>
</body>
```

Example 17.2 b

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.Mobile;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.MobileControls;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
namespace Custom
{
        public class Traffic
        {
                string strHighWayName,strTravelTime;
                public Traffic (string hwn, string tt)
```

```
		{
			strHighWayName = hwn;
			strTravelTime = tt;
		}
		public string HighWayName
		{
			get
			{
				return strHighWayName;
			}
		}
		public string TravelTime
		{
			get
			{
				return strTravelTime;
			}
		}
	}
}
namespace MobileWebApplication8
{
	using Custom;
	public class MobileWebForm1 : System.Web.UI.MobileControls.MobilePage
	{
		protected System.Web.UI.MobileControls.List List1;
		protected System.Web.UI.MobileControls.Form Form1;
		private void Page_Load(object sender, System.EventArgs e)
		{
			ArrayList TrafficList = new ArrayList();
			TrafficList.Add(new Traffic("Street1","Time1"));
			TrafficList.Add(new Traffic("Street2","Time2"));
			TrafficList.Add(new Traffic("Street3","Time3"));
			List1.DataSource = TrafficList;
```

```
                        List1.DataBind();
                        // Put user code to initialize the page here
                }
}
```