MONASH University

MONASH
INFORMATION
TECHNOLOGY

FIT5192 Module 2
Internet Applications Development
Lecture 5

GROUP OF EIGHT AUSTRALIA

MONASH University

Lecture Overview

1. Introduction
2. ASP.NET Validation Controls
3. Navigation controls
4. Login Control

GROUP OF EIGHT AUSTRALIA

MONASH University

Introduction

MONASH University

Validation Controls

ASP.NET Validation Controls

- RequiredFieldValidator
- RangeValidator
- CompareValidator
- RegularExpressionValidator
- CustomValidator

GROUP OF EIGHT AUSTRALIA

ASP Server Control Categories

Standard

- Basic controls such as buttons, links, images, lists etc

Validation

- Used to validate data input by a users.

Navigation

- navigation features, such as menues, to allow users to navigate around a web site.

Login

- provide a login mechanism for uses to gain access to a site.

Data

- Provide access to data sources such as databases or XML files

MONASH University

Validation Controls

RequiredFieldValidator

```
<table>
<tr>
<td>Departure Date:</td>
<td><asp:TextBox id="flightDepBox" runat="server" Width="80px"
   Height="22px"/><br /></td>
</tr>
</table>…..
```

```
<asp:RequiredFieldValidator id="vldDepDate" runat="server"
   ErrorMessage= "Please enter a valid Departure Date"
   ControlToValidate="flightDepBox" />
```

…..

Validation takes place when the page is submitted

the validation is done before any other action

will not continue until the validation has successfully completed.

RangeValidator

Used to ensure that a user entered value is between lower and upper boundaries

Can be used to check numbers, dates or alphabetic characters.

Boundaries can be constants or values derived from another control.

Valid values for the *type* attribute are:

Currency

Date

Double

Integer

String

RangeValidator

```
<table> <tr> <td>Number of seats:</td>
 <td>
<asp:TextBox id="txtSeats" runat="server" Width="40px"
Height="22px"/><br /> </td>
</tr> </table>
……..
<asp:RangeValidator id="rngSeats" runat="server"
ControlToValidate="txtSeats" ErrorMessage="Please enter between
1 and 4 seats" MinimumValue="1" MaximumValue="4"
Type="Integer" />
```

CompareValidator

Used to compare the value of

two controls or

of a control and a specified value

CompareValidator

```
<table> <tr> <td>Number of seats:</td> <td> <asp:TextBox
id="txtSeats" runat="server" Width="40px" Height="22px"/> <br /> </td>
</tr>

<tr> <td>Confirm number of seats:</td> <td> <asp:TextBox
id="txtSeats2" runat="server" Width="40px" Height="22px"/>

<br /> </td> </tr> </table>

……..

<asp:CompareValidator ID="cmpSeats" runat="server"
ControlToValidate="txtSeats" ControlToCompare="txtSeats2"
ErrorMessage="Please enter same value for Seats and Confirm Seats"
/>
```

RegularExpressionValidator

Checks that the input into a textbox follows a certain pattern.

For Example checking format of an email address.

compares the input to the regular expression (Regex)

".*@.*\..*".

Assuming that the valid email address will be of the form:

"someone@somewhere.com"

Email RegEx explained

.*	the dot means any character, and the star means any number of them
@	means the @ character
.*	as above
\.	means the dot character, escaped with a \
.*	as above

Regular Expressions (Part 1)

. (the full stop) stands for any character except \n

\d for any digit, \D for any non-digit

\s for white space, \S any non-whitespace character

\w any word character: a to z, A to Z or 0 to 9 or underscore

x* for zero or more x's,(xy)* for zero or more (xy)'s

x? for zero or one x,(xy)? for zero or one (xy)

x+ for one or more x's, (xy)+ for one or more (xy)'s

[x] the single character 'x'

[^x] any character except 'x'

[xyz] to include one of a group of values: x or y or z

[^xyz] any character except x or y or z

Regular Expressions (Part 2)

[0-9] any digit from 0 to 9

[a-d] any character a,b,c or d

this | that: have "this" or "that" in the element content. Extra vertical bars can be added for additional choices

x{5} to have exactly five x's in a row

x{5,} to have at least five x's

x{5,8} to have at least 5 and at most 8 x's

(xyz){2} to have 2 xyz's. Parentheses control the curly brackets and other modifiers (?,+,*)

The characters + ? . * ^ $ ( ) [ ] { } | \ and usually / must be escaped with a backslash \ to be taken literally.

RegularExpressionValidator

```
<table> <tr> <td>Email:</td>

<td> <asp:TextBox id="txtEmail" runat="server" Width="200px"
Height="22px"/>

</td></table>

…….

<asp:RegularExpressionValidator id="validateEmail" runat="server"
ControlToValidate="txtEmail" ValidationExpression = ".*@.*\..*"
ErrorMessage="Invalid email" Display="dynamic" />
```

CustomValidator

Allows the developer to write their own validation method.

The *CustomValidator* method is specified by the *OnServerValidate* attribute.

```
<asp:CustomValidator id="vldFlightDates" runat="server"
ControlToValidate="flightDepBox" OnServerValidate="ValidateTravelData" />
```

The *args* argument

is set to false and will only be

set to true if all the validation checks within the function evaluate are successful.

CustomValidator example (Part 1)

```
protected void ValidateTravelData(object source,
   ServerValidateEventArgs args) {
// Since we have a bit to validate,
//assume that the entry is invalid....
args.IsValid = false;
DateTime departDate, returnDate;
feedbackLabel.ForeColor = System.Drawing.Color.Maroon;
```

CustomValidator example (Part 2)

```
try {
    departDate = DateTime.Parse(flightDepBox.Text);
}
catch (Exception ex)
{
    feedbackLabel.Text = "Departure Date is invalid.<br /> " +
    "Enter a valid date, for example: 05/12/2011";
return;
}
```

CustomValidator example (Part 3)

```
// Verify that the departure date is less than the
// return date - no same day trips in this system!
if (departDate >= returnDate) {
    feedbackLabel.Text = "The Departure Date must be " +
        "earlier than the Return Date and no same-day " +
        "returns for this travel package!";
return;
}
```

CustomValidator example (Part 4)

```
// Everthing is valid - set the IsValid flag...

args.IsValid = true;
```

MONASH University

Navigation controls

MONASH University

ASP.NET Navigation Controls

- Site Map
- SiteMapPath
- TreeView
- Menu

GROUP OF EIGHT AUSTRALIA

Site map

- A web.sitemap file is created to match the website
- Each siteMapNode control has

  a url attribute pointing to the page for that node,

  a title attribute containing the text that will appear on a fly out menu,

  and a description which will appear on the associated tool tip.
- The sitemapnodes, their nesting within each other, indicates the layout of the site.
- The sitemap control must have one and only one sitemapNode as its root element, as this file is XML compliant.
- Normally the root siteMapNode does not have any attributes and is just a container for the other sitemapNodes.

Sitemap example (Simplified)

```
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
  <siteMapNode>
      <siteMapNode url="~/default.aspx" title="Title page" />
      <siteMapNode url="~/ADU.aspx" title="Adults"
          description="Books for adult readers”>
          <siteMapNode url="~/AG.aspx" title="Australiana & General
  Interest"          description="Books about Australia and of general interest”
  />
      </siteMapNode>
</siteMapNode>
</siteMap>
```

SitemapPath Control

Bread crumbs are implemented using the <SiteMapPath> control

This control takes its data from the *web.sitemap* file.

```
<form id="form1" runat="server">
<asp:SiteMapPath ID="SiteMapPath1"  runat="server">
<PathSeparatorTemplate> -->
</PathSeparatorTemplate>
</asp:SiteMapPath>
</form>
```

More SitemapPath Control

Images can be used in the bread crumbs:

```
<form id="form1" runat="server“>
<asp:SiteMapPath ID="SiteMapPath1"  runat="server">
<PathSeparatorTemplate>
<asp:Image ID="emerald" runat="server"
ImageUrl="pix/emerald.gif" />
</PathSeparatorTemplate>
</asp:SiteMapPath>
</form>
```

MONASH University

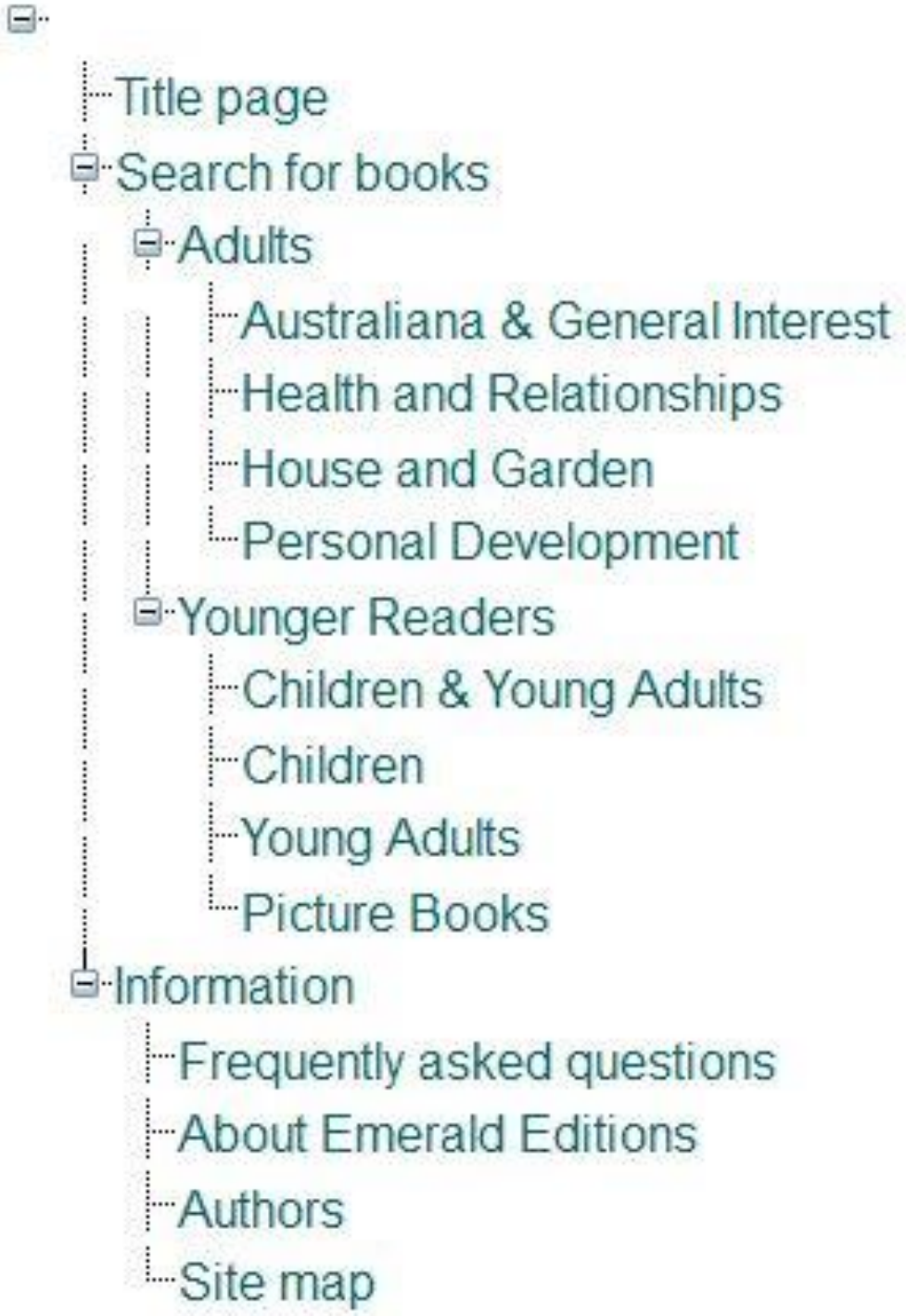
TreeView Control

Used to display a site map of a website.

The map is clickable:

clicking on an entry loads the relevant page,

the tree can be expanded and contracted.



GROUP OF EIGHT AUSTRALIA

TreeView Control

```
<asp:SiteMapDataSource ID="SiteMapDataSource1" runat="server" />

<asp:TreeView ID="TreeView1" runat="server"
   DataSourceID="SiteMapDataSource1" ShowLines="True" />
```

Menu control

Used to construct a fly out menu.

data is taken from the *web.sitemap* file.

```
<asp:SiteMapDataSource ID="SiteMapDataSource1"  runat="server" />


<asp:Menu StaticDisplayLevels="2" ID="Menu1" runat="server"
    DataSourceID="SiteMapDataSource1" StaticSubMenuIndent="25"
    ForeColor="Black">
    <StaticMenuItemStyle CssClass="MenuStaItm" />
    <DynamicHoverStyle CssClass="MenuDynHov" />
    <DynamicMenuItemStyle CssClass="MenuDynItm" />
    <StaticHoverStyle CssClass="MenuStaHov" />
</asp:Menu>
```

Login Controls

IIS and ASP.NET implementation of Authentication

Anonymous authentication

ASP.NET impersonation

Basic authentication

Forms based authentication

Windows authentication

the use of the ASP.NET Login Control

Anonymous Authentication

- Allowed by default for all new IIS Applications
- all application pools operate under the Network Service account
- or the Default Application Pool account
- These user accounts have low-level user access rights.
- Users do not have to log on to access the web site.

ASP.NET Impersonation

- IIS allows us to specify a user to impersonate when executing scripts
- For example, when students register for this unit for A2 depolyment web server access, an account, a directory and virtual directory are created for them.
- This process requires Adminstration rights to the student web server, so this particular process impersonates the Administrator user to enable these functions to be successfully completed.

Basic Authentication

- Requires that users provide a valid user name and password to access content.
- This authentication also works across firewalls and proxy servers.
- A good choice when you want to restrict access to some, but not all, content on a server.
- Disadvantage of Basic authentication is that it transmits unencrypted base64-encoded passwords across the network.
- Anonymous Authentication must be disabled in order for Basic Authentication to work.

Windows Authentication

- A method more suited to an Intranet environment rather than the Internet.
- It works by attempting to log on to an application using the credentials the user used when they logged on to the domain.
- If this fails, the user will then be requested to enter a Username and Password.
- This authentication method does hash (encrypt) the credentials sent, but does not work over HTTP proxy servers.
- Best for Intranets where it is known that all the clients are within a single domain.

Forms Authentication

- Enabled for either part of or the whole of an IIS Application
- An unauthorised user requests a restricted page, is directed to a login page (this can be either specified via the IIS Manager - or in the web.config file).
- Login page determines if the user's credentials are valid. If so:
  - to create a forms authentication ticket,
  - redirect the user back to the page they were attempting to visit.
- Authentication ticket included in subsequent requests.
- Authentication ticket passed via a cookie, or passed, in an encrypted form, as part of the URL

ASP.NET Login Control

Used in a Forms authentication application to check user entered credentials against a MS Access database.

For example contains 4 files:

- default.aspx
- login.aspx
- login.mdb
- style1.css

The Authentication for the Virtual Directory must have Anonymous and Forms Authentication enabled as shown in the example web.config file (a few slides later).

ASP.NET Login Control (Continued)

Forms Authentication needs to be enabled, but need Anonymous Authentication for login page (login.aspx)

default.aspx will trigger the authentication process

the user entered credentials will be checked against the login.mdb database.

default.aspx does not force the authentication, that action is handled by IIS/ASP.NET.

As soon as an unauthenticated user attempts to access a file within the Virtual Directory, they are redirected to *login.aspx*

Login Control (Part 1)

```
<asp:Login ID="Login" runat="server"
   OnAuthenticate="Login_Authenticate"

   CssClass="login"

   TitleText="<br />Please enter your details <br /> below to login for this
   site.<br /><br />"

   UserNameLabelText="Username:"
   UserNameRequiredErrorMessage="Username required<p />"
   PasswordLabelText="Password:"
   PasswordRequiredErrorMessage="Password required"

   Height = "250" Width = "330"

   LoginButtonText="Click to login" DisplayRememberMe="false" >

   <LabelStyle CssClass="loginText" />
   <TitleTextStyle CssClass="loginText" />
   <ValidatorTextStyle CssClass="loginValidator" />

</asp:Login>
```

Login Control (Part 2)

```
<asp:ValidationSummary id="vlSummary1" Font-Names="Arial"
    Visible="true" CssClass="vldSummary" runat="server"
    ValidationGroup="Login" HeaderText="Please correct the following
    errors:" />
```

In order to display the "required" validation messages, a Validation Summary control must be added to the page which has its ValidationGroup property set to the ID of the Login control.

If committed, the validation will still work, but will only display an asterisk * next to each required field.

ASP.NET Login Control - web.config

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
<system.web>
      <authentication mode="Forms">
      <forms name="ass" path="/"
loginUrl="login.aspx" protection="All" timeout="10"/>
      </authentication>
      <authorization>
            <deny users="?" />
      </authorization>
</system.web>
</configuration>
```

MONASH University

Summary

1. Introduction
2. ASP.NET Validation Controls
3. Navigation controls
4. Login Control

GROUP OF EIGHT AUSTRALIA

MONASH University

What you will do in the Studio

Complete topic 7 exercises

Read ahead ASP examples from topic 8 and work on Major Task question

Run using Visual Studio 2015/2017

GROUP OF EIGHT AUSTRALIA

MONASH University

Thanks and See you in the Studio!