SHRI MAHAVIR VIDYAMANDIR TRUST
सा विद्या या विमुक्तये

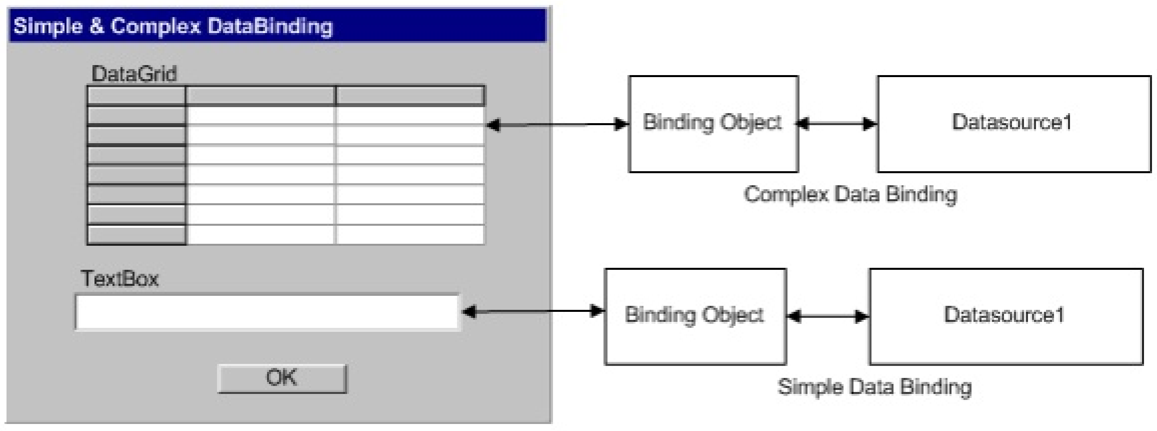
ASP.NET UNIT 2

Unit No 2. Server Control

1. What is Simple and Complex Data binding? Explain it with example.

DataBinding is a powerful feature provided by the .NET framework that enables Controls to connect to a datasource such as DataSets, DataViews, Arrays etc. Some of the controls in the client can be TextBox, Datagrid etc. A two-way connection is established such that any changes made to the datasource are reflected immediately in the visual element and vice versa.

Below is a graphical description of the concept of databinding:



The .NET framework provides a very flexible and powerful approach to databinding and allows the programmer to have a filled control over the steps involved in the whole process. One of the biggest improvements with .Net has been the introduction of databinding to web pages through the use of .Net server-side web controls. Hence, building data driven web applications has been greatly simplified.

Advantages of DataBinding

1. Databinding in .NET can be used to write data driven applications quickly. .NET data binding allows you to write less code with fast execution but still get the work done in the best way.

2. .NET automatically writes a lot of databinding code for you in the background, so the developer does not have to spend time writing code for basic databinding, but still has the flexibility of modifying any code that he would like to. We get the benefits of bound as well as unbound approach.

3. Control over the Databinding process by using events.

Disadvantages of DataBinding

- More optimized code can be written by using the unbound or traditional methods.
- Complete flexibility can only be achieved by using the unbound approach.

For databinding to take place data provider and a data consumer should exist so that a synchronized link is established between the two. Data providers contain the data and the data consumers use the data exposed by the data providers and display them.

.NET has expanded the scope of possible data providers. In .NET any class or component that implements the IList interface is a valid DataSource. If a component implements the IList interface then it is transformed into an index based collection.

Some of the classes that support the IList interface in the NET framework are given below. Please note that any class that implements the IList interface is a valid data provider.

1. Arrays
2. DataColumn
3. DataTable
4. DataView
5. DataSet

Ex.

```
Dim ds As New DataSet
Dim cmd As New SqlCommand
Dim da As New SqlDataAdapter
Dim cn As New SqlConnection(strConnection)
cn.Open()
cmd.Connection = cn
cmd.CommandText = "Select Code, CourseName from CourseMst order by Code"
da.SelectCommand = cmd
da.Fill(ds)
GridView1.DataSource = ds.Tables(0)
GridView1.DataBind()
```

2. Difference Between client side and server side validation. What are the advantages of validator control? Explain each validation control with example.

Validations can be performed on the server side or on the client side (web browser). The user input validation take place on the Server Side during a post back session is called Server Side Validation and the user input validation take place on the Client Side (web browser) is called Client Side Validation. Client Side Validation does not require a postback. If the user request requires server resources to validate the user input, you should use Server Side Validation. If the user request does not require any server resources to validate the input , you can use Client Side Validation.

Server Side Validation
In the Server Side Validation, the input submitted by the user is being sent to the server and validated using one of server side scripting languages such as ASP.Net, PHP etc. After the validation process on the Server Side, the feedback is sent back to the client by a new dynamically generated web page. It is better to validate user input on Server Side because you can protect against the malicious users, who can easily bypass your Client Side scripting language and submit dangerous input to the server.

Client Side Validation
In the Client Side Validation you can provide a better user experience by responding quickly at the browser level. When you perform a Client Side Validation, all the user inputs validated in the user's browser itself. Client Side validation does not require a round trip to the server, so the network traffic which will help your server perform better. This type of validation is done on the browser side using script languages such as JavaScript, VBScript or HTML5 attributes.
For example, if the user enter an invalid email format, you can show an error message immediately before the user move to the next field, so the user can correct every field before they submit the form.
Mostly the Client Side Validation depends on the JavaScript Language, so if users turn JavaScript off, it can easily bypass and submit dangerous input to the server . So the Client Side Validation cannot protect your application from malicious attacks on your server resources and databases.
As both the validation methods have their own significances, it is recommended that the Server side validation is more SECURE!

ASP.NET validation controls validate the user input data to ensure that useless, unauthenticated, or contradictory data don't get stored.

ASP.NET provides the following validation controls:

- **RequiredFieldValidator**
- **RangeValidator**
- **CompareValidator**
- **RegularExpressionValidator**
- **ValidationSummary**

RequiredFieldValidator Control
The RequiredFieldValidator control ensures that the required field is not empty. It is generally tied to a text box to force input into the text box.
The syntax of the control is as given:

```
<asp:RequiredFieldValidator ID="rfvcandidate"
   runat="server" ControlToValidate ="ddlcandidate"
   ErrorMessage="Please choose a candidate"
   InitialValue="Please choose a candidate">

</asp:RequiredFieldValidator>
```

Rangevalidator Control

The RangeValidator control verifies that the input value falls within a predetermined range.
The syntax of the control is as given:

```
<asp:RangeValidator ID="rvclass" runat="server" ControlToValidate="txtclass"
   ErrorMessage="Enter your class (6 - 12)" MaximumValue="12"
   MinimumValue="6" Type="Integer">

</asp:RangeValidator>
```

CompareValidator Control

The CompareValidator control compares a value in one control with a fixed value or a value in another control.
The syntax of the control is as given:

```
<asp:CompareValidator ID="CompareValidator1" runat="server"
   ErrorMessage="CompareValidator">

</asp:CompareValidator>
```

RegularExpressionValidator Control

The RegularExpressionValidator allows validating the input text by matching against a pattern of a regular expression. The regular expression is set in the ValidationExpression property.
The syntax of the control is as given:

```
<asp:RegularExpressionValidator ID="string" runat="server"
ErrorMessage="string"
   ValidationExpression="string" ValidationGroup="string">

</asp:RegularExpressionValidator>
```

ValidationSummary Control

The ValidationSummary control does not perform any validation but shows a summary of all errors in the page. The summary displays the values of the ErrorMessage property of all validation controls that failed validation.
The following two mutually inclusive properties list out the error message:

- **ShowSummary** : shows the error messages in specified format.
- **ShowMessageBox** : shows the error messages in a separate window.

The syntax of the control is as given:

```
<asp:ValidationSummary ID="ValidationSummary1" runat="server"
   DisplayMode = "BulletList" ShowSummary = "true" HeaderText="Errors:" />
```

3. Navigation controls

A web site has many web pages which provide information to the user. Users have to navigate across web pages easily and quickly. Navigation in a web site is essential and important. It provides a means for allowing the user to access the required information quickly in the web application. Hyperlinks act as a basic mechanism of web site navigation. The Hyperlinks act as static links. The navigation mechanisms used may be simple or complex, depending on the web site.

In addition, ASP.NET has three new navigation controls:

- Site Map Path
- Menu Control
- TreeViews

The Site Map

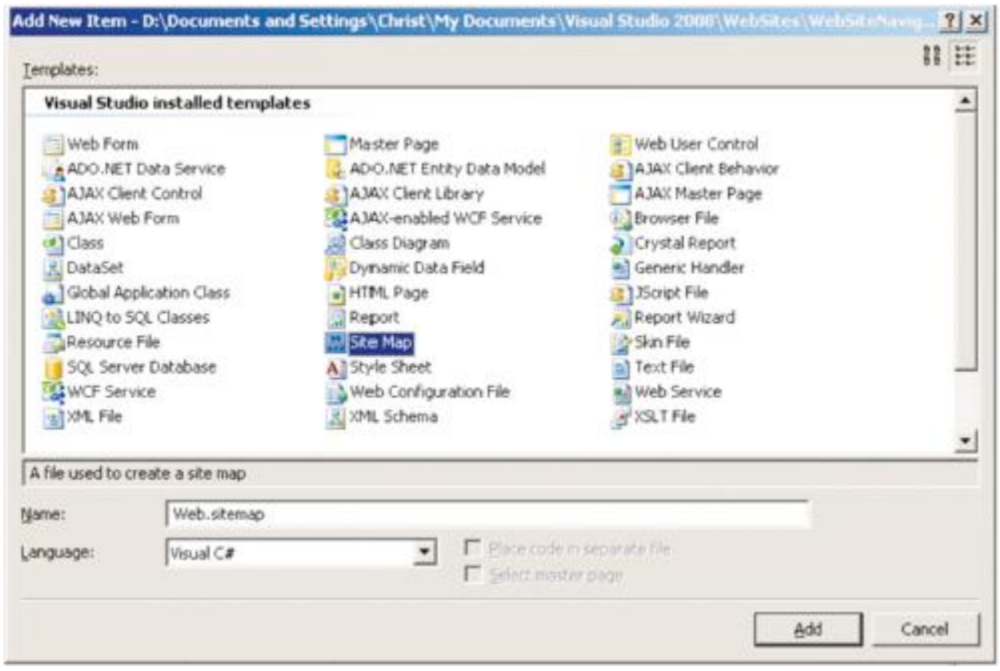
The Site Map provides a logical structure of the pages in a website. The Site Map information is stored in a file named Web.sitemap. The Web.sitemap file is stored in the root directory of the web application. ASP.NET has the XmlSiteMapProvider which is used to access the information from the Web.sitemap file. It automatically reads data in the Web.sitemap file present in the root folder of the web application and stores it in a SiteMap object. Let us now consider the banking scenario where Banking can be of two types, namely Personal Banking and Corporate Banking. Further more personal banking has Accounts, Loans and Insurance.Similarly, Corporate Banking has Credit and Capital Market. The Web.sitemap file can be created with the required information.

Demonstration: Creating the Web.sitemap file

Steps to create the Web.sitemap file.

1) Create a New Web Site named “WebSiteNavigation”.

2) Right-click the project in the Solution Explorer and Select Add New Item and select Site Map as shown below and click the Add button.


Add New Item - D:\Documents and Settings\Christ\My Documents\Visual Studio 2008\WebSites\WebSiteNavig

A file used to create a site map

Name: Web.sitemap

Language: Visual C#

3) Enter the following content below in the Web.sitemap file.

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
    <siteMapNode url="" title=""  description="">
       <siteMapNode url="Home.aspx" title="Home"  description="Home Page" />
       <siteMapNode url="javascript:;" title="Services"  description="Services Page">
           <siteMapNode url ="Consulting.aspx" title="Consulting"
               description="Consulting Page"></siteMapNode>
           <siteMapNode url ="Outsourcing.aspx" title="Outsourcing"
              description="Outsourcing Page"></siteMapNode>
       </siteMapNode>
       <siteMapNode url="About.aspx" title="About"description="About Us Page" />
       <siteMapNode url="Contact.aspx" title="Contact"description="Contact Us Page" />
</siteMapNode>
</siteMap>
```

The SiteMapDataSource Control

The SiteMapDataSource control obtains the site map information from the SiteMap object.The Navigation controls can display the data from the SiteMapDataSource control. For example, you can use a tree view control to display the information stored in the Web.sitemap file by using the SiteMapDataSource control. It is easy to use the SiteMapDataSource control.The SiteMapDataSource control can be created by dragging the control available in the Data tab of the Toolbox into the Design View of the Web Form. It can also be created by entering the code below in the Source View of the Web form.

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

The Menu Control

The Menu Control can be used to display hierarchical data. It is very easy to navigate through the Menu Control when there are many options in the hierarchy. Menus occupy less space in the web page. Menus can be expanded at various levels or collapsed as required by the user.The Menu control can be bound to a SiteMapDataSource to display the information available in the Web.sitemap file. The DataSourceID property of the Menu Control is set to the ID of the SiteMapDataSource control. Below is the tag for the Menu Control.

```
< asp:Menu ID="Menu1"  runat="server"   DataSourceID="SiteMapDataSource1" >
< /asp:Menu >
```

Demonstration: Creating a Menu Control to display the information stored in the Web.sitemap file using the SiteMapDataSource Control.

Steps to create the Web Page.

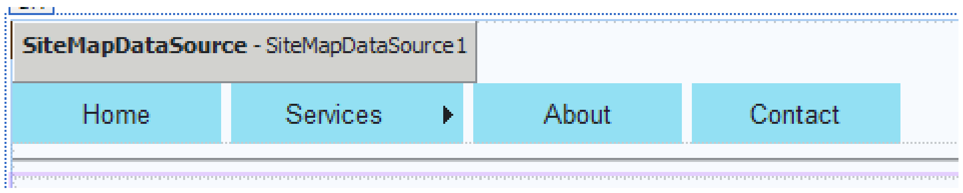
1) Create a new web form named MenuControlDemo.aspx in the WebSiteNavigation Project.

2) Add a Label and set its Text as "Menu Control Demo"

3) Drag a SiteMapDataSource Control from the Data Tab of the Toolbox into the Web Form.

4) Drag a Menu Control from the Navigation Tab of the Toolbox into the Web Form.

5) Click on the Smart Tag of the Menu Control and in the Choose Data Source Drop Down, select SiteMapDataSource1.



The TreeView Control

The TreeView control can be used to display a complex hierarchical structure of items. TreeView can be used fo r purposes, such as to display the Windows File System Directory Structure, or any structure with a hierarchy. The TreeView can be expanded to many levels.The TreeView is an advanced server control. It can display the details contained in the Web.sitemap file by using the SiteMapDataSource control. The DataSourceID property of the TreeView control is set to the ID of the SiteMapDataSource. Below is the tag for TreeView Control.

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

Demonstration: Creating a TreeView Control to display the information stored in the Web.sitemap file using the SiteMapDataSource Control.

Steps to create the Web Page.

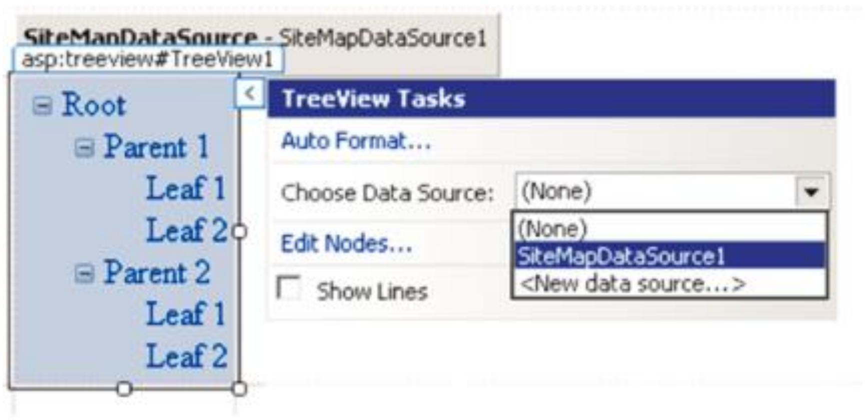
1) Create a new web form named TreeViewDemo.aspx in the WebSiteNavigation Project.

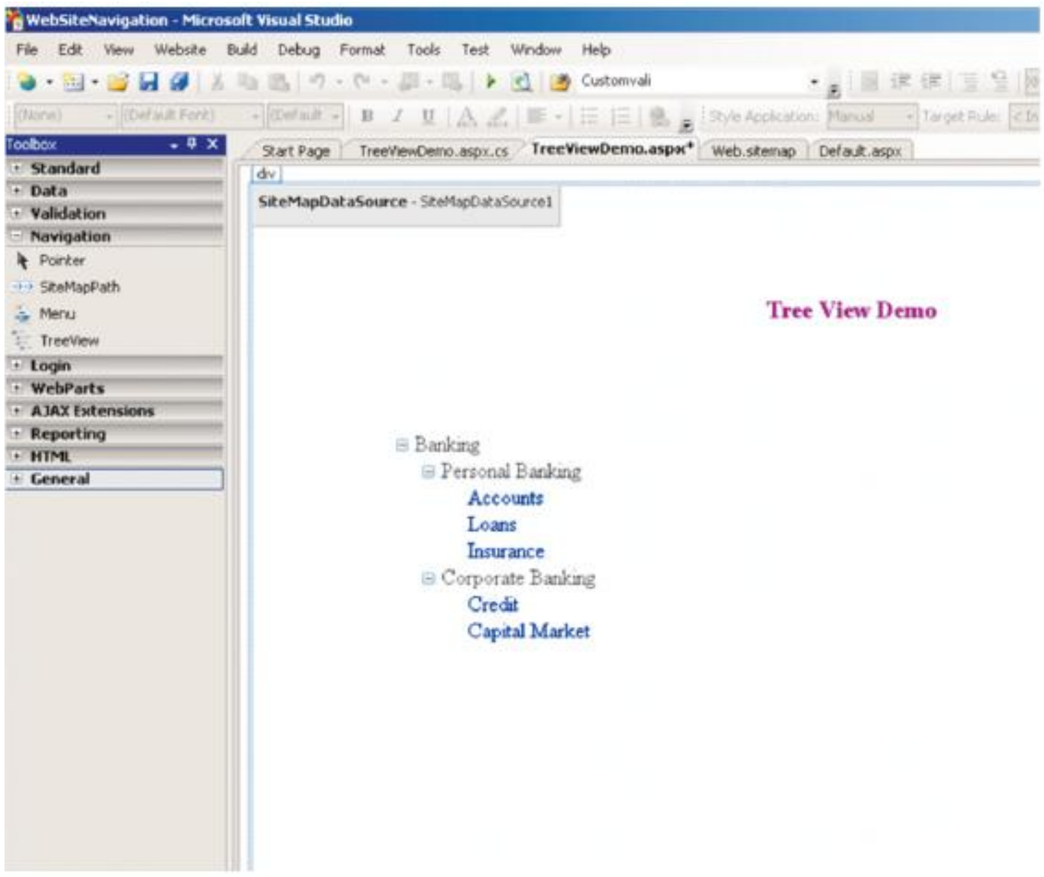
2) Add a Label and set its Text as "Tree View Demo"

3) Drag a SiteMapDataSource Control from the Data Tab of the Toolbox into the Web Form.

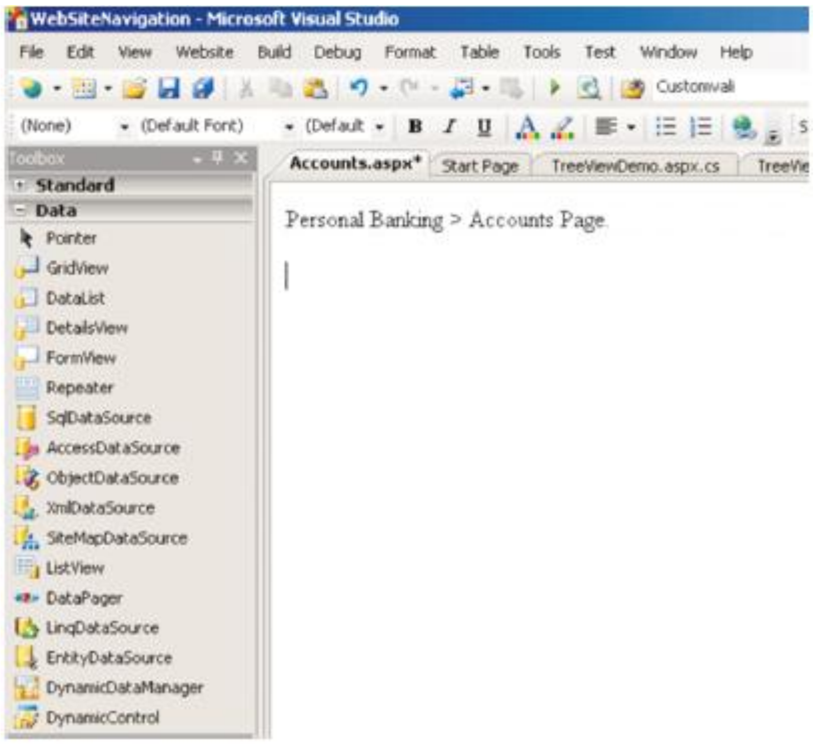
4) Drag a TreeView Control from the Navigation Tab of the Toolbox into the Web Form.

5) Click on the Smart Tag of the TreeView Control and in the Choose Data Source Drop Down, select SiteMapDataSource1.





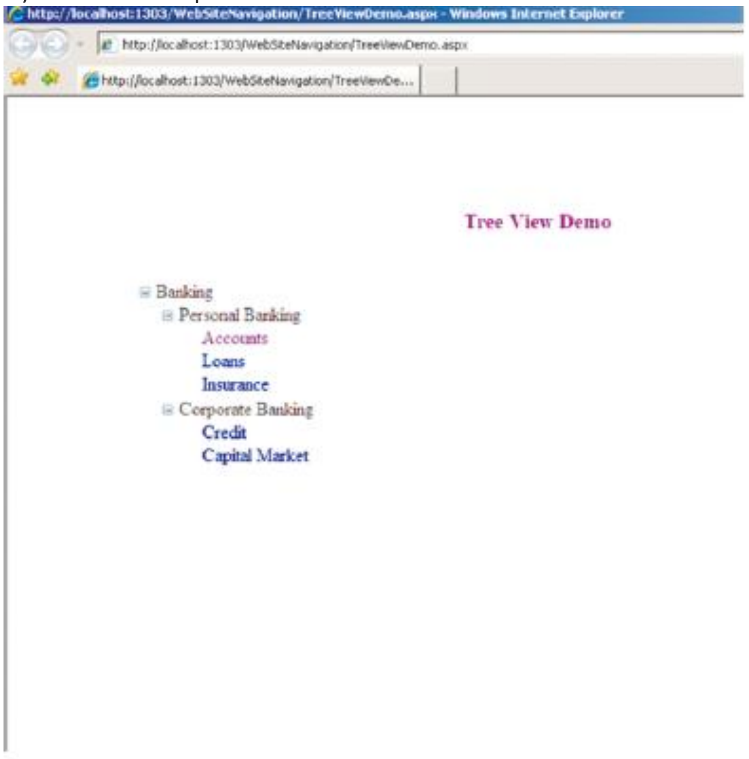
6) Create a New Web Form named Accounts.aspx and enter “Personal Banking > Accounts Page” inside the page. The Accounts page design should look as below:



Personal Banking > Accounts Page

7) Right-Click the TreeViewDemo.aspx file and select “Set As Start Page” and execute the project.

8) View the Output 1 below :


http://localhost:1303/WebSiteNavigation/TreeViewDemo.aspx - Windows Internet Explorer
http://localhost:1303/WebSiteNavigation/TreeViewDemo.aspx

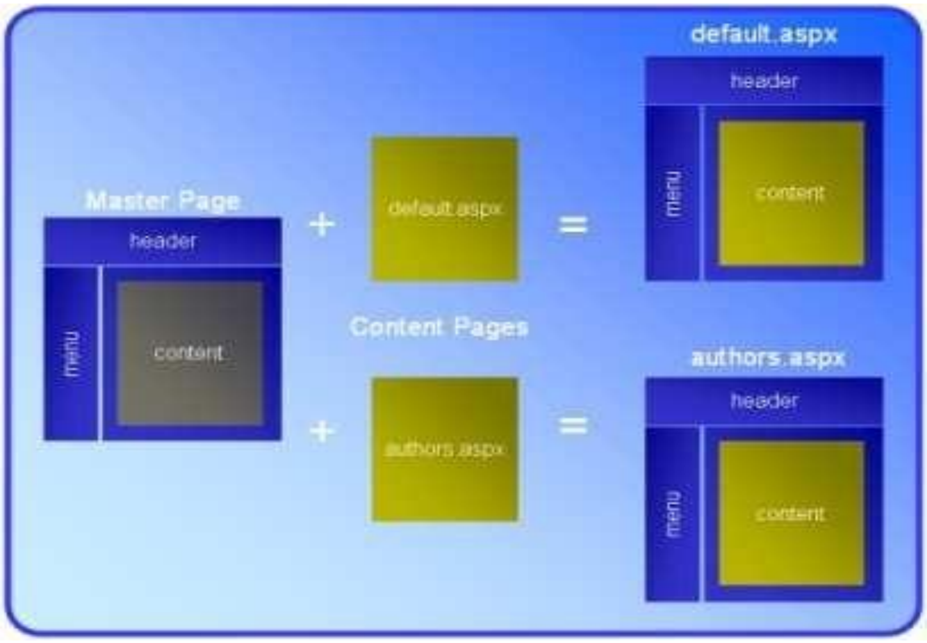
Tree View Demo

- Banking
  - Personal Banking
    - Accounts
    - Loans
    - Insurance
  - Corporate Banking
    - Credit
    - Capital Market

4. Explain master page and Themes.

MASTER PAGES

- Masterpagesallowyoutocreateaconsistentlookandbehaviorforallthepagesinweb application.
- Amasterpageprovidesatemplateforotherpages,withsharedlayoutandfunctionality.The master page defines placeholders for the content, which can be overridden by content pages.
- The output result is a combination of the master page and the content page.
- The content pages contain the content you want to display.
- When users request the content page, ASP.NET merges the pages to produce output that combines the layout of the master page with the content of the content page.
- Master page having .Master extension.
- Define master page using @Master directive on top a web page.
- The Content of master pages is fixed for all derived pages. To place the changing data for every pages master page provide <ContentPlaceHolder> where each page hold its local content.
- Major characteristics of a master page:
  - ✓ Defining common properties of a website, such as header, footer, banners, navigation menus and other elements that can be accessed by the content pages.
  - ✓ Allowing single or multiple content pages to access single or multiple master pages.
  - ✓ Displaying the content of each content page in content place holder.



MasterPage.master

```
<%@ Master Language="VB"
CodeFile="MasterPageTest.master.vb"
Inherits="MasterPageTest" %>
<html>
<head runat="server">
    <title>Untitled Page</title>
    <asp:ContentPlaceHolder id="head"
    runat="server"></asp:ContentPlaceHolder>
</head>
<body><form id="form1" runat="server">
    <div><h1>Standard Header From Masterpage</h1>
        <asp:ContentPlaceHolder id="CPH1" runat="server">
        </asp:ContentPlaceHolder>
      <div>
        <asp:Button ID="btnContent2" runat="server" Text="Content-
        1" />
    </div></div></form></body></html>
```

MasterPage.master(Coding)

```
Public Class MasterPage
    Inherits System.Web.UI.MasterPage
  Protected Sub btnContent2_Click(ByVal sender As Object, ByVal
e As EventArgs) Handles btnContent2.Click
        Response.Redirect("Control.aspx")
   End Sub
End Class
```

Default.aspx

```
<%@ Page Title="" Language="VB" MasterPageFile="~/MasterPage.master"
AutoEventWireup="false" CodeFile="Default.aspx.vb" Inherits="_Default" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="cph1" Runat="Server">
    <h1> Home Page </h1>
<br /> <br />
<asp:Label ID="lblData" runat="server" Font-Size="Large"></asp:Label>
</asp:Content>
```

- **The code for the MainMasterPage.master should look like this:**

```
<%@ Master Language="VB" CodeFile="MasterPage.master.vb" Inherits="MasterPage" %>
<head id="Head1" runat="server">
    <title>V.T. Poddar BCA College</title>
    <link href="css/my.css" rel="stylesheet" />
    <asp:ContentPlaceHolder id="head" runat="server">
    </asp:ContentPlaceHolder>
</head>
```

```
<body>
        <header id="header">
            <h1>V.T. Poddar BCA College</h1>
        </header>
        <ul>
                <li><a href="Default.aspx">Home</a></li>
                <li><a href="Sum.aspx">Sum</a></li>
                <li><a href="Course.aspx">Course</a></li>
                <li><a href="YearAdd.aspx">Year</a></li>
                <li><a href="Student.aspx">Student</a></li>
                <li><a href="DataListPage.aspx">DataList</a></li>
                <li><a href="RepeaterPage.aspx">Repeater</a></li>
                <li><a href="CookiesPage.aspx">Cookie</a></li>
                <li><a href="TreeviewPage.aspx">Treeview</a></li>
                <li><a href="FormViewPage.aspx">Formview</a></li>
                <li><a href="QueryStringPage.aspx">QeryStr</a></li>
                <li><a href="ViewStatePage.aspx">ViewState</a></li>
                <li><a href="ImageUpload.aspx">FileUpload</a></li>

                <li><a href="Loginpage.aspx" class="log">Logout</a></li>
                <asp:Label ID="username" runat="server" Text="Username"></asp:Label>
            </ul>
        </nav>

        <div id="con">
            <asp:ContentPlaceHolder ID="cph1" runat="server">

            </asp:ContentPlaceHolder>
        </div>

        <footer id="footer">
            copyright VTP BCA College @2020
        </footer>
    </body>
</html>
```

THEMES WITH SKIN FILE

Theme

- ThemesinAsp.Netenableyoutodefinethestylepropertiesandthechangetheappearance ofaWeb Page.
- They provides User friendly interface with similar all over look with optimized code.
- You can set theme in master page directive in master page as well in web.config also.
- Theme can be defined as collection of element that is common look and feel for whole application.
- The Key Features are as follows:

✓ Providing flexibility to easily change the appearance of all web pages just by making changes in few template files with minimum possible efforts.

✓ Providing various options to customize a web page. E.g. in Orkut, you can change website appearance as per your requirement.

- Themes contain following elements:

Skin

- Itishaving.skinextensionthatcontainstags,propertysettingsandformattingoptionsfor serverside controls. Such as Label, TextBox and Button.
- Skin is divided into following two groups:
- **Default:** Default skin is applied automatically to all the controls of the same type in web page, when themeisappliedtothepage.SkinIDattributesisusedtodeterminewhetherthe applied skin is default or not. If SkinID is not present in current tag means it is a default skin.

  Named: Itisacustomizedskinthatisappliedtocontrolswhenthemeisapplied.IthasSkinID property which allows different styles to different controls.

Applying existing theme to an application

- There are 3 different options to apply themes to our website:

1. Setting the theme at the page level: the Theme attribute is added to the page directive of the page.
   <%@ Page Language="VB" CodeFile="Default.aspx.cs"Inherits="Default" Theme="Theme1"%>
2. Setting the theme at the site level: to set the theme for the entire website you can set the theme in the web.config of the website. Open the web.config file and locate the <pages> element and add the theme attribute to it:

   <pages theme="Theme1">

   ....

   </pages>
3. Setting the theme programmatically at runtime: here the theme is set at runtime through coding. Page.Theme =Theme1;

CSS

- Refers to a mechanism of adding fonts, colors, styles and behavior to web pages.
- CSS is used to provide the visual inheritance to all the web pages in the website.
- It is an Asp.Net file with .css extension and applies a style sheet as a part of theme in a web page.
- It should include in App_theme folder.

Create CSS File & Theme

- Add Theme:

✓ Solution Explorer -> Right click -> Add ASP.NET folder -> Themes.

A folder App_Themes is added to the Solution Explorer and a new folder Theme1 is added inside it.

- ✓ Theme1 -> Right click -> Add new item -> Skin File → Skin1.skin
- ✓ Inside Skin1.skin

```
<asp:label runat="server" width="300px" height="40px" font-
bold="true" font- size="x-large" forecolor="yellow"
backcolor="green" SkinID=”lbl”/>

<asp:textbox runat="server" font-bold="true" font-
size="medium" forecolor="blue" backcolor="silver"
font-italic="true" SkinID=”txt”/>

<asp:button runat="server" forecolor="red" backcolor="yellow"
SkinID=”btn”/>
```

Create CSS

- **Add Theme file:**
- ✓ Solution Explorer -> Right click → Add New Item → Select StyleSheet
- ✓ Inside StyleSheet1.css

```
p
  {
        font-family:Impact;
        text-align:center;
         color:Green;
     }
```

Applying CSS to a webpage;

Content.aspx

```
<%@ Page Language="vb" AutoEventWireup="false"
CodeBehind="Content.aspx.vb" Inherits="CSSExample.Content"
%>
<head runat="server">
    <title></title>
    <Link rel="Stylesheet" type="text/css" href="StyleSheet1.css"
    />
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <p> Hello My Name is ABCXYZ</p>
        <p> I am Student of V.T. PODDAR BCA COLLEGE</p>
        <p> I am a Software Engineer</p>
   </div>
   </form>
</body>
</html>
```

5. Explain AdRotator control with example.

The AdRotator control randomly selects banner graphics from a list, which is specified in an external XML schedule file. This external XML schedule file is called the advertisement file.

The AdRotator control allows you to specify the advertisement file and the type of window that the link should follow in the AdvertisementFile and the Target property respectively.

The basic syntax of adding an AdRotator is as follows:

```
<asp:AdRotator  runat = "server" AdvertisementFile = "adfile.xml"  Target =
"_blank" />
```

Before going into the details of the AdRotator control and its properties, let us look into the construction of the advertisement file.

The Advertisement File

The advertisement file is an XML file, which contains the information about the advertisements to be displayed.

Extensible Markup Language (XML) is a W3C standard for text document markup. It is a text-based markup language that enables you to store data in a structured format by using meaningful tags. The term 'extensible' implies that you can extend your ability to describe a document by defining meaningful tags for the application.

XML is not a language in itself, like HTML, but a set of rules for creating new markup languages. It is a meta-markup language. It allows developers to create custom tag sets for special uses. It structures, stores, and transports the information.

The following code illustrates an advertisement file ads.xml:

```
<Advertisements>
   <Ad>
      <ImageUrl>rose1.jpg</ImageUrl>
      <NavigateUrl>http://www.1800flowers.com</NavigateUrl>
      <AlternateText>
         Order flowers, roses, gifts and more
      </AlternateText>
      <Impressions>20</Impressions>
      <Keyword>flowers</Keyword>
   </Ad>

   <Ad>
      <ImageUrl>rose2.jpg</ImageUrl>
      <NavigateUrl>http://www.babybouquets.com.au</NavigateUrl>
      <AlternateText>Order roses and flowers</AlternateText>
      <Impressions>20</Impressions>
      <Keyword>gifts</Keyword>
   </Ad>

   <Ad>
      <ImageUrl>rose3.jpg</ImageUrl>
      <NavigateUrl>http://www.flowers2moscow.com</NavigateUrl>
      <AlternateText>Send flowers to Russia</AlternateText>
      <Impressions>20</Impressions>
      <Keyword>russia</Keyword>
   </Ad>

   <Ad>
      <ImageUrl>rose4.jpg</ImageUrl>
      <NavigateUrl>http://www.edibleblooms.com</NavigateUrl>
      <AlternateText>Edible Blooms</AlternateText>
      <Impressions>20</Impressions>
      <Keyword>gifts</Keyword>
   </Ad>
</Advertisements>
```

Create a new web page and place an AdRotator control on it.

```
<form id="form1" runat="server">
   <div>
      <asp:AdRotator ID="AdRotator1" runat="server" AdvertisementFile
="~/ads.xml" onadcreated="AdRotator1_AdCreated" />
   </div>
</form>
```

The ads.xml file and the image files should be located in the root directory of the web site.

6. Explain File upload control with example.

ASP.NET has two controls that allow users to upload files to the web server. Once the server receives the posted file data, the application can save it, check it, or ignore it. The following controls allow the file uploading:

- **HtmlInputFile** - an HTML server control
- **FileUpload** - and ASP.NET web control

Both controls allow file uploading, but the FileUpload control automatically sets the encoding of the form, whereas the HtmlInputFile does not do so.
The FileUpload control allows the user to browse for and select the file to be uploaded, providing a browse button and a text box for entering the filename.

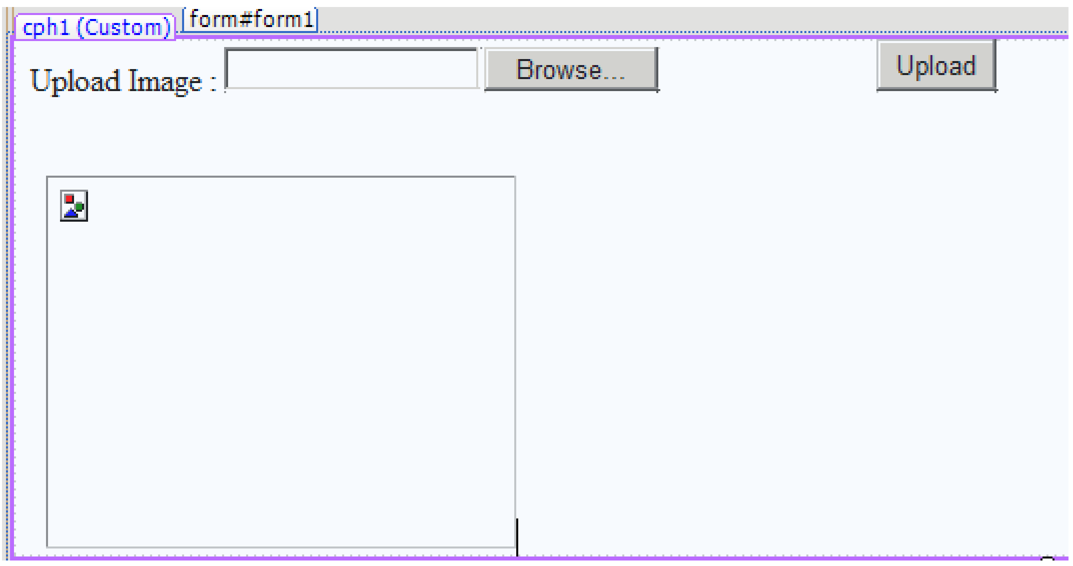
Once, the user has entered the filename in the text box by typing the name or browsing, the SaveAs method of the FileUpload control can be called to save the file to the disk.

The basic syntax of FileUpload is:

```
<asp:FileUpload ID= "Uploader" runat = "server" />
```

Example

The following example demonstrates the FileUpload control and its properties. The form has a FileUpload control along with a save button and a label control for displaying the file name, file type, and file length.
In the design view, the form looks as follows:



The content file code is as given:

```
<form id="form1" runat="server">
    Upload Image :
      <asp:FileUpload ID="FileUpload1" runat="server" Width="290px" />
      <asp:Button ID="btnupload" runat="server" Text="Upload" />
      <asp:Image ID="imgPic" runat="server" Height="186px" Width="235px" />
</form>
```

The code behind the save button is as given:

```
    Protected Sub btnupload_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles btnupload.Click
        Dim strSavedFilePath As String
        strSavedFilePath = Server.MapPath("Images")

        If FileUpload1.HasFile = True Then
            strSavedFilePath = strSavedFilePath & "\" &
      FileUpload1.FileName
            FileUpload1.SaveAs(strSavedFilePath)
            imgPic.ImageUrl = "~/Images/" & FileUpload1.FileName
        End If
    End Sub
```