## Rich Controls

ASP.NET provides large set of controls. These controls are divided into different categories, depends upon their functionalities. The followings control comes under the rich controls category.

* FileUpload control
* Calendar control
* AdRotator control
* MultiView control
* Wizard control

#### **FileUpload control**

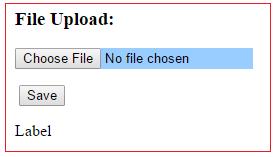
FileUpload control is used to browse and upload files. After the file is uploaded, you can store the file on any drive or database. FileUpload control is the combination of a browse button and a text box for entering the filename.

**The FileUpload control supports the following important properties.**

* **FileBytes:** It returns the contents of uploaded file as a byte array
* **FileContent:** You can get the uploaded file contents as a stream.
* **FileName:** Provides the name of uploaded file.
* **HasFile:** It is a Boolean property that checks whether particular file is available or not.
* **PostedFile:** Gets the uploaded file wrapped in the HttpPostedFile object.

#### Example

using System;  
using System.Text;  
public partial class RichControl : System.Web.UI.Page  
{  
 protected void Page\_Load(object sender, EventArgs e)  
 {  
 }  
 protected void btnSave\_Click(object sender, EventArgs e)  
 {  
 StringBuilder sb = new StringBuilder();  
 if (FileUpload1.HasFile)  
 {  
 try  
 {  
 sb.AppendFormat(" Uploaded file: {0}", FileUpload1.FileName);  
 //save the file  
 FileUpload1.SaveAs(@"C:\" + FileUpload1.FileName);  
 //Showing the file information  
 sb.Append("<br/> File Name: {0}" + FileUpload1.PostedFile.FileName);  
 sb.Append("<br/> File type: {0}"+ FileUpload1.PostedFile.ContentType);   
 sb.Append("<br/> File length: {0}" + FileUpload1.FileBytes.Length);   
 Label1.Text = sb.ToString();   
 }  
 catch (Exception ex)  
 {  
 sb.Append("<br/> Error <br/>");  
 sb.Append(ex.Message);  
 Label1.Text = sb.ToString();  
 }  
 }  
 else  
 {  
 Label1.Text = sb.ToString();  
 }  
 }  
}



#### **Uploading Large Files**

You can upload files up to 4MB using FileUpload control. If you want to upload the files larger than 4MB, then you cannot do with default settings. If you want to upload a file more than four megabytes of data, you need to change this setting. You need to configure the httpRuntime maxRequestLength and httpRuntime requestLengthDiskThreshold settings. This number is in KB.

<configuration>  
 <system.web>   
 <httpRuntime  
 maxRequestLength="10240"  
 requestLengthDiskThreshold="100" />  
 </system.web>  
</configuration>

The above web configuration file can upload up to 10MB file data.

#### **Calendar control**

Calendar control provides you lots of property and events. By using these properties and events you can perform the following task with calendar control.

* Select date.
* Selecting a day, a week or a month.
* Customize the calendar's appearance.

**The Calendar control supports three important events:**

| **Event** | **Description** |
| --- | --- |
| **SelectionChanged** | This event is fired when you select a day, a week or an entire month. |
| **DayRender** | This event is fired when each data cell of the calendar control is rendered. |
| **VisibleMonthChanged** | It is raised when user changes a month. |

Calendar control supports SelectionMode property that allows you to select a single day, week, or entire month.

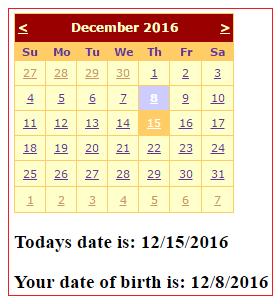
#### Example

using System;  
using System.Text;  
public partial class RichControl : System.Web.UI.Page  
{  
 protected void Page\_Load(object sender, EventArgs e)  
 {  
 }   
 protected void Calendar1\_SelectionChanged(object sender, EventArgs e)  
 {  
 Label1.Text ="Todays date is: "+ Calendar1.TodaysDate.ToShortDateString();  
 Label2.Text = "Your date of birth is: " + Calendar1.SelectedDate.ToShortDateString();  
 }  
}

When you select a date, SelectionChanged event will fired and displays the date in a label controls.

In this example the date format is MM/DD/YYYY.

**Output:**



#### **AdRotator control**

AdRotator control is used to display different advertisements randomly in a page. The list of advertisements is stored in either an XML file or in a database table. Lots of websites uses AdRotator control to display the advertisements on the web page.

To create an advertisement list, first add an XML file to your project.

#### Code for XML file

<?xml version="1.0" encoding="utf-8" ?>  
<Advertisements>  
 <Ad>  
 <ImageUrl>∼ /Images/logo1.png</ImageUrl>  
 <NavigateUrl>http://www.TutorialRide.com</NavigateUrl>  
 <AlternateText>Advertisement</AlternateText>  
 <Impressions>100</Impressions>  
 <Keyword>banner</Keyword>   
 </Ad>  
 <Ad>  
 <ImageUrl>∼ /Images/logo2.png</ImageUrl>  
 <NavigateUrl>http://www.TutorialRide.com</NavigateUrl>  
 <AlternateText>Advertisement</AlternateText>  
 <Impressions>100</Impressions>  
 <Keyword>banner</Keyword>   
 </Ad>  
 <Ad>  
 <ImageUrl>∼ /Images/logo3.png</ImageUrl>  
 <NavigateUrl>http://www.CareerRide.com</NavigateUrl>  
 <AlternateText>Advertisement</AlternateText>  
 <Impressions>100</Impressions>  
 <Keyword>banner</Keyword>   
 </Ad>  
 <Ad>  
 <ImageUrl>∼ /Images/logo4.png</ImageUrl>  
 <NavigateUrl>http://www.TutorialRide.com</NavigateUrl>  
 <AlternateText>Advertisement</AlternateText>  
 <Impressions>50</Impressions>  
 <Keyword>banner</Keyword>   
 </Ad>  
</Advertisements>

In the given XML file 'Images' is the name of the folder, where we stored all the images to display.

Now set the AdRotator control's AdvertisementFile property. Set the path of the XML file that you created above to AdRotator control's AdvertisementFile property.

**Important properties of AdRotator control.**

* **ImageUrl:** The URL of the image that will be displayed through AdRotator control.
* **NavigateUrl:** If the user clicks the banner or ad then the new page is opened according to given URL.
* **AlternateText:** It is used for displaying text instead of the picture if picture is not displayed. It is also used as a tooltip.
* **Impressions:** It is a number that sets how frequently an advertisement will appear.
* **Keyword:** It is used to filter ads or identifies a group of advertisement.

#### **MultiView control**

MultiView control can be used when you want to create a tabbed page. In many situations, a web form may be very long, and then you can divide a long form into multiple sub forms. MultiView control is made up of multiple view controls. You can put multiple ASP.NET controls inside view controls. One View control is displayed at a time and it is called as the active view. View control does not work separately. It is always used with a Multiview control.

If working with Visual Studio 2010 or later, you can drag and drop a MultiView control onto the form. You can drag and drop any number of View controls inside the MultiView control. The number of view controls is depends upon the need of your application.

**The MultiView control supports the following important properties**

* **ActiveViewIndex:** It is used to determine which view will be active or visible.
* **Views:** It provides the collection of View controls contained in the MultiView control.

For understand the Multiview control, first we will create a user interface as given below.

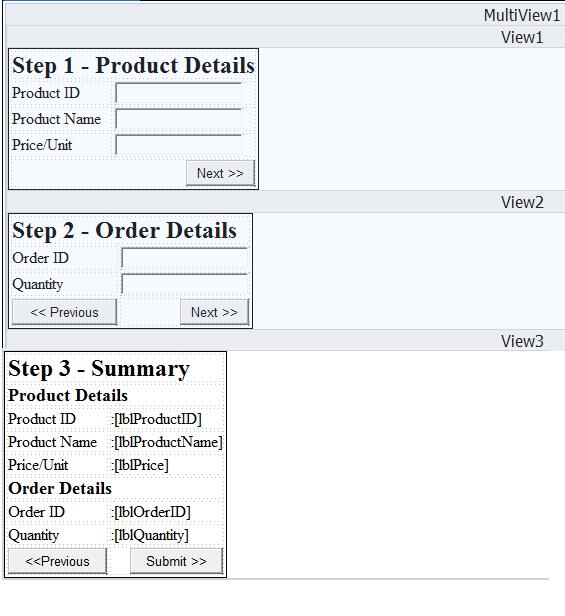
In the given example, in Multiview control, we have taken three separate View control.

**1.** In First step we will design to capture Product details

**2.** In Second step we will design to capture Order details

**3.** Next we will show summary for confirmation.

Here we have not used any database programming to save the data into the database. We will show only the confirmation page, that data has been saved.



#### MultiViewControlDemo.aspx file

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="RichControl.aspx.cs" Inherits="RichControl" %>  
  
<! DOCTYPE html>  
  
<html xmlns="http://www.w3.org/1999/xhtml">  
<head runat="server">  
 <title></title>  
</head>  
<body>  
 <form id="form1" runat="server">  
 <div>  
 <asp:MultiView ID="MultiView1" runat="server">  
 <asp:View ID="View1" runat="server">  
 <table style="border:1px solid black">  
 <tr>  
 <td colspan="2">  
 <h2>Step 1 - Product Details</h2>  
 </td>  
 </tr>  
 <tr>  
 <td>Product ID</td>  
 <td>  
 <asp:TextBox ID="txtProductID" runat="server"></asp:TextBox>  
 </td>  
 </tr>  
 <tr>  
 <td>Product Name</td>  
 <td>  
 <asp:TextBox ID="txtProductName" runat="server"></asp:TextBox>  
 </td>  
 </tr>  
 <tr>  
 <td>Price/Unit</td>  
 <td>  
 <asp:TextBox ID="txtProductPrice" runat="server"></asp:TextBox>  
 </td>  
 </tr>  
 <tr>  
 <td colspan="2" style="text-align:right">  
 <asp:Button ID="btnStep2" runat="server"  
 Text="Next >>" onclick="btnStep2\_Click" />  
 </td>  
 </tr>  
 </table>  
 </asp:View>  
 <asp:View ID="View2" runat="server">  
 <table style="border:1px solid black">  
 <tr>  
 <td colspan="2">  
 <h2>Step 2 - Order Details</h2>  
 </td>  
 </tr>  
 <tr>  
 <td>Order ID</td>  
 <td>  
 <asp:TextBox ID="txtOrderID" runat="server"></asp:TextBox>  
 </td>  
 </tr>  
 <tr>  
 <td>Quantity</td>  
 <td>  
 <asp:TextBox ID="txtQuantity" runat="server"></asp:TextBox>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <asp:Button ID="btnBackToStep1" runat="server" Text="<< Previous"  
 onclick="btnBackToStep1\_Click" />  
 </td>  
 <td style="text-align:right">  
 <asp:Button ID="btnStep3" runat="server" Text="Next >>"  
 onclick="btnGoToStep3\_Click" />  
 </td>  
 </tr>  
 </table>  
 </asp:View>  
 <asp:View ID="View3" runat="server">  
 <table style="border:1px solid black">  
 <tr>  
 <td colspan="2"><h2>Step 3 - Summary</h2></td>  
 </tr>  
 <tr>  
 <td colspan="2"><h3>Product Details</h3></td>  
 </tr>  
 <tr>  
 <td>Product ID</td>  
 <td>  
 <asp:Label ID="lblProductID" runat="server"></asp:Label>  
 </td>  
 </tr>  
 <tr>  
 <td>Product Name</td>  
 <td>  
 <asp:Label ID="lblProductName" runat="server"></asp:Label>  
 </td>  
 </tr>  
 <tr>  
 <td>Price/Unit</td>  
 <td>  
 <asp:Label ID="lblPrice" runat="server"></asp:Label>  
 </td>  
 </tr>  
 <tr>  
 <td colspan="2"><h3>Order Details</h3></td>  
 </tr>  
 <tr>  
 <td>Order ID</td>  
 <td>  
 <asp:Label ID="lblOrderID" runat="server"></asp:Label>  
 </td>  
 </tr>  
 <tr>  
 <td>Quantity</td>  
 <td>  
 <asp:Label ID="lblQuantity" runat="server"></asp:Label>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <asp:Button ID="btnBackToStep2" runat="server" OnClick="btnBackToStep2\_Click" style="height: 26px" Text="<<Previous" />  
 </td>  
 <td style="text-align:right">  
 <asp:Button ID="btnSubmit" runat="server" Text="Submit >>" OnClick="btnSubmit\_Click"  
 />  
 </td>  
 </tr>  
 </table>  
 </asp:View>  
 </asp:MultiView>   
 </div>  
 </form>  
</body>  
</html>

#### MultiViewControlDemo.aspx.cs file

using System;  
using System.Text;  
public partial class RichControl : System.Web.UI.Page  
{  
 protected void Page\_Load(object sender, EventArgs e)  
 {  
 if (! IsPostBack)  
 {  
 MultiView1.ActiveViewIndex = 0;  
 }  
 }  
 protected void btnStep2\_Click(object sender, EventArgs e)  
 {  
 MultiView1.ActiveViewIndex = 1;  
 }  
 protected void btnBackToStep1\_Click(object sender, EventArgs e)  
 {  
 MultiView1.ActiveViewIndex = 0;  
 }  
 protected void btnGoToStep3\_Click(object sender, EventArgs e)  
 {   
 MultiView1.ActiveViewIndex = 2;  
 lblProductID.Text = txtProductID.Text;  
 lblProductName.Text = txtProductName.Text;  
 lblPrice.Text = txtProductPrice.Text;  
 lblOrderID.Text = txtOrderID.Text;  
 lblQuantity.Text = txtQuantity.Text;  
 }   
 protected void btnSubmit\_Click(object sender, EventArgs e)  
 {  
 Response.Redirect("SaveData.aspx");  
 }  
 protected void btnBackToStep2\_Click(object sender, EventArgs e)  
 {  
 MultiView1.ActiveViewIndex = 1;  
 }  
}

ActiveViewIndex property of MultiView control is zero based.

#### **Wizard Control**

This control is same as MultiView control but the main difference is that, it has inbuilt navigation buttons.

The wizard control enables you to design a long form in such a way that you can work in multiple sub form. You can perform the task in a step by step process. It reduces the work of developers to design multiple forms. It enables you to create multi step user interface. Wizard control provides with built-in previous/next functionality.

The Wizard control can contains one or more WizardStep as child controls. Only one WizardStep is displayed at a time. WizardStep control has an important property called as StepType. The StepType property determines the type of navigation buttons that will be displayed for that step. The possible values are:

The StepType associated with each WizardStep determines the type of navigation buttons that will be displayed for that step. **The StepTypes are:**

* Start:
* Step:
* Finish:
* Complete:
* Auto:

Drag the Wizard control on the web page from toolbox, you will get the following code.

<asp:Wizard ID="Wizard1" runat="server" Height="75px" Width="140px">  
 <WizardSteps>  
 <asp:WizardStep runat="server" title="Step 1">  
 </asp:WizardStep>  
 <asp:WizardStep runat="server" title="Step 2">  
 </asp:WizardStep>  
 </WizardSteps>  
</asp:Wizard>

You can put WizardStep according to application need.

**Important events of Wizard control are as follows:**

* ActiveStepChanged:
* CancelButtonClick:
* FinishButtonClick:
* NextButtonClick:
* PreviousButtonClick:

Now we will create an application as we had done with MultiView control. We will create three different WizardStep in Wizard control.

**1.** In First step we will design to capture Product details

**2.** In Second step we will design to capture Order details

**3.** Next we will show summary for confirmation.





#### WizardControlDemo.aspx.cs file

using System;  
using System.Web.UI.WebControls;  
  
public partial class WizardControl : System.Web.UI.Page  
{  
 protected void Page\_Load(object sender, EventArgs e)  
 {   
 }  
 protected void Wizard1\_FinishButtonClick(object sender, WizardNavigationEventArgs e)  
 {  
 Response.Redirect("SaveData.aspx");  
 }  
 protected void Wizard1\_NextButtonClick(object sender, WizardNavigationEventArgs e)  
 {  
 if (e.NextStepIndex == 2)  
 {  
 lblProductID.Text = txtProductID.Text;  
 lblProductName.Text = txtProductName.Text;  
 lblPrice.Text = txtProductPrice.Text;  
 lblOrderID.Text = txtOrderID.Text;  
 lblQuantity.Text = txtQuantity.Text;  
 }  
 }  
}

Register the event for Next button by using property window with event tab. In the given example, for going third step from second we have to set e.NextStepIndex == 2. Here StepIndex is zero based.