FormView Web Server Control Overview

The FormView control is used to display a single record at a time from a data source. When you use the FormView control, you create templates to display and edit data-bound values. The templates contain controls, binding expressions, and formatting that define the look and functionality of the form. The FormView control is often used in combination with a GridView control for master/detail scenarios.

This topic contains:

Background

The FormView control lets you work with a single record from a data source, similar to the DetailsView control. The difference between the FormView and the DetailsView controls is that the DetailsView control uses a tabular layout where each field of the record is displayed as a row of its own. In contrast, the FormView control does not specify a pre-defined layout for displaying the record. Instead, you create a template that contains controls to display individual fields from the record. The template contains the formatting, controls, and binding expressions used to create the form.

The FormView control is typically used for updating and inserting new records. It is often used in master/detail scenarios where the selected record of the master control determines the record to display in the FormView control. For more information and an example, see Modifying Data Using a FormView Web Server Control.

The FormView control relies on the capabilities of the data source control to perform tasks such as updating, inserting, and deleting records. The FormView control displays only a single data record at a time, even if its data source exposes multiple records.

The FormView control can automatically page over the data in its associated data source one record at a time. This requires that the data is represented by an object that implements the ICollection interface, or that the underlying data source supports paging. The FormView control provides the user interface (UI) for navigating between records. To enable paging behavior, set the AllowPaging property to **true** and specify a PagerTemplate value.

The FormView control exposes several events that you can handle to execute your own code. The events are raised before and after insert, update, and delete operations of the associated data source control. You can also write handlers for the ItemCreated and ItemCommand events.

Note
The event model of the FormView control is like that of the GridView control. However, the FormView control does not support a selection event, because the current record is always the selected item.

Data Binding with the FormView Control

The FormView control gives you these options for binding to data:

- Data binding using the DataSourceID property, which enables you to bind the FormView control to a data source control. This is the recommended approach because it enables the FormView control to take advantage of the capabilities of the data source control and to provide built-in functionality for updating and paging.
- Data binding using the DataSource property, which enables you to bind to various objects, including ADO.NET datasets and data readers. This approach requires that you write the code for any additional functionality such as updating and paging.

When you bind to a data source using the DataSourceID property, the FormView control supports two-way data binding. In addition to the control displaying data, you can enable the control to automatically support insert, update, and delete operations on the bound data.

For more information, see Data Source Web Server Controls.

Creating the FormView Control User Interface

You build the user interface (UI) for the FormView control by creating templates. You specify different templates for different actions. You create an ItemTemplate template for display, insert, and edit modes. You can control paging using a PagerTemplate template, and you can customize the FormView control's header and footer using a HeaderTemplate and FooterTemplate, respectively. By using an EmptyDataTemplate, you can also specify a template to display when the data source returns no data. For more information, see Creating Templates for the FormView Web Server Control.

The item templates that you create for the FormView control specify the content of the control. As with the DetailsView control, you can also customize the display format of the FormView control by using style properties such as the EditRowStyle, EmptyDataRowStyle, FooterStyle, HeaderStyle, InsertRowStyle, PagerStyle, and RowStyle properties.

The following example shows an ASP.NET page that uses a FormView control to display data.

C#

```
<%@ Page language="C#" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
  <head runat="server">
    <title>FormView Example</title>
</head>
<body>
    <form id="form1" runat="server">
      <h3>FormView Example</h3>
        <table cellspacing="10">
          <tr>
            <td valign="top">

              <asp:FormView ID="ProductsFormView"
                DataSourceID="ProductsSqlDataSource"
                AllowPaging="true"
                runat="server">
```

```
                <HeaderStyle forecolor="white" backcolor="Blue" />

                <ItemTemplate>
                  <table>
                    <tr>
                      <td align="right"><b>Product ID:</b></td>
                      <td><asp:Label id="ProductIDLabel" runat="server" Text='<%#
Eval("ProductID") %>' /></td>
                    </tr>
                    <tr>
                      <td align="right"><b>Product Name:</b></td>
                      <td><asp:Label id="ProductNameLabel" runat="server" Text='<%#
Eval("ProductName") %>' /></td>
                    </tr>
                    <tr>
                      <td align="right"><b>Category ID:</b></td>
                      <td><asp:Label id="CategoryIDLabel" runat="server" Text='<%#
Eval("CategoryID") %>' /></td>
                    </tr>
                    <tr>
                      <td align="right"><b>Quantity Per Unit:</b></td>
                      <td><asp:Label id="QuantityPerUnitLabel" runat="server"
Text='<%# Eval("QuantityPerUnit") %>' /></td>
                    </tr>
                    <tr>
                      <td align="right"><b>Unit Price:</b></td>
                      <td><asp:Label id="UnitPriceLabel" runat="server" Text='<%#
Eval("UnitPrice") %>' /></td>
                    </tr>
                  </table>
                </ItemTemplate>

                <PagerTemplate>
                  <table>
                    <tr>
                      <td><asp:LinkButton ID="FirstButton" CommandName="Page"
CommandArgument="First" Text="<<" RunAt="server"/></td>
                      <td><asp:LinkButton ID="PrevButton"  CommandName="Page"
CommandArgument="Prev"  Text="<"  RunAt="server"/></td>
                      <td><asp:LinkButton ID="NextButton"  CommandName="Page"
CommandArgument="Next"  Text=">"  RunAt="server"/></td>
                      <td><asp:LinkButton ID="LastButton"  CommandName="Page"
CommandArgument="Last"  Text=">>" RunAt="server"/></td>
                    </tr>
                  </table>
                </PagerTemplate>

              </asp:FormView>

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

        <asp:SqlDataSource ID="ProductsSqlDataSource"
          SelectCommand="SELECT ProductID, ProductName, CategoryID,
QuantityPerUnit, UnitPrice FROM [Products]"
          connectionstring="<%$ ConnectionStrings:NorthwindConnection %>"
```

```
            RunAt="server" />

        </form>
    </body>
</html>
```

By default, the FormView control displays its contents by using an HTML table. This can make it difficult to apply CSS style to the contents of the control. You can configure the FormView control not to render HTML table tags by setting the RenderOuterTable property to **false**. For more information, see Creating Templates for the FormView Web Server Control.

Back to top

Code Examples

Creating Templates for the FormView Web Server Control

Paging in a FormView Web Server Control

Modifying Data Using a FormView Web Server Control

Walkthrough: Displaying Formatted Data in Web Pages with the FormView Web Server Control

Back to top

Class Reference

The following table lists the key classes that relate to the FormView control.

Member	Description
FormView	The main class for the control.

Back to top

See Also

Concepts
ASP.NET Web Server Controls Templates
Other Resources
Data Toolbox Controls
ASP.NET Data Access Options