

ASP.NET 2.0 Data Control Tips and Tricks

Paul Litwin

Collaborative Data Services (CDS)
Fred Hutchinson Cancer Research Center

plitwin@fhcrc.org

CDS Brownbag Series

- This is the restart of the popular CDS brownbag series
- Materials for the series can be downloaded from cfs.fhcrc.org
- Sign up to be on our email list at cfs.fhcrc.org

CDS

- Collaborative Data Services
 - Providing "data services" through FHCRC (and beyond)
 - Our services include...
 - Telephone interviewing of subjects
 - Data entry & scanning
 - Programming
 - Web and database hosting
- More info at <http://cfs.fhcrc.org>

Slides & Samples Download

- You can download them from:
 - cbs.fhcrc.org

Agenda

- ASP 2.0 Data Binding Overview
- Programming GridView
- Programming DetailsView and FormView
- Data Control How To's

ASP 2.0 Data Binding

Two Types Of Data-related Controls

- **Data-bound Controls**

- UI controls that render data
 - GridView, DetailsView, FormView, TreeView, Menu
- Can auto-bind to data exposed from a data source control (via DataSourceID)
- Will fetch data at the appropriate time in lifecycle
- Can optionally take advantage of data source capabilities
- Can also work directly with Datasets (via DataSource)

- **Data Source Controls**

- Non-UI controls that serve as middleman/woman between data-bound controls and data source
 - SQLDataSource, AccessDataSource, ObjectDataSource, XmlDataSource, SiteMapDataSource
- Represent different backend data sources
 - SQL Databases, Access, Business Objects, XML/Web Services, SiteMap
 - Can easily extend by writing custom data sources
- Provide rich capabilities over data
 - Sorting, paging, filtering, updating, deleting, inserting
- Expose data through tabular or hierarchical interfaces to data-bound controls

Programming GridView

GridView

```
<asp:GridView ID="gvCustomers" runat="server"
DataSourceID="sdsCustomers" DataKeyNames="CustomerID">
  <Columns>
    <asp:CommandField ShowDeleteButton="True" ShowEditButton="True" />
    <asp:BoundField DataField="CustomerID" HeaderText="CustomerID" ReadOnly="True"
      SortExpression="CustomerID" />
    <asp:BoundField DataField="City" HeaderText="City" SortExpression="City" />
  </Columns>
</asp:GridView>
```

```
<asp:SqlDataSource ID="sdsCustomers" runat="server"
ConnectionString="<%$ ConnectionStrings:cnxNwind %>"
SelectCommand="SELECT [CustomerID], [CompanyName], [ContactName], [City],
[Country] FROM [Customers]"
DeleteCommand="DELETE FROM [Customers] WHERE [CustomerID] = @CustomerID"
...>
  <DeleteParameters>
    <asp:Parameter Name="original_CustomerID" Type="String" />
  </DeleteParameters>
...
</asp:SqlDataSource>
```

Programming GridView: Things You Need to Know

1. How to hook in

- What events are most useful
- When to use events of GridView vs. Data Source Control

2. How to programmatically access controls and fields

GridView Events of Note

- **DataBound**
 - Use to modify data as it is being loaded into control, e.g., setting a default value
- **RowUpdating**
 - Use to manipulate data before row is updated in database, e.g., saving a timestamp field.
- **RowDeleting**
 - Use to manipulate data before row is deleted from database, e.g., stopping delete.
- **RowUpdated**
 - Use to react to database update, e.g. refreshing another control on page or handling update exceptions.
- **RowDeleted**
 - Use to react to database delete, e.g. refreshing another control on page or handling delete exceptions

Programming DetailsView and FormView

The Single-Record Form Problem

- ASP.NET 1.x provided no help to the developer having to create a single-record “form” view of data
- ASP.NET 2.0 gives us two solutions
 - DetailsView – places fields in a table
 - uses BoundFields by default (with no ID property)
 - FormView – uses a free-form template
 - uses template fields by default
 - more flexibility
- Both support editing, inserts, pagination, template fields, validation controls

DetailsView

```
<asp:DetailsView ID="dvCustomers" runat="server" ...
  DataKeyNames="CustomerID" DataSourceID="sdsCustomers"
>
<Fields>
  <asp:BoundField DataField="CustomerID" HeaderText="CustomerID" ReadOnly="True"
SortExpression="CustomerID" />
  <asp:BoundField DataField="CompanyName" HeaderText="CompanyName"
SortExpression="CompanyName" />
  <asp:CommandField ShowDeleteButton="True" ShowEditButton="True" ShowInsertButton="True" />
</Fields>
</asp:DetailsView>

<asp:SqlDataSource ID="sdsCustomers" runat="server"
  ConnectionString="<%= $ConnectionStrings.cnxNwind %>"
  SelectCommand="SELECT [CustomerID], [CompanyName], ... FROM [Customers] ORDER BY
[CompanyName]"
  DeleteCommand="DELETE FROM [Customers] WHERE [CustomerID] = @CustomerID"
  InsertCommand="INSERT INTO [Customers] ([CustomerID], ...) VALUES (@CustomerID, ...)"
  UpdateCommand="UPDATE [Customers] SET [CompanyName] = @CompanyName, ... WHERE [CustomerID]
= @CustomerID">
  <UpdateParameters>
    <asp:Parameter Name="CompanyName" Type="String" />
    <asp:Parameter Name="ContactName" Type="String" />
    <asp:Parameter Name="ContactTitle" Type="String" />
  </UpdateParameters>
</asp:SqlDataSource>
```

FormView

```
<asp:FormView ID="fvCustomers" runat="server" ...  
  DataKeyNames="CustomerID" DataSourceID="sdsCustomers"  
>  
  <ItemTemplate>  
    <strong><span style="color: white; background-color: #9471DE">CustomerID:  
    <asp:Label ID="CustomerIDLabel" runat="server" Text='<%=# Eval("CustomerID") %>'></asp:Label>  
    </span></strong><br />  
    CompanyName:  
    <asp:Label ID="CompanyNameLabel" runat="server" Text='<%=# Bind("CompanyName") %>'>  
    </asp:Label><br />  
  ...  
</ItemTemplate>  
  <EditItemTemplate>  
    CustomerID:  
    <asp:Label ID="CustomerIDLabel1" runat="server" Text='<%=# Eval("CustomerID") %>'>  
    </asp:Label><br />  
    CompanyName:  
    <asp:TextBox ID="CompanyNameTextBox" runat="server" Text='<%=# Bind("CompanyName") %>'>  
    </asp:TextBox><asp:RequiredFieldValidator ID="rfvCompanyName"  
    ControlToValidate="CompanyNameTextBox" runat="server"  
    ErrorMessage="Company Name is required."></asp:RequiredFieldValidator>  
    <br />  
  ...  
</asp:FormView>
```

Programming DetailsView and FormView: Things You Need to Know

1. How to hook in

- What events are most useful
- When to use events of Data Control vs. Data Source Control

2. How to programmatically access controls and fields

Details/Form View Events of Note (1 of 2)

DataBound and the “ing” Events

- **DataBound**
 - Use to modify data as it is being loaded into control, e.g., setting a default value
- **ItemUpdating**
 - Use to manipulate data before row is updated in database, e.g., saving a timestamp field.
- **ItemInserting**
 - Use to manipulate data before row is inserted to database, e.g., saving a timestamp field.
- **ItemDeleting**
 - Use to manipulate data before row is deleted from database, e.g., stopping delete.

Details/Form View Events of Note (2 of 2)

The “ed” Events

- **ItemUpdated**
 - Use to react to database update, e.g. refreshing another control on page or handling update exceptions.
- **ItemInserted**
 - Use to react to database insert, e.g. refreshing another control on page or handling insert exceptions
- **ItemDeleted**
 - Use to react to database delete, e.g. refreshing another control on page or handling delete exceptions
- **DataSourceControl Inserted**
 - Useful for grabbing Identity field value and refreshing page to display new record
 - Note: this is an event of the DataSource control, not the Data control!

Data Control How To's

How do I...

1. Filter one data control based on another?
2. Set a default value of a control for an insert?
3. Confirm a record delete for a data control?
4. Use a drop-down control for editing with a data control?
5. Update fields that aren't shown (e.g., time stamp fields) when a row is updated?
6. Refresh one control when another changes?
7. Not pass an identity column during an insert to the database? ...and... After the insert, retrieve the newly inserted record, complete with identity column?
8. Handle exceptions?

1. How do I...

- Filter one data control based on another?
 - Create a ControlParameter in the second control's Data Source control that points to the selected value of the first control
- Example: [GridViewFiltered.aspx](#)

```
<asp:SqlDataSource ...>  
  <SelectParameters>  
    <asp:ControlParameter ControlID="ddlCountryFilter" Name="Country"  
      PropertyName="SelectedValue"  
      Type="String" />  
  </SelectParameters>  
</asp:SqlDataSource>
```

Different Types of Parameters

- You're not limited to using Control Parameters
- Also available
 - `<asp:QueryStringParameter>`
 - `<asp:CookieParameter>`
 - `<asp:ProfileParameter>`
 - `<asp:SessionParameter>`
 - `<asp:Parameter>`

2. How do I...

- Set a default value of a control for an insert?
- Use DataBound event, checking the CurrentMode property

```
protected void fvCustomers_DataBound(object sender, EventArgs e) {  
    if (fvCustomers.CurrentMode == FormViewMode.Insert) {  
        ((TextBox)fvCustomers.FindControl("CityTextBox")).Text = "Seattle";  
        ((TextBox)fvCustomers.FindControl("CountryTextBox")).Text = "USA";  
    }  
}
```

- Example: FormView.aspx

2a. How do I...

- What about finding control when using DetailsView?
 - Problem: you can't get at controls by name unless you use template field. Solution: use template field

```
<asp:TemplateField HeaderText="Country">
  <ItemTemplate>
    <asp:Label runat="server" ID="lblCountry" Text='<%=#Bind("Country")%>' />
  </ItemTemplate>
  <EditItemTemplate>
    <asp:Textbox runat="server" ID="txtCountryEdit" Text='<%=#Bind("Country")%>' />
  </EditItemTemplate>
  <InsertItemTemplate>
    <asp:Textbox runat="server" ID="txtCountryInsert" Text='<%=#Bind("Country")%>' />
  </InsertItemTemplate>
</asp:TemplateField>
```

- Example: DetailsViewDefault.aspx

3. How do I...

- Confirm a record delete for a data control?
 - Use a templated LinkButton control w/ client-side JavaScript

```
<asp:TemplateField>
  <ItemTemplate>
    <asp:LinkButton ID="lbDelete" Runat="server"
      OnClientClick="return confirm('OK to delete record?');"
      CommandName="Delete"
      Text = "Delete"></asp:LinkButton>
  </ItemTemplate>
</asp:TemplateField>
```

- Example: [GridViewUpdate2DeleteConfirm.aspx](#)

4. How do I...

- Use a drop-down control for editing with a data control?
 - Use a template field

```
<asp:TemplateField HeaderText="ReportsTo" SortExpression="ReportsTo">
  <EditItemTemplate>
    <asp:DropDownList ID="ddlReportsTo" runat="server"
      SelectedValue='< %# Bind("ReportsTo") %>'
      DataSourceID="sdsReportsTo" DataTextField="EmployeeName"
      DataValueField="EmployeeId"></asp:DropDownList>
  </EditItemTemplate>
  <ItemTemplate>
    <asp:Label ID="lblReportsTo" runat="server"
      Text='< %# Bind("ReportsToName") %>'></asp:Label>
  </ItemTemplate>
</asp:TemplateField>
```

Example: [EmployeeGridWithDropDown.aspx](#)

4. How do I...

- Use a drop-down control for editing with a data control? *(continued)*
 - Don't forget to provide a NULL to the list!

```
<asp:SqlDataSource ID="sdsReportsTo" runat="server" ..."  
    SelectCommand=  
    "SELECT EmployeeId,  
    LastName + ', ' + FirstName AS EmployeeName  
    FROM Employees  
    UNION  
    SELECT NULL, NULL  
    ORDER BY LastName + ', ' + FirstName">  
</asp:SqlDataSource>
```

5. How do I...

- Update fields that aren't shown (e.g., time stamp fields) when a row is updated?
 - Template the fields so you can control updatability/visibility during edits/inserts
 - For Insert
 - Use `ItemInserting` event and `Values` collection
 - For Update
 - Use `ItemUpdating` event and `NewValues` collection
- Example: `DetailsViewHiddenFields.aspx`

6. How do I...

- Refresh one control when another changes?
 - Call DataBind() method of second control from first control's "ed" event
- Example: DrillDown1Page.aspx

7. How do I...

- Not pass an identity column during an insert to the database?
 - Set **InsertVisible** property of control to false
- After the insert, retrieve the newly inserted record, complete with identity column?
 - Uses **Inserted** event of **DataSource** control and grab the value of the identity column using `e.Command.Parameters` collection
- Example: [EmpDrillDown.aspx/EmpDetails.aspx](#)

8. How do I...

- Handle Exceptions?
 - Use one of “ed” events & retrieve exception:

```
Exception exp = e.Exception;  
if (exp != null)  
{  
    lblMsg.Text = "Insert failed. Details: " + exp.Message;  
    // If using ObjectDataSource use exp.InnerException.Message instead  
    e.ExceptionHandled = true;  
    e.KeepInInsertMode = true; // keeps user editing.  
}
```

- Example: FormViewExp.aspx

Gotchas to Watch Out For

- Gotcha: Problem updating your control?
 - Check for correct DataKeyNames property of data control
- Gotcha: When hooking up stored procedures to DataSource controls containing output parameters, Visual Studio will set the Direction to “inputoutput” instead of “output” which will create problems
 - Fix: Change direction to “output”
- Gotcha: If a page hides and unhides DetailsView/FormView controls, it may produce a bogus “Failed to load viewstate...” error message
 - Fix: Set “EnableViewState” property to False for each of the DetailsView/FormView controls

Questions?

Thank You!

- Contact: plitwin@fhcrc.org
- Download slides & samples from
 - cfs.fhcrc.org