

## Re: GridView DataBind() Not Working

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.aspnet.buildingcontrols/2007-02/msg>

---

- *From:* [stcheng@xxxxxxxxxxxxxxxxxxxxxx](mailto:stcheng@xxxxxxxxxxxxxxxxxxxxxx) (Steven Cheng[MSFT])
  - *Date:* Wed, 21 Feb 2007 08:54:33 GMT
- 

Thanks for your reply Mark,

yes, based on my research, if you completely add the rendering code logic(both for runtime and design-time) in your control's code(CreateChildcontrols), the designer (of your toplevel custom control) will not render the child GridView. Also, as you discussed in another thread with my colleague Walter, for child controls added into top custom control's controls collection, it hasn't a designer associated like the top level control.

Currently, what I'm still trying is use our own custom control designer and render the control (and its child controls) manually. And I've got the following result so far:

\*in our custom control designer, we can create a new instance of our custom control type and render it(or you can set some additional attributes on it)  
e.g.

```
=====
[DefaultProperty("Text")]
[ToolboxData("<{0}:MyComposite runat=server></{0}:MyComposite>")]
[Designer(typeof(MyCompositeDesigner))]
public class MyComposite : CompositeControl
{
    Label _label;
    GridView _grid;

    [Bindable(true)]
    [Category("Appearance")]
    [DefaultValue("")]
    [Localizable(true)]
    public string Text
    {
        get
        {
            String s = (String)ViewState["Text"];
            return ((s == null) ? String.Empty : s);
        }
    }
}
```

## Re: GridView DataBind() Not Working

```
}

set
{
ViewState["Text"] = value;
}
}

protected override void CreateChildControls()
{
Controls.Clear();

_label = new Label();
_label.ID = "lblMessage";
Controls.Add(_label);

_label.Text = Text;

_grid = new GridView();
_grid.ID = "gvData";
Controls.Add(_grid);

_grid.AutoGenerateColumns = true;

_grid.DataSource = HelperUtil.GetDummyListItems();
_grid.DataBind();

}

}

internal class HelperUtil
{
internal static ListItem[] GetDummyListItems()
{
ListItem[] items = new ListItem[5];
for (int i = 0; i < items.Length; i++)
{
items[i] = new ListItem("key" + i, "value" + i);
}

return items;
}
}
```

## Re: GridView DataBind() Not Working

```
internal class MyCompositeDesigner : CompositeControlDesigner
{
    public override string GetDesignTimeHtml()
    {
        string html = string.Empty;

        MyCompositeControl mc1 = new MyCompositeControl();

        ICompositeControlDesignerAccessor accessor = mc1 as
        ICompositeControlDesignerAccessor;

        accessor.RecreateChildControls();

        StringWriter sw = new StringWriter();
        XhtmlTextWriter hw = new XhtmlTextWriter(sw);

        mc1.RenderControl(hw);

        html += sw.ToString();

        return html;
    }
}
=====
```

However, if I try using the control instance from the designer's "ViewControl" property to render, the child GridView always not display. e.g.

```
=====
public override string GetDesignTimeHtml()
{
    string html = string.Empty;

    WebControl mc1 = new MyComposite();

    mc1 = this.ViewControl as WebControl;

    ICompositeControlDesignerAccessor accessor = mc1 as
    ICompositeControlDesignerAccessor;

    accessor.RecreateChildControls();
```

Re: GridView DataBind() Not Working

```
StringWriter sw = new StringWriter();
XhtmlTextWriter hw = new XhtmlTextWriter(sw);

mc1.RenderControl(hw);

html+= sw.ToString();

return html;
}
=====
```

I will do some further research on this to see whether it does be not supported to use the original instance for manual rendering in designer or if there is any other means. Anyway, at least you can do custom rendering in custom control designer by manually creating a control instance. I'll update you when I get any progress at the earliest time.

Sincerely,

Steven Cheng

Microsoft MSDN Online Support Lead

This posting is provided "AS IS" with no warranties, and confers no rights.

.