

Re: Custom Sorting a DataTable Using DataGrid

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.windowsforms.controls/2004-08/037>

From: ClayB [SynCFusion] (*clayb_at_synCFusion.com*)

Date: 08/18/04

Date: Wed, 18 Aug 2004 16:04:31 -0400

I think you should be able to do this creating a wrapper class for your column that implements IComparable. Here is a little sample that sorts columns based on string length instead of the default sorting.

=====

Clay Burch, .NET MVP

Visit www.synCFusion.com for the coolest tools

```
using System;
```

```
using System.Drawing;
```

```
using System.Collections;
```

```
using System.ComponentModel;
```

```
using System.Windows.Forms;
```

```
using System.Data;
```

```
namespace GDBGDataView
```

```
{
```

```
/// <summary>
```

```
/// Summary description for Form1.
```

```
/// </summary>
```

```
public class Form1 : System.Windows.Forms.Form
```

```
{
```

```
private System.Windows.Forms.DataGrid dataGrid1;
```

```
/// <summary>
```

```
/// Required designer variable.  
  
/// </summary>  
  
private System.ComponentModel.Container components = null;  
  
public Form1()  
{  
//  
// Required for Windows Form Designer support  
//  
InitializeComponent();  
//  
// TODO: Add any constructor code after InitializeComponent call  
//  
}  
  
/// <summary>  
/// Clean up any resources being used.  
/// </summary>  
  
protected override void Dispose( bool disposing )  
{  
if( disposing )  
{  
if (components != null)  
{  
components.Dispose();  
}  
}  
  
base.Dispose( disposing );  

```

```
}  
  
#region Windows Form Designer generated code  
  
/// <summary>  
  
/// Required method for Designer support – do not modify  
  
/// the contents of this method with the code editor.  
  
/// </summary>  
  
private void InitializeComponent()  
  
{  
  
this.dataGrid1 = new System.Windows.Forms.DataGrid();  
  
((System.ComponentModel.ISupportInitialize)(this.dataGrid1)).BeginInit();  
  
this.SuspendLayout();  
  
//  
  
// dataGrid1  
  
//  
  
this.dataGrid1.Anchor =  
((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles.Top  
p | System.Windows.Forms.AnchorStyles.Bottom)  
  
| System.Windows.Forms.AnchorStyles.Left)  
  
| System.Windows.Forms.AnchorStyles.Right)));  
  
this.dataGrid1.DataMember = "";  
  
this.dataGrid1.HeaderForeColor = System.Drawing.SystemColors.ControlText;  
  
this.dataGrid1.Location = new System.Drawing.Point(24, 24);  
  
this.dataGrid1.Name = "dataGrid1";  
  
this.dataGrid1.Size = new System.Drawing.Size(248, 224);  
  
this.dataGrid1.TabIndex = 0;  
  
//  
  
// Form1
```

```
//  
  
this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);  
  
this.ClientSize = new System.Drawing.Size(292, 266);  
  
this.Controls.Add(this.dataGrid1);  
  
this.Name = "Form1";  
  
this.Text = "Form1";  
  
this.Load += new System.EventHandler(this.Form1_Load);  
  
((System.ComponentModel.ISupportInitialize)(this.dataGrid1)).EndInit();  
  
this.ResumeLayout(false);  
  
}  
  
#endregion  
  
/// <summary>  
/// The main entry point for the application.  
/// </summary>  
  
[STAThread]  
  
static void Main()  
  
{  
  
Application.Run(new Form1());  
  
}  
  
private void Form1_Load(object sender, System.EventArgs e)  
  
{  
  
DataTable dt = new DataTable("MyTable");  
  
int nCols = 4;  
  
int nRows = 10;  
  
for(int i = 0; i < nCols; i++)
```

```
dt.Columns.Add(new DataColumn(string.Format("Col{0}", i),  
typeof(MyObject)));
```

```
Random r = new Random();
```

```
for(int i = 0; i < nRows; ++i)
```

```
{
```

```
DataRow dr = dt.NewRow();
```

```
for(int j = 0; j < nCols; j++)
```

```
dr[j] = new MyObject("row" + new string('a', 1 + r.Next(10)));
```

```
dt.Rows.Add(dr);
```

```
}
```

```
this.dataGrid1.DataSource = dt;
```

```
}
```

```
}
```

```
public class MyObject : IComparable
```

```
{
```

```
public MyObject(string s)
```

```
{
```

```
this.s = s;
```

```
}
```

```
public override string ToString()
```

```
{
```

```
return s;
```

```
}
```

```
string s;
```

```
public string Val
```

```
{
```

```
get{return s;}

set{s = value;}

}

public int CompareTo(object obj)

{

return this.ToString().Length - obj.ToString().Length;

}

}

}
```

"Dean Rettig" <rettigcd@bigfoot.com> wrote in message
news:f678be33.0408181107.14159850@posting.google.com...
> *I have a DataTable with several columns. One column is a string that
> represents distances.(1 foot, 5 meters, 3 cm, etc) I want to click on
> the column header and have the rows sort based on the distances: 3cm,
> 1 foot, 5 meters.*
>
> *I know how to create the "sort value" of each distance but I don't
> know how to make the column use a "sort value" instead of the normal
> string value.*
>
> *Research shows lots of people sorting ArrayLists but I also want to do
> normal sorts on other columns.*
>
> *Possible solutions that I've come up with include:*
>
> *1) creating a hidden column that contains the "Sort Value" and trying
> to redirect the table to sort on a different column than I clicked on.*
>
> *2) storing custom objects with the IComparable interface in the
> "distance" column but I can't figure out how to get the
> TextEditControl to load the data into my new object nor display the
> data as text.*
>
> *3) Derive a class from String and override the CompareTo function.*
>
> *4) Write a new GridDataColumnStyle class that displays distance
> information.*
>
> *Are any of these on the right track. I've spent 8 hours on this and
> my head is going to explode if I don't get it figured out soon.*
>
> *thanks,*

microsoft.public.dotnet.framework.windowsforms.controls: Re: Custom Sorting a DataTable Using DataGrid

> *Dean*

>

>

> *am I close?*