How to Naturally Sort a DataView with something like IComparable

Asked 16 years, 3 months ago   Modified 5 years, 4 months ago   Viewed 10k times

4

My DataView is acting funny and it is sorting things alphabetically and I need it to sort things numerically. I have looked all across the web for this one and found many ideas on how to sort it with ICompare, but nothing really solid.

So my questions are

1. How do I implement ICompare on a DataView (Looking for code here).
2. How to correctly decipher from a column full of strings that are actual strings and a column full of numbers(with commas).

I need code to help me out with this one guys. I am more or less lost on the idea of ICompare and how to implement in different scenarios so an over all good explanation would be great.

Also, Please don't hand me links. I am looking for solid answers on this one.

Some Code that I use.

```
    DataView dataView = (DataView)Session["kingdomData"];
    dataView.Sort = e.SortExpression + " " +
ConvertSortDirectionToSql(e.SortDirection);
    gvAllData.DataSource = dataView;
    gvAllData.DataBind();



    private string ConvertSortDirectionToSql(SortDirection sortDirection)
{
    string newSortDirection = String.Empty;
    if (Session["SortDirection"] == null)
    {
        switch (sortDirection)
        {
            case SortDirection.Ascending:
                newSortDirection = "ASC";
                break;
            case SortDirection.Descending:
                newSortDirection = "DESC";
                break;
        }
    }
    else
    {
        newSortDirection = Session["SortDirection"].ToString();
        switch (newSortDirection)
        {
            case "ASC":
                newSortDirection = "DESC";
                break;
            case "DESC":
                newSortDirection = "ASC";
                break;
        }
    }
    Session["SortDirection"] = newSortDirection;
    return newSortDirection;
}
```

For the scenario, I build a datatable dynamically and shove it into a dataview where I put the dataview into a gridview while also remembering to put the dataview into a session object for sorting capabilities.

When the user calls on the gridview to sort a column, I recall the dataview in the session object and build the dataview sorting expression like this:

```
dataview.sort = e.sortexpression + " " + e.Sortdirection;
```

Or something along those lines. So what ussually comes out is right for all real strings such as

Car; Home; scott; zach etc...

But when I do the same for number fields WITH comma seperated values it comes out something like

900; 800; 700; 600; 200; 120; 1,200; 12,340; 1,000,000;

See what I mean? It just sorts the items as an alpha sort instead of a Natural sort. I want to make my Dataview NATURALLY sort the numeric columns correctly like

120; 200; 600; 700; 800; 900; 1,200; 12,340; 1,000,000;

Let me know what you can do to help me out.
P.S. I have looked through countless articles on how to do this and all of them say to shove into a List/Array and do it that way, but is there a much more efficient way?

.net asp.net dataview icomparable

edited Aug 21, 2019 at 15:51

community wiki
8 revs, 3 users 75%
SpoiledTechie.com

Share

Improve this question

Follow

2 Answers

Sorted by: Highest score (default)

3

For the first issue - IIRC you can't sort a DataView with a comparer. If you just need to sort numerically a field you must be sure that the column type is numeric and not string. Some code would help to elucidate this.

For the second issue also you can't do that directly in the DataView. If you really need to sort the records based on some processing of data in a column then I'd copy the data in an array and use an IComparer on the array:

```
DataView dv = new DataView(dt);
ArrayList lst = new ArrayList();
lst.AddRange(dv.Table.Rows);
lst.Sort(new MyComparer());
foreach (DataRow dr in lst)
    Debug.WriteLine(dr[0]);
```

The comparer is like this:

```
    class MyComparer : IComparer
    {
        public int Compare(object x, object y)
        {
            DataRow rx = x as DataRow;
            DataRow ry = y as DataRow;
            string datax = (string)rx[colName];
            string datay = (string)ry[colName];
            // Process datax and datay here then compare them (ASC)
            return datax.CompareTo(datay);
        }
    }
```

This will increase the memory consumption, so you need to think if there is maybe a better way to preprocess the data in the table so that you can sort directly the DataView by a column.

P.S. colName is the name of the column you're interested to sort by. Replace the comment with actual code to extract the sorting info from the column. You can also use this method to extract sorting info from more columns. Just use something like this:

```
int cmp = colAx.CompareTo(colAy);
if (cmp != 0)
```

```
    return cmp;
cmp = colBy.CompareTo(colBx);
return cmp;
```

This will compare ascending by colA values and then descending by colB values (not that the second compare has *y* first and then *x*)

Edit: OK, I interpreted wrongly the term comma separated values. From your example I think that you actually meant numbers with thousand separators (1,000,000 = one million). If you store numbers like this in the database then it must be that you're using a text field and that should be the reason your sorting order is alphanumeric.

Based on this assumption I would propose to change the type of that column to numeric, keep normal numbers inside and format them (with thousand separators) only when you display them. This way the sort should work directly in the DataView and you don't have to copy the data.

Share

Improve this answer

Follow

edited Sep 19, 2008 at 7:01

answered Sep 18, 2008 at 6:26

rslite

84.5k ● 4 ● 47 ● 47

1

It's ugly, but:

```
        DataView dv = GetDataViewSomewhere();

        //Naturally sort by COLUMN_TO_SORT_ON
        try
        {
            List<string> rowList = new List<string>();
            foreach (DataRowView drv in dv)
                rowList.Add((string)drv["COLUMN_TO_SORT_ON"]);
            rowList.Sort(new NaturalComparer());
            Dictionary<string, int> sortValueHash = new Dictionary<string, int>
();
            for (int i = 0; i < rowList.Count; i++)
                sortValueHash.Add(rowList[i], i);

            dv.Table.Columns.Add("NATURAL_SORT_ORDER", typeof(int));
            foreach (DataRowView drv in dv)
                drv["NATURAL_SORT_ORDER"] =
sortValueHash[(string)drv["COLUMN_TO_SORT_ON"]];
            dv.Sort = "NATURAL_SORT_ORDER";
        }
        catch (Exception)
        {
            DEBUG_TRACE("Could not naturally sort");
            dv.Sort = "COLUMN_TO_SORT_ON";
        }
```

Where `NaturalComparer` is this class.

Share

Improve this answer

Follow

answered Oct 28, 2009 at 23:41

community wiki
srmark

can't really copy and paste this into my app – aron Apr 8, 2010 at 19:37

1 I don't see what more you might want. – srmark Apr 10, 2010 at 22:56

@aron - Grab the class from the provided link and save it in your app. This code works if you have a datatable already. Simply change DataView dv = GetDataViewSomewhere(); to DataView dv = DatatableIalreadyHave.DefaultView; Then run the code (after changing the value of COLUMN_TO_SORT_ON to the column you need to sort on). Then at the end you need to send the data back in to your datatable with DatatableIalreadyHave = dv.ToTable(); – Dave Lucre Jul 3, 2013 at 5:43