

## 17.7.4 GridView – Sortierung der Daten

Die Komponente GridView verfügt über keine *eigene* Methode, um die Daten im Gitter spaltenweise zu sortieren, obgleich es für die GridView die Eigenschaft *Sorted* gibt. Die fehlende Sortier-Methode müssen Sie selbst programmieren, wie ein Blick in die Hilfe zeigt: *Sorting the data is not done automatically. It must be done by user code.* Im Kapitel 7.4 Array finden Sie die grundlegenden Prozeduren zum Sortieren eines abgeleiteten Arrays. Die Grundidee besteht darin, alle Zeilen der GridView in einem Array zu speichern, das Array nach einer vorgegebenen Spalte zu sortieren und dann mit Hilfe der Prozedur 'ArrayToGrid' die GridView mit den Elementen des sortierten Arrays zu füllen.

Das Hauptprogramm nutzt die beiden vorgestellten Klassen *Variant[].class* und *CSort.class* aus dem Kapitel 7, um die Spalten in der Gitteransicht auf- oder absteigend sortiert anzuzeigen. Die entsprechenden Stellen sind im (vollständigen) Quelltext farbig hervorgehoben:

```
[1] ' Gambas class file
[2]
[3] Private hSort As New CSort
[4] Private iLast As Integer = -1
[5] Private iCount As Integer
[6]
[7] Public Sub Form_Open()
[8]
[9]     GridView1.Header = GridView1.Horizontal
[10]    GridView1.Sorted = True
[11]
[12]    GridView1.Columns.Count = 6
[13]
[14]    GridView1.Columns[0].Width = 90
[15]    GridView1.Columns[0].Title = "Number"
[16]    GridView1.Columns[1].Width = 90
[17]    GridView1.Columns[1].Title = "Boolean"
[18]    GridView1.Columns[2].Width = 80
[19]    GridView1.Columns[2].Title = "String"
[20]    GridView1.Columns[3].Width = 160
[21]    GridView1.Columns[3].Title = "Date1"
[22]    GridView1.Columns[4].Width = 90
[23]    GridView1.Columns[4].Alignment = Align.Center
[24]    GridView1.Columns[4].Title = "Number2"
[25]    GridView1.Columns[5].Title = "Date2"
[26]
[27]    FMain.Center
[28]    FMain.Resizable = False
[29]    iCount = 999
[30]    FMain.Text = "Ausgabe von " & Str((iCount + 1) * Str(GridView1.Columns.Count)) & " Zufallsdaten, \\  
die sortiert werden können!"
[31] End ' Form_Open
[32]
[33] Public Sub btnFillGrid_Click()
[34]     Dim i As Integer
[35]     Dim vMatrix As Variant[]
[36]     Dim aStrs As String[] = ["Merkur", ..., "Mars", "Jupiter", "Saturn", "Uranus", "Neptun", "Pluto"]
[37]
[38]     hSort.Clear() ' Spaltensortierung zurücksetzen
[39]     iLast = -1
[40]
[41]     GridView1.Clear
[42]     Randomize
[43]
[44]     For i = 0 To iCount Step 1
[45]         vMatrix = New Variant[]
[46]         vMatrix.Add(Rnd(0, 10), -2) ' Reelle Zahl
[47]         vMatrix.Add(CBool(Rnd(0, 1))) ' Wahrheitswert
[48]         vMatrix.Add(aStrs[CInt(Rnd(0, aStrs.Count))]) ' Zeichenkette
[49]         vMatrix.Add(CDate(Rnd(CFloat(Now()), CFloat(Now() + 1000)))) ' Datum 1
[50]         vMatrix.Add(CInt(Rnd(-10, 10))) ' Ganze Zahl
[51]         vMatrix.Add(CDate(Rnd(CFloat(Now()), CFloat(Now() + 2000)))) ' Datum 2
[52]         hSort.Add(vMatrix)
[53]     Next ' i
[54]
[55]     ArrayToGrid()
[56]
[57] End ' FillGrid
[58]
[59] Public Sub ArrayToGrid()
[60]     Dim i As Integer
[61]     Dim vMatrix As Variant[]
[62]
[63]     GridView1.Rows.Count = hSort.Count
```

## 17.7.4 - GridView – Sortierung der Daten

```

[64] For i = 0 To hSort.Count - 1 Step 1
[65]     vMatrix = hSort[i]
[66]     GridView1[i, 0].Text = Str$(vMatrix[0])
[67]     GridView1[i, 1].Text = Str$(vMatrix[1])
[68]     GridView1[i, 2].Text = Str$(vMatrix[2])
[69]     GridView1[i, 3].Text = Str$(vMatrix[3])
[70]     GridView1[i, 4].Text = Str$(vMatrix[4])
[71]     GridView1[i, 5].Text = Str$(vMatrix[5])
[72] Next i
[73]
[74] End ' ArrayToGrid
[75]
[76] Public Sub GridView1_ColumnClick(Column As Integer)
[77] ' Beim wiederholten Klick auf die gleiche Spalte wird die Sortierung invertiert
[78] If iLast = Column Then
[79]     hSort.Reverse() ' Optimierung, um nicht mit gb.Descent wiederholt zu sortieren
[80] Else
[81]     hSort.SortField(Column)
[82] Endif ' iLast
[83]
[84] ArrayToGrid()
[85] iLast = Column
[86] End ' GridView1_ColumnClick(..)

```

- Die markierte Zeile 3 erstellt ein neues Objekt *hSort* der *CSort*-Klasse, um alle *Variant[]* zu erfassen, die auch im *GridView* dargestellt werden.
- Die *GridView* bekommt in den Zeilen 9 und 10 eine obere, horizontale Kopfzeile und den Status zugewiesen, nach dem sortiert wird. Es wird auch die Anzahl der zu erzeugenden Zufallsdatensätze (*iCount*) auf 999 gesetzt, so dass 1000 Datensätze erzeugt werden. Bei einem Klick auf den Button *bntFillGrid* zur Erzeugung der Zufalls-Datensätze wird zuerst *hSort* geleert, um Platz für die neuen Zufallsdaten zu schaffen. Diese werden – vergleichen Sie mit den Zeilen 44 bis 53 – in ein neues *Variant[]* geschrieben und dieses *Variant[]* wird in *hSort* eingetragen. Die abschließend aufgerufene Routine *ArrayToGrid()* in der Zeile 55 liest jedes einzelne *Variant[]* aus *hSort* ein und stellt es als *GridView*-Zeile dar.
- Die Sortierung findet bei einem Klick auf den horizontalen Kopf der *GridView* statt. Hier wird *hSort.SortField(Column)* aufgerufen. Der Parameter *Column* entspricht der Spaltennummer, die der Benutzer angeklickt hat. Nach erfolgter Sortierung wird *hSort* wieder über *ArrayToGrid()* graphisch dargestellt.

Number	Boolean	String	Date1	Number2	Date2
0,01	True	Merkur	03.07.2013 20:50:37	4	13.05.2014 05:06:20
0,03	False	Merkur	29.09.2014 10:11:48	-2	25.04.2016 20:58:41
0,08	True	Erde	28.02.2014 06:42:51	-3	09.05.2013 18:12:28
0,08	False	Neptun	11.09.2013 16:52:19	-2	04.10.2017 21:14:44
0,08	True	Merkur	21.04.2014 17:12:41	0	07.08.2016 11:37:44
0,1	False	Saturn	09.10.2012 21:10:29	-2	05.09.2016 07:11:32
0,1	False	Saturn	15.10.2012 07:29:03	1	24.06.2015 22:32:37
0,1	True	Pluto	13.10.2012 15:24:10	5	04.09.2016 19:50:48
0,12	False	Saturn	11.09.2012 07:34:13	-4	23.05.2014 11:31:43
0,13	True	Uranus	30.12.2014 13:20:49	1	09.10.2017 08:18:16

Abbildung 17.7.4.1: Sortierung nach der 1. Spalte in der GridView – aufsteigend

Number	Boolean	String	Date1	Number2	Date2
9,99	False	Pluto	29.08.2014 19:51:28	-3	08.08.2014 00:06:21
9,97	True	Venus	24.09.2013 08:08:56	1	27.01.2014 09:48:01
9,97	False	Pluto	25.08.2014 09:39:05	0	01.11.2015 15:35:44
9,94	True	Mars	11.11.2013 03:52:08	-1	01.08.2016 23:00:51
9,93	False	Merkur	26.04.2015 07:51:52	-6	16.06.2014 05:17:41
9,93	True	Merkur	14.06.2013 08:28:03	-7	05.11.2015 23:31:12
9,92	True	Venus	11.04.2013 02:01:32	3	11.12.2012 20:47:21
9,9	False	Erde	20.12.2012 04:46:23	5	08.10.2012 15:39:49
9,9	True	Mars	30.09.2013 23:57:00	-4	11.06.2013 07:03:11
9,89	True	Saturn	05.10.2013 01:37:22	5	01.01.2017 04:59:10

Abbildung 17.7.4.2: Sortierung nach der 1. Spalte in der GridView – absteigend

- Als Optimierung wird in den Zeilen 78 - 82 der Fall angenommen, dass der Benutzer bei einem zweiten Klick auf die gleiche Spalte den Sortier-Modus wechseln möchte. Statt nun ein weiteres Mal mit einem anderen Modus zu sortieren, werden die Elemente im Array `hSort` über die Methode `hSort.Reverse()` in der Reihenfolge invertiert.
- Für den Fall, dass die aktuellen Daten durch neue Zufallsdaten ausgetauscht wurden, werden die Steuerdaten für die Optimierung in Zeile 38 zurückgesetzt, weil jetzt eine Neusortierung zwingend erforderlich ist.