7.4.9.2 Sortierung eines mehrdimensionalen nativen Arrays

Die Array-Sort()-Methode lässt sich auch bei mehrdimensionalen Arrays anwenden – doch sind die Ergebnisse nicht so, wie Sie es vielleicht erwarten würden!

In einer GridView werden in die einzelnen Zeilen jeweils ein Name (1. Feld) und das Alter eingetragen. Anschließend sollen alle Zeilen in der GridView nach den Namen sortiert angezeigt werden. Gut zu wissen, dass eine GridView über die Eigenschaft *GridView.Sorted* verfügt. In der GridView-Dokumentation steht aber, dass man die erforderlichen Sortier-Algorithmen selbst entwickeln muss. Es keimt die Idee, alle Felder der GridView in einem String-Array zu speichern, das Array zu sortieren und das sortierte Array wieder in die GridView einzulesen. Gedacht – getan. In die Elemente eines statischen, zweidimensionalen String-Arrays vom Typ String[5,2] werden beispielsweise diese GridView-Daten als *Zeichenkette* eingetragen, da alle Felder in einer GridView vom Daten-Typ 'String' sind:

```
[ 0 | 0 ] = Fuchs
[ 1 | 0 ] = Dachs
[ 2 | 0 ] = Bär
[ 3 | 0 ] = Uhu
[ 4 | 0 ] = Adler
[ 0 | 1 ] = 16
[ 1 | 1 ] = 15
[ 2 | 1 ] = 16
[ 3 | 1 ] = 18
[ 4 | 1 ] = 15
```

Die formatierte Darstellung der Elemente des Arrays in einer GridView (→ Prozedur *ArrayToGrid(..)*) sieht dann so aus:

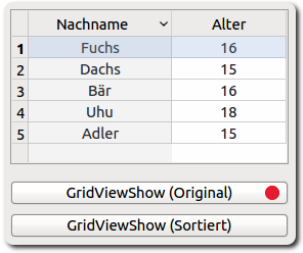

Abbildung 7.4.9.2.1: ArrayToGridView

Der Quelltext wird vollständig angegeben:

```
[1]  Private $2DGridArray As String[]
[2]
[3]  Public Sub Form_Open()
[4]    FMain.Center
[5]    FMain.Resizable = False
[6]    SetGridProperty()
[7]    FillGridView()
[8]    $2DGridArray = GridToArray()
[9]  End ' Form_Open()
[10]
[11] Private Sub SetGridProperty()
[12]   GridView1.Header = GridView1.Both ' Kopfzeile wird angezeigt
[13]   GridView1.Mode = Select.Single ' Die selektierte Zeile wird farbig markiert
[14]   GridView1.Columns.Count = 2 ' Anzahl der Spalten
[15]   GridView1.Sorted = True
[16]   GridView1.Columns[0].Width = 140
[17]   GridView1.Columns[0].Alignment = Align.Center
[18]   GridView1.Columns[0].Title = "Nachname"
[19]   GridView1.Columns[1].Width = 60
[20]   GridView1.Columns[1].Alignment = Align.Center
[21]   GridView1.Columns[1].Title = "Alter"
[22] End ' SetGridProperty()
[23]
[24] Public Sub FillGridView()
[25]   Dim iCount As Integer
[26]   Dim aNames As String[] = ["Adler", "Bär", "Dachs", "Fuchs", "Meise", "Uhu", "Elch"] '
[27]
[28]   Randomize
[29]   For iCount = 0 To 4 ' Datensätze mit Zufallsdaten
[30]     Inc GridView1.Rows.Count ' Anzeige der generierten Datensätze (Zeilen in der Gitteransicht)
[31]     GridView1.MoveTo(GridView1.Rows.Count - 1, 0) ' Optional
[32]     GridView1[iCount, 0].Text = aNames[Int(Rnd(0, aNames.Count))]
```

```
[33]     GridView1[iCount, 1].Text = Str(Int(Rnd(15, 20))) ' 15..19
[34]   Next ' iCount'
[35]
[36] End ' FillGridView()
[37]
[38] Public Function GridToArray() As String[]
[39]   Dim i, j As Integer
[40]   Dim a2DGridArray As New String[5, 2] ' 5 Zeilen, 2 Spalten
[41]
[42]   For i = 0 To GridView1.Rows.Max
[43]     For j = 0 To GridView1.Columns.Max
[44]       a2DGridArray[i, j] = GridView1[i, j].Text
[45]       ' Print i, j, a2DGridArray[i, j]
[46]     Next ' j
[47]   Next ' i
[48]
[49]   Return a2DGridArray
[50]
[51] End ' Function GridToArray()
[52]
[53] Public Sub ArrayToGrid(aArray As String[])
[54]   Dim i, j As Integer
[55]
[56]   GridView1.Clear
[57]   GridView1.Rows.Count = aArray.Bounds[0]
[58]   For i = 0 To aArray.Bounds[0] - 1
[59]     For j = 0 To aArray.Bounds[1] - 1
[60]         GridView1[i, j].Text = aArray[i, j]
[61]     Next ' j
[62]   Next ' i
[63]
[64] End ' ArrayToGrid(..)
[65]
[66] Public Sub GridViewShow(Optional bSorted As Boolean)
[67]   Dim aGArray As New String[]
[68]
[69]   aGArray = $2DGridArray.Copy()
[70]   If bSorted = True Then
[71]      aGArray.Sort
[72]      ArrayToGrid(aGArray)
[73]   Else
[74]     ArrayToGrid($2DGridArray)
[75]   Endif
[76]
[77] End ' GridViewShow(..)
[78]
[79] Public Sub btnGridViewShow_Click()
[80]   GridViewShow()
[81] End ' btnGridViewShow_Click()
[82]
[83] Public Sub btnGridViewShowS_Click()
[84]   GridViewShow(True)
[85] End ' btnGridViewShowS_Click()
```

Kommentare:

- Die Funktion in den Zeilen 38 bis 51 liefert als Funktionswert ein Array mit den GridView-Daten.
- In den Zeilen 53 bis 64 werden die Elemente des erzeugten Arrays a2DGridArray in der Prozedur ArrayToGrid(..) in die GridView eingetragen (→ Abbildung 7.4.4.2.1), der als Parameter das Arrays *a2DGridArray* übergeben wurde.
- Die Prozeduren in den Zeilen 66 bis 86 realisieren das Anzeigen des originalen und des sortierten Arrays in der GridView.

Das Ergebnis ist ernüchternd – wie die Anzeige der Elemente in der GridView und in der IDE-Konsole verdeutlicht – obwohl die Elemente des zweidimensionalen Arrays *korrekt* sortiert sind:

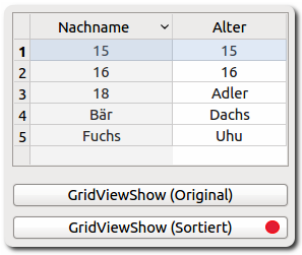

Abbildung 7.4.9.2.2: Anzeige der Elemente des sortierten Arrays in der GridView

```
[ 0 | 0 ] = 15
[ 1 | 0 ] = 15
[ 2 | 0 ] = 16
[ 3 | 0 ] = 16
[ 4 | 0 ] = 18
[ 0 | 1 ] = Adler
[ 1 | 1 ] = Bär
[ 2 | 1 ] = Fuchs
[ 3 | 1 ] = Dachs
[ 4 | 1 ] = Uhu
```

Eine akzeptable Lösung wird Ihnen im → Kapitel 7.4.9.3 Sortierung abgeleiteter Arrays vorgestellt.