17.5.1 Projekt 1

Das Projekt demonstriert neben der Verwendung von Eigenschaften des Steuerelements ColumnView die Sortierung der Daten und besonders den Einsatz der diversen Move()-Methoden.

Abbildung 17.5.1.1: Programmfenster – Sortierung nach Nachnamen eingeschaltet

- Sie können jederzeit neue (Zufalls)Daten erzeugen (12 Elemente). Es ist von Vorteil, wenn Sie ein Element als Zeile in einer ColumnView auffassen.

```
Public Sub btnNewData_Click()

  covData.Clear() ' Inhalt der ColumnView löschen. Gilt nicht für die optionale Kopf-Zeile!
  SetData(12) ' 12 Elemente mit Zufallsdaten neu einfügen

' Markierung des ersten Elementes - wenn mindestens eins existiert
  If Not covData.MoveFirst() Then covData.Key = covData.Item.Key
  txbCurrentKey.Text = covData.Item.Key

End
```

```
Private Sub SetData(iCount As Integer)

  Dim i As Integer

  Dim aFirstNames As String[] = ["Hans", "Maria", ..., "Robert", "Stefan", "Emma", "Yvonne", "Claus"]
  Dim aSurNames As String[] = ["Meyer", "Lehmann", ..., "Müller", "Grahn", "Kaiser", "Vogt", "Zechlin"]
  Dim aPictures As String[] = ["led_green16.png", "led_red16.png", "led_blue16.png"]

  For i = 1 To iCount
    covData.Add(Str(i), Str(i), Picture[aPictures[Rand(0, aPictures.Max)]])            ' 1. Spalte + Bild
    covData[Str(i)][1] = aFirstNames[Rand(0, aFirstNames.Max)]                         ' 2. Spalte
    covData[Str(i)][2] = aSurNames[Rand(0, aSurNames.Max)]                             ' 3. Spalte
    covData[Str(i)][3] = Format$(Date(CFloat(Now() - Rnd(7000, 8200))), "dd.mm.yyyy")  ' 4. Spalte
  Next

End
```

- Die Daten können auch sortiert ausgegeben werden (→ CheckBox), wobei die Daten zuerst nach der 3. Spalte 'Nachname' sortiert werden. Anschließend können alle weiteren Spalten ebenso sortiert werden.

```
Public Sub ckboxSorting_MouseDown()

  txbCurrentKey.Text = covData.Item.Key

  If ckboxSorting.Value = True Then
     covData.Sorted = False
  Else
     covData.Sorted = True            ' Es soll sortiert werden!
     covData.Columns.Ascending = True ' A → Z
     covData.Columns.Sort = 2         ' Sortieren nach Nachname
  Endif

End
```

- Den internen Zeiger können Sie *sofort* zum ersten oder zum letzten Element bewegen, sofern es mindestens ein Element gibt – was geprüft wird. Damit Sie die aktuelle End-Position sehen können, werden das erste oder das letzte Element anschließend markiert:

```
Public Sub btnFirst_Click()

  If covData.Count > 0 Then ' Es existiert mindestens ein Element ...
     covData.MoveFirst()
     txbCurrentKey.Text = covData.Item.Key
    ' Element markieren (ColumnView ohne Focus: Markierung grau)
     covData.Key = covData.Item.Key
  Endif

End
```

```
Public Sub btnLast_Click()

  If covData.Count > 0 Then ' Es existiert mindestens ein Element ...
     covData.MoveLast()
     txbCurrentKey.Text = covData.Item.Key
    ' Element markieren (ColumnView ohne Focus: Markierung grau)
     covData.Key = covData.Item.Key
  Endif

End
```

- Von jeder beliebigen Position aus können Sie den internen Zeiger *schrittweise* zum ersten oder zum letzten Element bewegen:

```
Public Sub btnUp_Click()

  If covData.Count > 0 Then ' Existiert mindestens ein Element ...
     If Not covData.MoveAbove() Then
        txbCurrentKey.Text = covData.Item.Key & " : " & covData.Item[1]
        covData.Key = covData.Item.Key
     Else
        covData.MoveFirst()
     Endif
  Endif

End
```

```
Public Sub btnDown_Click()

  If covData.Count > 0 Then ' Existiert mindestens ein Element ...
     If Not covData.MoveBelow() Then
        txbCurrentKey.Text = covData.Item.Key & " : " & covData.Item[1]
        covData.Key = covData.Item.Key
     Else
        covData.MoveLast()
     Endif
  Endif

End
```

- Von jeder beliebigen Position aus können Sie den internen Zeiger vom ersten bis zum letzten Element bewegen (Iteration). Diese Prozedur ist gut geeignet, um Daten aus einer ColumnView auszulesen oder in eine ColumnView einzulesen. Da die Markierung ausgeschaltet ist, um die Unabhängigkeit des internen Zeigers vom äußeren (Markierungs-)Zeiger zu demonstrieren, erkennen Sie die aktuelle Position bei der Iteration in der Textbox (Key und Text der 1. Spalte):

```
Public Sub btnIterationDown_Click()

  Dim iLastKey As Integer

  If covData.Count > 0 Then ' Es existiert mindestens ein Element ...
     iLastKey = covData.Key ' Start-Key speichern für ***
  Endif

' Den internen Zeiger auf das oberste Element setzen, wenn das möglich ist
  If Not covData.MoveFirst() Then ' Wenn mindestens ein Element existiert ...
     Repeat ' ... dann Iteration vom obersten Element
       txbCurrentKey.Text = covData.Item.Key & " : " & covData.Item[1]
     ' covData.Key = covData.Item.Key ' Element markieren abgeschaltet
       Wait 0.4
     ' Hier weitere Anweisungen zur Verarbeitung von Daten aus der ColumnView: Zum Beispiel
     ' Print covData.Item[0]; gb.Tab; covData.Item[1]; gb.Tab; covData.Item[2]; gb.Tab; covData.Item[3]
     Until covData.MoveBelow() ' ... bis das letzte Element in der ColumnView erreicht ist
     covData.MoveTo(iLastKey) ' ***
  Endif

End
```

Um den Quelltext vollständig anzugeben, wird u.a. gezeigt, wie das Layout der ColumnView festgelegt wird und Startwerte gesetzt werden:

```
Public Sub Form_Open()

  FMain.Resizable = False

  With covData
   ' Anzahl der Spalten festlegen
    .Columns.Count = 4

    .Header = True ' Kopfzeile anzeigen
   ' Header-Spalten-Namen festlegen
    .Columns[0].Text = ("RGB")
    .Columns[1].Text = ("Vorname")
    .Columns[2].Text = ("Nachname")
    .Columns[3].Text = "Geburtsdatum"
   ' Spalten-Weite festlegen
    .Columns[0].Width = 25
    .Columns[1].Width = 130
    .Columns[2].Width = 130
    .Columns[3].Width = 150
   ' Spalten-Ausrichtung festlegen
    .Columns[0].Alignment = Align.Center
    .Columns[1].Alignment = Align.Normal
    .Columns[2].Alignment = Align.Normal
    .Columns[3].Alignment = Align.Center

    .Mode = Select.Single
    .Sorted = False ' Es soll *nicht* sortiert werden!
    .AutoResize = True
    .Resizable = True
  End With

  ckboxSorting.Value = ckboxSorting.False
  SetData(12)

' Markierung des 7. Elementes
' If covData.Count > 0 Then ' Es existiert mindestens ein (oberes) Element ...
  If Not covData.MoveFirst() Then ' Es existiert mindestens ein (oberes) Element ...
     If covData.MoveTo(7) = True Then ' Wenn ein Fehler auftrat ...
        covData.MoveBack() ' ... dann zurück zum Ausgangselement
        covData.Key = covData.Item.Key ' Element markieren
     Else
      ' Element markieren (ColumnView hat Focus → Markierung hellgrün (Mint 17.3))
        covData.Key = covData.Item.Key
     Endif
  Endif

End

Public Sub ckboxSorting_MouseDown()

  txbCurrentKey.Text = covData.Item.Key

  If ckboxSorting.Value = True Then
     covData.Sorted = False
  Else
     covData.Sorted = True            ' Es soll sortiert werden!
     covData.Columns.Ascending = True ' A → Z
     covData.Columns.Sort = 2         ' Sortieren nach Nachname
  Endif

End

Public Sub Form_Close()
  FMain.Close()
End
```