17.7.7 GridView – Daten-Import aus einer csv-Datei

In vielen Programmen besteht die Möglichkeit, Daten in einer csv-Datei zu speichern, um diese dann in einem anderen Programm weiter zu verarbeiten. Sie kennen das sicher aus der Arbeit mit einem Kalkulationsprogramm. Probleme beim Daten-Import aus einer csv-Datei gibt es immer dann, wenn man

- nicht sicher ist, welche Text- und Feldtrennzeichen beim Daten-Export verwendet wurden,
- nicht weiß, ob der erste Datensatz in der csv-Datei Feldnamen enthält oder
- ein Text- und Feldtrennzeichen in einem Feld enthalten ist. Das ist zum Beispiel der Fall, wenn die System-Lokalisation auf deutsch eingestellt ist und reelle Zahlen mit einem Komma als Dezimaltrennzeichen speichert werden.

In der ersten Variante für den Daten-Import aus einer csv-Datei wird von einer (englischen) Lokalisation ausgegangen, bei der u.a. reelle Zahlen mit einem Punkt als Dezimaltrennzeichen speichert werden. Das Projekt besteht aus dem Hauptprogramm von dem ein Formular für die Ermittlung der Text- und Feldtrennzeichen aufgerufen werden kann sowie einem Modul. Das Modul stellt dem Programm über 2 Funktionen eine Datenmatrix zur Verfügung, in der alle Feld-Inhalte aus der csv-Datei gespeichert sind. Ideale Voraussetzungen, die Daten 'en bloc' blitzschnell in die GridView einzufügen und anzuzeigen.

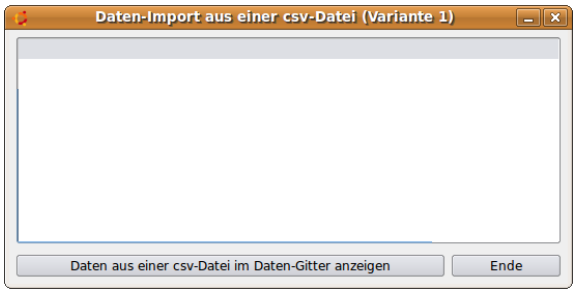

Abbildung 17.7.7.1: Hauptprogramm mit der GridView

Zuerst wählen Sie die csv-Datei (Dateifilter) in einem Dialog:

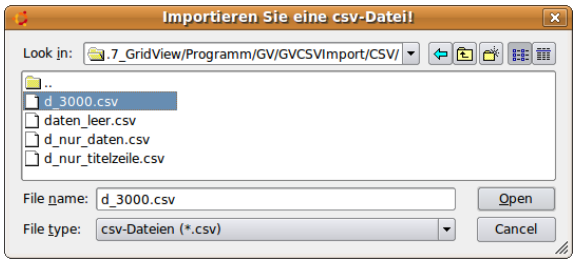

Abbildung 17.7.7.2: FileOpen-Dialog

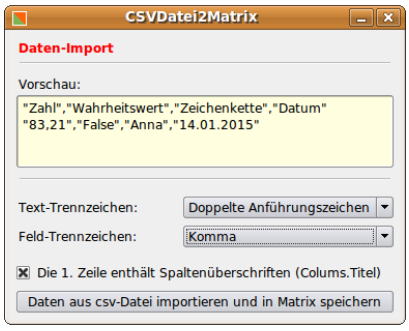

Abbildung 17.7.7.3: Festlegung der Trennzeichen

Im o.a. Formular werden maximal die ersten beiden Zeilen der csv-Datei angezeigt, damit Sie erkennen und auswählen können:

- welche Text- und Feldtrennzeichen in der csv-Datei verwendet wurden und
- ob der erste Datensatz in der csv-Datei Feldnamen enthält.

Abschließend wird die csv-Datei geöffnet und deren Feld-Inhalte sowie u.U. auch die Feldnamen in einer Datenmatrix gespeichert und deren Inhalt sofort in die GridView eingefügt und angezeigt:

Abbildung 17.7.7.4: Anzeige der Felder – hier mit Feldnamen – in der GridView

Interessant ist vor allem der Quelltext des Moduls, der komplett angegeben wird, während vom Hauptprogramm nur 2 Prozeduren kommentiert werden:

```
[1]  PUBLIC hFile AS File
[2]  PUBLIC sFeldTrennzeichen AS String
[3]  PUBLIC sTextTrennzeichen AS String
[4]  PUBLIC bExistTitel AS Boolean
[5]  PUBLIC bFileOpenCancel AS Boolean
[6]  PUBLIC iFieldCount AS Integer
[7]  PUBLIC aPreView AS Variant[]
[8]  PUBLIC aMatrix AS Variant[]
[9]
[10] PUBLIC FUNCTION FileOpen() AS Boolean
[11]   DIM sLine AS String
[12]
[13]   Dialog.Title = "Importieren Sie eine csv-Datei!"
[14]   Dialog.Filter = ["*.csv", "csv-Dateien"]
[15]   IF Dialog.OpenFile(FALSE) = TRUE THEN ' Multiselect=False (Standard)
[16]      Message.Info("Das Öffnen der csv-Datei wurde abgebrochen!")
[17]      RETURN FALSE ' Cancel-Button gedrückt
[18]   ELSE
[19]      hFile = OPEN Dialog.Path FOR INPUT
[20]      IF Lof(hFile) = 0 THEN
[21]         Message.Info("Die csv-Datei ist leer!")
[22]         hFile.Close
[23]         RETURN FALSE
[24]      ELSE
[25]         aPreView = NEW Variant[]
[26]         WHILE NOT Eof(hFile)
[27]           LINE INPUT #hFile, sLine
[28]           aPreView.Add(sLine)
[29]         WEND
[30]      ENDIF ' Lof(File) = 0?
[31]   ENDIF ' Dialog.OpenFile(FALSE) = TRUE?
[32] ' Wenn KEIN Fehler eintrat, dann ...
[33]   RETURN TRUE
[34] END ' FileOpen()
[35]
[36] PUBLIC FUNCTION ImportCSV2Matrix(pTextTrenner AS String, pFeldTrenner AS String) AS Boolean
[37]   DIM sLine AS String
[38]   DIM iCount AS Integer
[39]   DIM aFieldList AS String[] ' Der Typ ist durch den Funktionswert von Split() festgelegt
[40]
[41]   aMatrix = NEW Variant[]
[42]   SEEK #hFile, 0 ' Notwendig, weil das existierende Datei-Handle genutzt wird
[43]   WHILE NOT Eof(hFile)
[44]     LINE INPUT #hFile, sLine
[45]     sLine = Replace(sLine, pTextTrenner, "")
[46]     aFieldList = NEW String[] ' Für jede Zeile wird ein neues Array genutzt
[47]     IF iCount = 0 THEN
[48]        iFieldCount = Split(sLine, pFeldTrenner).Count
[49]        INC iCount
[50]     ENDIF ' iCount = 0
[51]     aFieldList = Split(sLine, pFeldTrenner) ' Lokalisation englisch!
[52]     aMatrix.Add(aFieldList)
[53]   WEND ' NOT Eof(hFile)
[54] ' Wenn KEIN Fehler eintrat, dann ...
[55]   RETURN TRUE
[56] END ' ImportCSV2Matrix
```

- In den Zeilen 25 bis 29 wird eine Matrix aPreView mit den *originalen* Datensätzen aus der csv-Datei gefüllt. Diese Matrix wird im Formular *CSVDatei2Matrix* für die Vorschau auf die ersten beiden Zeilen in der csv-Datei verwendet.
- In der Zeile 48 wird *die Anzahl der Felder* ermittelt und für die Festlegung der Spaltenzahl in der GridView benötigt. Die *Anzahl der Zeilen* in der GridView entspricht der Anzahl der Elemente in der Matrix mit dem Namen aMatrix.
- Das Aufteilen eines Datensatzes in Zeile 51 unter Einsatz des Feldtrennzeichen liefert einzelne Zeichenketten, die als Elemente in eine Matrix aFieldList vom Typ String[] eingefügt werden.
- Wenn in beiden Funktionen keine Fehler auftraten, dann wird die GridView sofort mit den Daten aus aMatrix gefüllt, wie Sie der Abbildung 17.7.7.4 entnehmen können. Zum Einsatz kommt auch hier die das bewährte GridView_Data-Ereignis:

```
PUBLIC SUB GridView1_Data(Row AS Integer, Column AS Integer)
  GridView1.Data.Text = MCSV.aMatrix[Row][Column]
  IF Row MOD 2 = 1 THEN
     GridView1.Data.Background = Color.RGB(224, 224, 224) ' hellgrau
  ENDIF ' MOD 2 = 0?
END ' GridView1_Data(..)
```

Im Hauptprogramm sind es die Funktion *ImportCSVFile()* und die Prozedur *DisplayGridView(), mit denen Sie* den FileOpen-Dialog, das Speichern der importierten Feld-Daten in einer Matrix, das Setzen der GridView-Eigenschaften und das Anzeigen der Feldinhalte in der GridView realisieren:

```
PUBLIC FUNCTION ImportCSVFile() AS Boolean
  IF MCSV.FileOpen() = FALSE THEN
     'Message.Warning("Fehler beim Öffnen der CSV-Datei!")
     RETURN FALSE
  ENDIF ' MCSV.FileOpen() = FALSE?
  FImport.ShowModal()
  IF MCSV.bFileOpenCancel = TRUE THEN
     Message.Warning("Daten-Import abgebrochen!")
     RETURN FALSE
  ENDIF ' MCSV.bFileOpenCancel = TRUE?
  IF MCSV.ImportCSV2Matrix(MCSV.sTextTrennzeichen, MCSV.sFeldTrennzeichen) = FALSE THEN
     Message.Warning("Fehler beim Daten-Import aus der CSV-Datei!")
     RETURN FALSE
  ENDIF ' Fehler beim Auslesen der csv-Datei?
' Wenn KEIN Fehler eintrat, dann ...
  RETURN TRUE
END ' ImportCSVFile()

PUBLIC SUB DisplayGridView()
  IF ImportCSVFile() = TRUE THEN
     SetGridProperty(MCSV.iFieldCount, MCSV.bExistTitel, MCSV.aMatrix[0])
     IF MCSV.bExistTitel = TRUE THEN MCSV.aMatrix.Remove(0)
     GridView1.Rows.Count = MCSV.aMatrix.Count ' Erst jetzt wird das Gitter mit Werten gefüllt!
     GridView1.SetFocus
     GridView1.MoveTo(0, 0)
  ELSE
    Message.Warning("Es ist ein Fehler aufgetreten!")
  ENDIF ' ImportCSVFile() = FALSE?
END ' DisplayGridView
```

Der Datenimport aus einer csv-Datei mit dem u.a. Inhalt funktioniert *ohne* Fehlermeldungen und ergibt die danach eingefügte Anzeige in einer GidView, weil die Zahl ein Feldtrennzeichen enthält:

```
"Zahl","Wahrheitswert","Zeichenkette","Datum"
"83,21","False","Anna","14.01.2015"
"8,015","False","Hans","24.06.2014"
```

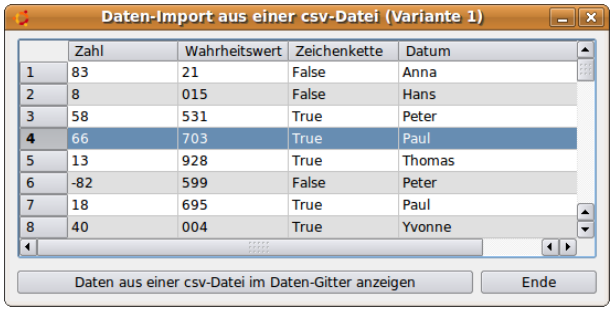

Abbildung 17.7.7.5: Fehlerhafte Anzeige der Felder in der GridView

Auf einem System mit deutscher Lokalisation werden Sie ständig mit diesem Problem zu tun haben. Die Lösungen können vielfältig ausfallen, weil es sehr unterschiedliche Ansätze geben kann. Einer wäre die Verwendung eines anderen Feldtrennzeichens, obgleich der Standard das Komma ist.

In der vorgestellten Lösung kommt ein DEA (Deterministischer Endlicher Automat) im Modul zum Einsatz. Sie brauchen deshalb nur die Funktion ImportcSV2Matrix aus dem vorangegangenen Projekt gegen die folgende tauschen:

```
[1]  PUBLIC FUNCTION ImportCSV2Matrix(pTextTrenner AS String, pFeldTrenner AS String) AS Boolean
[2]    DIM sLine, sZustand, sField, sMitteilung, sLeer AS String
[3]    DIM sEingabeZeichen AS String
[4]    DIM iCount, iZeile, k, i AS Integer
[5]    DIM aFieldList AS Variant[]
[6]    DIM aFieldsList AS Variant[]
[7]
[8]    aFieldList = NEW Variant[]
[9]    aFieldsList = NEW Variant[]
[10]   aMatrix = NEW Variant[]
[11]
[12]   sZustand = "z0"
[13]   sField = ""
[14]   SEEK #hFile, 0
[15]
[16]   WHILE NOT Eof(hFile)
[17]     INC iZeile
[18]     LINE INPUT #hFile, sLine
[19]     sLine = Trim(sLine)
[20]     IF NOT Eof(hFile) THEN
[21]        sLine &= Chr(30) ' Chr(30)=Record Separator
[22]     ELSE
[23]        sLine &= Chr(3) & Chr(30) ' Chr(3)= ETX (End of text)
[24]     ENDIF ' NOT Eof(hFile)?
[25]
[26]   ' Anfang Deterministischer Endlicher Automat
[27]     FOR iCount = 1 TO Len(sLine)
[28]         sEingabeZeichen = Mid(sLine, iCount, 1)
[29]         SELECT sZustand
[30]         '-------------------------------------------------------------------------------
[31]         CASE "z0"
[32]             IF sEingabeZeichen = pTextTrenner THEN
[33]                sZustand = "z1"
[34]             ELSE
[35]                sZustand = "zF"
[36]             ENDIF
[37]         '-------------------------------------------------------------------------------
[38]         CASE "z1"
[39]             IF sEingabeZeichen <> pTextTrenner THEN sField &= sEingabeZeichen
[40]             IF sEingabeZeichen = pTextTrenner AND icount < Len(sLine) THEN
[41]                aFieldList.Add(sField)
[42]                sField = ""
[43]                sZustand = "z2"
[44]             ENDIF
[45]         '-------------------------------------------------------------------------------
[46]         CASE "z2"
[47]             IF sEingabeZeichen = Chr(30) THEN
[48]                sZustand = "z0"
[49]                aFieldList.Add(sField)
[50]                aFieldsList.Add(aFieldList)
[51]                aFieldList = NEW Variant[]
[52]             ENDIF
[53]             IF sEingabeZeichen = pFeldTrenner THEN sZustand = "z0"
[54]             IF sEingabeZeichen = Chr(3) THEN sZustand = "z3"
[55]             IF sEingabeZeichen <> pFeldTrenner AND sEingabeZeichen <> Chr(30) AND /
[56]                sEingabeZeichen <> Chr(3) THEN
[57]                sZustand = "zF"
[58]             ENDIF
[59]         '-------------------------------------------------------------------------------
[60]         CASE "z3"
[61]              IF sEingabeZeichen = Chr(30) THEN
[62]               ' Message.Info("<center><font color='DarkGreen'><b>Der Endzustand ist erreicht!</b> /
[63]                              <hr>Fehler = NULL</font></center>")
[64]                aFieldsList.Add(aFieldList)
[65]                aMatrix = aFieldsList
[66]              ENDIF
[67]         '-------------------------------------------------------------------------------
[68]         CASE "zF"
[69]             sMitteilung = "<font color='Red'><b>Fehler!</b></font><br />"
[70]             sMitteilung &= "<hr>Zeile = " & Str(iZeile) & "<br />"
[71]             sMitteilung &= "Zeichen = " & Str(iCount - 1)
[72]             Message.Error(sMitteilung)
[73]             hFile.Close
```

```
[74]             RETURN FALSE
[75]         END SELECT
[76]     NEXT ' iCount
[77] ' Ende Deterministischer Endlicher Automat
[78]   WEND ' NOT Eof(hFile)
[79]
[80]   hFile.Close
[81]   iColumnCount = aFieldList.Count
[82] ' Wenn KEIN Fehler eintrat, dann ...
[83]   RETURN TRUE
[84] END ' ImportCSV2GridView(...)
```

- In den Zeilen 20 bis 24 werden das Ende jedes Datensatzes und das Ende des letzten Datensatzes in besonderer Weise markiert.
- Der DEA kommt in den Zeilen 26 bis 77 zum Zuge. Sie sehen, wie in Abhängigkeit vom Eingabezeichen und dem aktuellen Zustand ein neuer Zustand eingenommen wird und u.U. auch Etwas ausgegeben wird.
- Der Fehlerzustand wird eingenommen, wenn eine Zeile aus der CSV-Datei syntaktisch nicht in Ordnung ist. Es wird der Funktionswert FALSE zurückgegeben und der Import abgebrochen.

Abbildung 17.7.7.6: Korrekte Anzeige der Felder in der GridView (Einsatz DEA)

Auch in einem anderen Fall können Sie in Bedrängnis geraten – nämlich genau dann, wenn Sie eine csv-Datei verwenden, in der u.a. der Name des Sängers *Mike O'Bryan* vorkommt und jemand voller Begeisterung für das csv-Dateiformat das einfache Anführungszeichen ' als Texttrennzeichen nutzt.

Lassen Sie sich durch Nichts und Niemanden aufhalten, um für diesen Fall eine eigene Lösung zu finden!