Verzeichnisstruktur im EXCEL-VBA bearbeiten

VBA-Blätter als Textdateien speichern

Karel Štípek DSK-537\\ALLTXT.XLS

Verwendung

Wenn man eine Programmiersprache lernt, ist es sinnvoll, aus fertigen Beispielprogrammen praktische Tips zu holen. Einige werden von Microsoft geliefert, viele weitere kann man sich von CompuServe oder Internet laden.

Damit sich man den Code anschauen kann, muß man jede einzelne XLS-Datei zuerst mit EXCEL öffnen. Wenn man eine größere Menge von Beispieldateien (eventuell auch in verschieden Verzeichnissen) hat, ist eine andere Lösung effizienter: Alle VBA-Blätter können als Textdateien gespeichert werden und in diesen Textdateien kann man dann mit z.B. FileFind aus Norton Utilities nach Schlüsselwörtern suchen.

Das vorgestellte Programm bearbeitet das angegebene Verzeichnis mit allen Unterverzeichnissen, öffnet nacheinander alle EXCEL Mappen und speichert alle VBA-Module in gleichnamige Textdateien. Es ist ein Beispiel, wie man die gegliederte Verzeichnisstruktur in einem rekursiven Vorgang behandeln kann. (Zum Thema Rekursion können Sie auf der Begleitdiskette zum Heft Nr.49 mehr Erklärung finden).

Beschreibung

Die Hauptprozedur

Der Name eines EXCEL-Blattes kann länger als 8 Zeichen sein und darf außerdem auch Leerzeichen enthalten, entspricht also nicht den DOS-Konventionen. Dieses Programm ist mit EXCEL Version 7 unter Windows 95 entwickelt worden, so konnte das Problem mit den langen Dateinamen umgangen werden. Sollte aber das Programm unter Windows 3.x lauffähig sein, müßte man aus dem Blattnamen einen gültigen DOS-Namen bilden (z.B. die ersten 8 nichtleeren Zeichen nehmen und doppelte Namen eventuell vermeiden).

Die vorgelegte Version enthält diese Namenskonversion nicht, das Programm muß also mit der Überprüfung der EXCEL-Version beginnen bei älteren EXCEL-Versionen beendet werden.

```
Sub main()
Dim startdir$
'funktioniert nur unter WINDOWS 95, weil mit langen Dateinamen arbeitet
If Application.Version < "7" Then
  MsgBox "Nur mit EXCEL Version 7 lauffähig"
  Exit Sub
End If
```

Das Startverzeichnis wird in einem standardmäßigen EXCEL-Dialog abgefragt und eventuell ein Backslash angehängt

```
startdir=Application.InputBox(Title:="S T A R T V E R Z E I C H N I S", _
    default:="C:\MSOFFICE\EXCEL", _
    prompt:="In diesem Verzeichnis und in allen Unterverzeichnissen " & _
    "wird in allen Mappen aus jedem VBA-Modul eine Textdatei erstellt")
If startdir = False Then ' Abbrechen
  Exit Sub
End If
If Right(startdir, 1) <> "\" Then startdir = startdir & "\"
```

Die Hauptprozedur endet mit dem Aufruf der Prozedur SCANDIR. Dieser Aufruf bearbeitet die Dateien und Unterverzeichnisse im Startverzeichnis.

```
scandir (startdir)
End Sub
'end of main
```

Die Prozedur SCANDIR

Die Prozedur SCANDIR bearbeitet die Einträge in einem Verzeichnis. Dieser Vorgang ist laut Microsoft folgendermaßen zu programmieren:

Die Funktion DIR$() wird mit dem gewünschten Namen (* oder ? möglich) und Dateityp als Parameter aufgerufen und liefert den Namen der gefundenen Datei (oder einen Leerstring, wenn keine gefunden wird). Der nächste Aufruf DIR$() ohne Parameter liefert dann den nächsten Dateinamen, der die vorher definierte Auswahlbedingung erfüllt.

Wenn der Eintrag im Verzeichnis ein Unterverzeichnis ist, muß man die gleiche Funktion für dieses Unterverzeichnis aufrufen und so geht es rekursiv weiter, bis die niedrigste Ebene erreicht ist. Ich habe leider festgestellt, daß die Funktion DIR$() bei rekursiven Aufruf nicht richtig funktioniert. Bei der Rückkehr aus dem Unterverzeichnis in die nächst höhere Ebene war die Auswahlbedingung nicht mehr definiert. Deswegen habe ich alle interessanten Verzeichniseinträge zuerst im Array FILEARR gespeichert und dieses Array statt des Verzeichnisses bearbeitet.

```
Sub scandir(actdir$)
Dim actfile$, i%
Dim filearr()
i = 0
ReDim filearr(0)
If Right(actdir, 1) <> "\" Then actdir = actdir & "\"

'alle Dateien inklusive Unterverzeichnisse suchen
actfile = Dir$(actdir & "*.*", vbDirectory)

While Len(actfile) > 0
  'speichere nur XLS und Unterverzeichnisse
  If (Left$(actfile, 1) <> ".") Then
    If (Right$(actfile, 3) = "XLS")
    Or (GetAttr(actdir & actfile) = vbDirectory) Then
      i = i + 1
      ReDim Preserve filearr(i)
      filearr(i) = actfile
    End If
  End If
  actfile = Dir$()
Wend
```

Das Array FILEARR enthält jetzt alle Dateien mit der Erweiterung XLS und alle Unterverzeichnisnamen des gerade bearbeiteten Verzeichnisses ACTDIR. In der nächsten Schleife wird das Array bearbeitet.

Wenn das Element ein Verzeichnis darstellt, wird die Funktion SCANDIR dafür aufgerufen, sonst handelt es sich um eine EXCEL-Mappe und die wird mit der Funktion SAVETXT behandelt. Dabei wird aber die Mappe mit diesem Programm und die persönliche Arbeitsmappe ausgeschlossen, weil die bereits offen sind.

```
For i = 1 To UBound(filearr)
  actfile = filearr(i)
  If GetAttr(actdir & actfile) = vbDirectory Then
    scandir (actdir & actfile)
  Else
    If actfile <> ThisWorkbook.Name And actfile <> "PERSONL.XLS" Then
      savetxt actdir, actfile
    End If
  End If
Next
End Sub
'end of scandir
```

Die Prozedur ALLTXT

Die Prozedur ALLTXT bearbeitet eine EXCEL-Mappe.

Die Mappe wird geöffnet und alle VBA-Module in gleichnamige Textdateien gespeichert. Wenn die Textdatei schon existiert, kann sie überschrieben, übersprungen oder das Programm beendet werden. Am Ende der Prozedur wird die Mappe wieder geschlossen.

```
Sub savetxt(actdir$, actfile$)
'Alle Modulle der Mappe werden in .TXT gespeichert
Dim w As Workbook
Dim m As Module
Dim i%, txtname$, answ%

Workbooks.Open filename:=actdir & actfile, ReadOnly:=True
Set w = ActiveWorkbook
For Each m In Modules
  txtname = m.Name & ".TXT"
  If Len(Dir$(actdir & txtname)) > 0 Then 'Textdatei existiert schon
    answ=MsgBox("Datei " & actdir & txtname & " überschreiben ?",
                vbYesNoCancel + vbQuestion + vbDefaultButton2, "ABFRAGE")
    Select Case answ
      Case vbYes
        Kill actdir & txtname
        m.SaveAs filename:=actdir & txtname, FileFormat:=xlText,
                 CreateBackup:=False
      Case vbCancel
        Application.Quit
    End Select
  Else
    m.SaveAs filename:=actdir & txtname, FileFormat:=xlText,
             CreateBackup:=False
  End If
Next
w.Saved = True
w.Close
End Sub
'end of savetxt
```