Alle Module eines WORD-Projekts exportieren und in ein WORD-Dokument importieren

Dr. Volker Thormählen, 02. April 2020

Inhalt

1 Aufgabenstellung 2
1.1 Ausgangssituation 2
1.2 Prozesskette 2
2 VBA-Makros 4
2.1 ExportModules 4
2.2 FileProcessing 5
3 Zusammenfassung 7

Abbildungen

Abbildung 1: Liste der VBA-Module eines Word-Projekts 3
Abbildung 2: Erfolgsmeldung der Prozedur 'Export_VBA_Modules' 5
Abbildung 3: Erfolgsmeldung der Prozedur 'TextfileProcessing' 7

Listings

Listing 1: Treiber-Prozedur 2
Listing 2: Alle Module jeweils als Textdatei exportieren 4
Listing 3: Exportierte Textdateien auswählen und in des Zieldokument importieren 6
Listing 4: Die Existenz einer Textdatei bestimmen 7

1 Aufgabenstellung

Wie kann der Quellcode aller Modulen eines umfangreichen Word-Projekts jeweils in ein Textdokument *exportiert* und dieser anschließend in Word-Dokument *importiert* werden, wenn diese beiden Vorgänge durch VBA-Prozeduren so weit wie möglich unterstützt werden sollen?

1.1 Ausgangssituation

Angenommen, der Quellcode aller VBA-Module[1] (s. Abbildung 1) eines Word-Projekts soll automatisch in ein vorgegebenes Word-Zieldokument eingefügt werden.

1.2 Prozesskette

Die Prozesskette für diese definierte Aufgabenstellung lässt sich wie folgt spezifizieren (s. Listing 1):

```
Sub App_Driver()
' 1. Vorgang: Alle Module jeweils als Textdatei exportieren.
Application.Run "ModulesProc.Export_VBA_Modules"
' 2. Vorgang: Exportierte Textdateien jeweils in ein bestimmtes Word-Dokument importieren.
Application.Run "TextfileProcessing"
End Sub
```

Listing 1: Treiber-Prozedur

Die ***Application.Run***-Methode führt jeweils ein bestimmtes VBA-Makro[2] aus (s. Listing 1):

- Der erste Vorgang benötigt das Makro **Export_VBA_Modules**
- Der zweite Vorgang benötigt das Makro **TextfileProcessing**

Diese beiden Makros werden im Folgenden ausführlich beschrieben. Zuvor werden die Namen der zu exportierenden VBA-Module präsentiert (s. Abbildung 1). Anhand der Namen der Module ist unschwer zu erkennen, dass sie zur Bereinigung von Word-Dokumenten dienen. Ihre Aufgaben werden hier allerdings *nicht* näher beleuchtet.

[1] Das Kürzel VBA steht für *Visual Basic for Applications*, einer einfachen und leistungsstarken Scriptsprache für die sog. Office-Anwendungen der Firma *Microsoft*.

[2] Makros (Prozeduren oder Funktionen) werden in Modulen zusammengefasst.

Abbildung 1: Liste der VBA-Module eines Word-Projekts

2 VBA-Makros

2.1 Export_VBA_Modules

Dieses Modul (s. Listing 2) exportiert den Inhalt aller Module (vgl. Abbildung 1) jeweils in eine Textdatei. Der entsprechende Dateizusatz lautet "*.txt".

```
Sub Export_VBA_Modules()
    ' Den Inhalt aller Module jeweils als TextDatei (*.txt) exportieren.
    Dim objDoc   As Document
    Dim vbProj   As VBIDE.VBProject
    Dim vbComp   As VBIDE.VBComponent
    Dim vbMod    As VBIDE.CodeModule
    Dim lngLines As Long
    Dim intCount As Integer
    Dim strCode  As String, strPath As String, strFileNm As String, strModuleNm As String
    Dim fs       As FileSystemObject
    Dim ts       As Scripting.TextStream
    On Error GoTo Error_Point
    Set objDoc = ActiveDocument
    objDoc.Select
    With objDoc
        Set vbProj = .VBProject
        strPath = .Path
    End With
    Set fs = CreateObject("Scripting.FileSystemObject")
    For Each vbComp In vbProj.VBComponents
        strModuleNm = vbComp.Name
        If vbComp.Type = vbext_ct_StdModule Then
            Set vbMod = vbComp.CodeModule
            lngLines = vbMod.CountOfLines
            If lngLines <> 0 Then
                strCode = vbMod.Lines(1, lngLines)
                If Len(Trim(strCode)) > 0 Then
                    intCount = intCount + 1
                    strFileNm = strPath & Application.PathSeparator & strModuleNm & ".txt"
                    Set ts = fs.CreateTextFile(strFileNm, True)
                    With ts
                        .Write strCode
                        .Close
                    End With
                End If
            End If
        End If
    Next vbComp
    MsgBox "Fertig! " & CStr(intCount) & " Module als Textdateien exportiert!", _
       vbExclamation, "Export_VBA_Modules"
Exit_Point:
    Set objDoc = Nothing
    Set vbMod = Nothing
    Set ts = Nothing
    Set fs = Nothing
    Exit Sub
Error_Point:
    If Err.Number = 76 Then 'Pfad nicht gefunden.
        fs.CreateFolder strPath
        Resume
    Else
        MsgBox Err.Numer & " - " & Err.Description
    End If
    Resume Exit_Point
End Sub
```

Listing 2: Alle Module jeweils als Textdatei exportieren

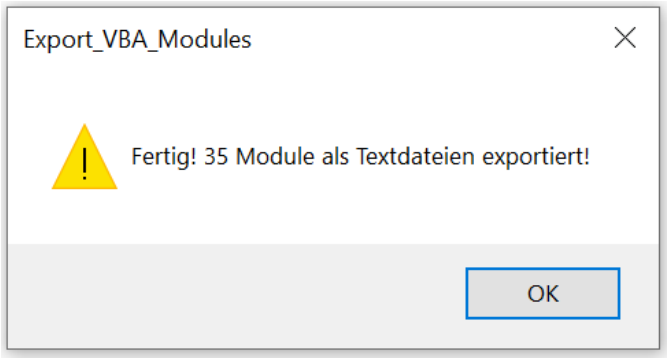

Abbildung 2: Erfolgsmeldung der Prozedur 'Export_VBA_Modules'

2.2 TextfileProcessing

Die Prozedur '**TextfileProcessing**' importiert die zuvor exportierten Textdateien in das Zieldokument '**AlleModule.docx**'. Dieses Zieldokument wird ggf. gelöscht und neu erstellt, falls es bereits existiert.

```
Sub TextfileProcessing()
    ' Zuerst die Prozedur 'Export_VBA_Code' in 'ModulesProc' ausführen.
    ' Dann mit der Prozedur 'FileProcessing' die erstellten Textdateien mit dem
    ' VBA-Code aller Module in das Zieldokument 'AlleModule.docx' einstellen.
    Const strDocName    As String = "AlleModule.docx"
    Dim objDoc          As Document ' Zieldokument
    Dim objRng          As Range    ' Bereich im Zieldokument
    Dim intChoice       As Integer  ' Entscheidung
    Dim intCount        As Integer  ' Zähler f. Textdateien
    Dim intInputNum     As Integer  ' Dateinummer f. Input
    Dim intOutputNum    As Integer  ' Dateinummer f. Output
    Dim intFileCount    As Integer  ' Dateizähler
    Dim strFolder       As String   ' aktueller Ordner
    Dim strFileNm       As String   ' Dateiname
    Dim strFileCont     As String   ' Dateiinhalt
    Dim strPath         As String   ' Pfad
    Dim strFullName     As String   ' vollständiger Name des Zieldokuments

    ' Ordner bestimmen, der die relevanten Textdateien enthält.
    strFolder = ActiveDocument.Path & Application.PathSeparator

    ' Vollständiger Name des Zieldokuments.
    strFullName = strFolder & strDocName

    ' Zieldokument löschen, falls schon vorhanden.
    If CheckFile(strFullName) = True Then
        MsgBox "Das alte Zieldokument '" & strFullName & "' wird gelöscht", vbExclamation,
"FileProcessing"
        Kill strFullName
    End If

    ' Zieldokument (ggf. neu) erstellen.
    Set objDoc = Documents.Add(DocumentType:=wdNewBlankDocument)
    objDoc.Select

    ' Zieldokument speichern, aber nicht schließen.
    objDoc.SaveAs FileName:=strFullName, FileFormat:=wdFormatXMLDocument

    ' Zieldokument verbergen.
    objDoc.Windows(1).Visible = False

    ' Die erste Textdatei im Ordner bestimmen.
    strFileNm = Dir(strFolder & "*.txt")
```

```
    ' Eine Schleife aufsetzen, um jede Textdatei zu durchlaufen.
```

```
    Do While strFileNm <> ""
        intChoice = MsgBox("Textdatei auswählen?", vbQuestion + vbYesNoCa.ncel, strFileNm)
        Select Case intChoice
            Case vbCancel
                GoTo Exit_Point
            Case vbNo:
                GoTo Next_File
                intReject = intReject + 1
            Case vbYes
                intCount = intCount + 1
        End Select

        If intFileCount Mod 10 Then
            Debug.Print "Bearbeite Datei" & intFileCount & "  :  " & strFileNm
        End If

        ' Textdatei öffnen.
        intInputNum = FreeFile
        Open strFolder & strFileNm For Input As #intInputNum
        ' Inhalt der Textdatei einer Variablen zuweisen.
        strFileCont = Input$(LOF(intInputNum), 1)

        ' Textdatei am Ende des Zieldokuments einfügen.
        Set objRng = objDoc.Content
        With objRng
            .Start = .End
            .InsertFile FileName:=strFolder & strFileNm
        End With

        ' Textdatei schließen.
        Close #intInputNum

        ' Die Ausgabedatei zum Schreiben vorbereiten.
        intOutputNum = FreeFile
        Open strFolder & strFileNm For Output As #intOutputNum
        Print #intOutputNum, strFileCont
        Close #intOutputNum
Next_File:
        ' Nächste Textdatei holen.
        strFileNm = Dir
    Loop
    ' Zieldokument wieder anzeigen.
    objDoc.Windows(1).Visible = True
    ' Zieldokument speichern.
    objDoc.SaveAs FileName:=strFullName, FileFormat:=wdFormatXMLDocument
    MsgBox CStr(intCount - intRejected) & " von " & CStr(intCount) & " Textdateien
ausgewählt!", _
    vbExclamation, "TextfileProcessing"
Exit_Point:
    Exit Sub
Error_Point:
    Resume Exit_Point
End Sub
```

Listing 3: Exportierte Textdateien auswählen und in des Zieldokument importieren

```
Function CheckFile(strFullPath As String) As Boolean
    ' Die Existenz einer Datei bestimmen.
    Dim strCheckPath As String
    strCheckPath = Dir(strFullPath)
    If Len(strCheckPath) > 1 Then
        CheckFile = True
    Else
        CheckFile = False
    End If
End Function
```

Listing 4: Die Existenz einer Textdatei bestimmen

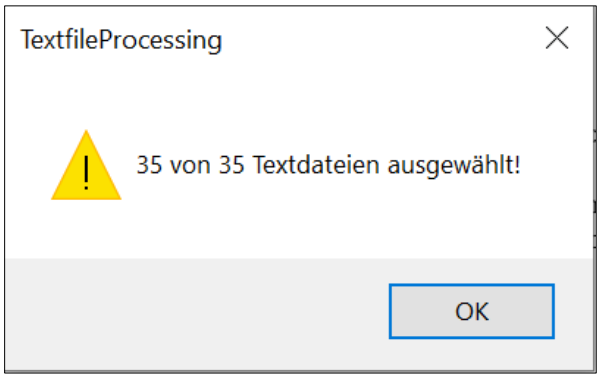

Abbildung 3: Erfolgsmeldung der Prozedur 'TextfileProcessing'

3 Zusammenfassung

Wenn ein WORD-Projekt viele Module enthält, kann ihr jeweiliger Export als Textdatei und anschließender Import in ein WORD-Zieldokument mit maßgeschneiderten VBA-Prozeduren schnell und sicher durchgeführt werden. – Es ist verdrießlich, diese Vorgänge zigmal schrittweise manuell durchzuführen.

Bei 35 Modulen (vgl. Abbildung 3) beträgt die Laufzeit der vorgestellten Prozeduren (einschl. Auswahl der zu importierenden Textdateien durch den Benutzer) ungefähr 1 Minute.