

Neu: Word-Dokument mit Benutzerformular bereinigen

Volker Thormählen, 5. Mai 2021

Inhaltsverzeichnis

1	Aufgabenstellung	2
2	Projekt	2
3	Benutzerformular	2
4	Module	4
4.1	Cleaning	4
4.2	Stetup	5
4.3	Tools	5
5	Quellcode	5
5.1	Modul <i>Cleaning</i>	5
5.2	Modul <i>Setup</i>	21
5.3	Modul <i>Tools</i>	23

1 Aufgabenstellung

In diesem Beitrag werden Ergänzungen eines Benutzerformulars beschrieben, das vom Autor bereits im Jahr 2017 veröffentlicht wurde¹.

- Das ergänzte Benutzerformular enthält im Rahmen „Suchen und ersetzen“ (s. Abbildung 2) zwei zusätzliche Optionsfelder.
- Der VBA-Quellcode zu allen 13 Optionsfeldern (s. Abbildung 2) wird präsentiert.

2 Projekt

Das zugehörige Projekt enthält 1 Benutzerformular mit dem Namen *frmGetFile* (s. Abbildung 1) sowie 3 Module mit dem Namen *Cleaning*, *Setup* bzw. *Tools*:

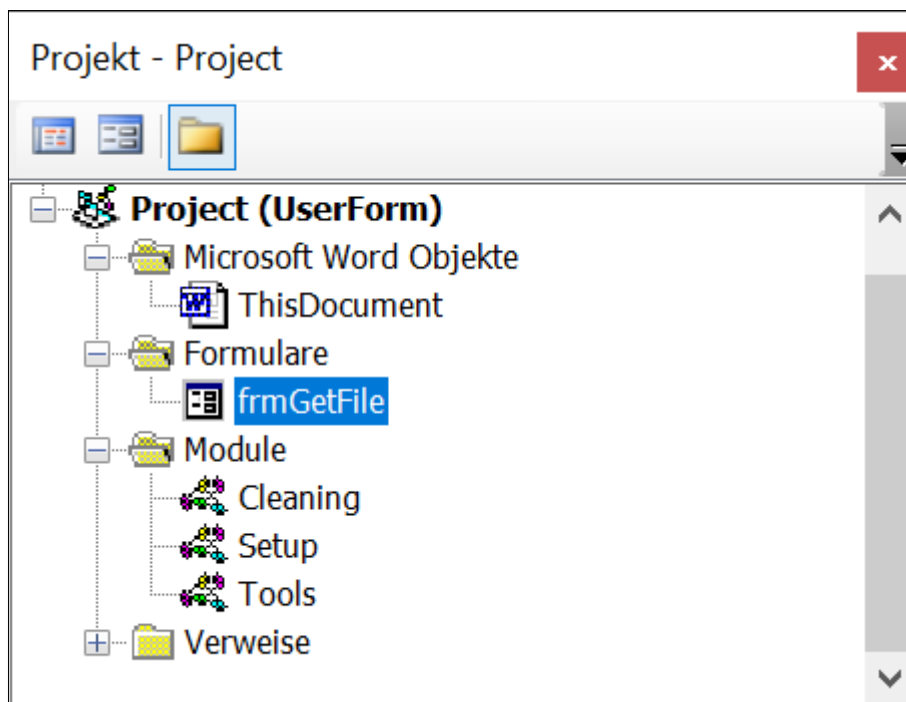


Abbildung 1: Das Projekt

3 Benutzerformular

Gruppirt in 5 Rahmen enthält das Benutzerformular *frmGetFile* insgesamt 13 Optionsfelder. In einem ersten Schritt muss zuvor das zu bereinigende Word-Dokument (sog. Zieldokument) interaktiv ausgewählt werden (s. Abbildung 2).

Außerdem enthält das Benutzerformular am unteren Rand vier weitere Schaltflächen zum Aufruf entsprechender Ereignisprozeduren:

¹ <https://www.dr-thormaehlen.de/WORD/Benutzerformular.pdf>

Bezeichnung der Schaltfläche	Aufgabe der Schaltfläche
1. Schritt: ...	Interaktive Auswahl des zu bereinigenden Zieldokuments.
OK	Bewirkt die Ausführung von interaktiven Bereinigungsprozeduren für die ausgewählten Optionsfelder.
Abbruch	Bewirkt den Abbruch bzw. das Ende des Benutzerformulars.
Alles zurücksetzen	Setzt der Wert aller Optionsfelder zurück auf den Anfangswert: False.
Alles auswählen	Setzt den Wert aller Optionsfelder auf „ausgewählt“: True.

Tabelle 1: Schaltflächen im Benutzerformular zum Aufruf von Ereignisprozeduren

Interaktiv ausgewähltes WORD-Dokument bereinigen

1. Schritt: Hier das zu bereinigende Word-Dokument auswählen.

Zeichenformatierung

- Unterstreichung in Kursivschrift ändern
- 3 Punkte höher gestellten Text in Hochstellung umwandeln

Leerzeichen, Tabulatoren und Umbrüche

- Zusätzliche Tabulatoren entfernen
- Multiple Leerzeichen durch 1 Leerzeichen ersetzen
- Leerzeichen um Absatzumbrüche entfernen
- Multiple Absatzumbrüche durch 1 Absatzumbruch ersetzen
- Abstand vor/nach Bindestrich/Auslassungszeichen korrigieren

Anführungszeichen und Zeichensetzung

- Gerade Anführungszeichen in typografische umwandeln o. umgekehrt
- Zeichensetzung (Interpunktion) prüfen

Ungültige Zeichen

- I-Zeichen, die 1 darstellen, in 1 umwandeln
- O-Zeichen, die 0 darstellen, in 0 umwandeln

Suchen und ersetzen

- Beliebige Zeichenfolge suchen und ersetzen
- Fußnoten ohne Satzendezeichen berichtigen

OK Abbruch Alles zurücksetzen Alles auswählen

Abbildung 2: Benutzerformular

4 Module

Das Projekt enthält 3 Module (s. Tabelle 2):

Modul	Inhalt	Prozeduren
<i>Cleaning</i>	Prozeduren zum Bereinigen des Zieldokuments.	13
<i>Setup</i>	Prozeduren zum Einrichten des Benutzerformulars <i>frmGetFile</i> .	6
<i>Tools</i>	Prozeduren zur interaktiven Auswahl des WORD-Zieldokuments.	2

Tabelle 2: Module des Projekts

4.1 Cleaning

Die 13 Prozeduren im Modul *Cleaning* führen folgende Aufgaben aus (s. Tabelle 3):

1. Unterstreichung in Kursivschrift ändern u. Unterstreichung aufheben.
2. Drei Punkte höher gestellten Text in Hochstellung umwandeln und Schriftgrad erhöhen.
3. Zusätzliche Tabulatoren entfernen.
4. Multiple Leerzeichen durch 1 Leerzeichen ersetzen.
5. Leerstelle(n) vor/nach Absatzmarken entfernen.
6. Ein oder mehr Leerzeichen vor/hinter Bindestrich entfernen.
7. Leerstelle(n) vor/nach Auslassungszeichen entfernen.
8. Gerade in typografische Anführungszeichen umwandeln oder umgekehrt.
9. Zeichensetzung prüfen.
10. l-Zeichen, die keine eins darstellen, in 1 umwandeln.
11. 0-Zeichen, die keine null darstellen, in 0 umwandeln.
12. Zeichenfolge finden und ersetzen.
13. Fehlendes Zeichen am Satzende in Fußnote einfügen.

Tabelle 3: Aufgaben der 13 Bereinigungsverfahren im Modul *Cleaning*

Die 13 Prozeduren im Modul *Cleaning* benötigen folgende Deklarationen von Konstanten bzw. Variablen (s. Tabelle 4):

Konstante/Variable	Datentyp	Kommentar
Const conLast	String = ".?!"	Zeichen für Satzende
Dim bolOption	Boolean	Optionswert
Dim CurrentColor	WdColorIndex	Farbindex
Dim ftIdx	Integer	Schleifenzähler f. Fußnoten
Dim ftRng	Range	Fußnotenbereich
Dim intChanges	Integer	Zähler f. Ersetzungen
Dim intChoice	Integer	Auswahl d. Benutzers
Dim intHits	Integer	Trefferzähler
Dim intLoop	Integer	Schleifenzähler
Dim intUpdCnt	Integer	Zähler f. Änderungen
Dim lngAsk	Long	Antwort auf Abfrage
Dim lngChrCnt	Long	Anfängliche Zeichenzahl im Dokument
Dim lngCnt	Long	Zähler f. entfernte Zeichen
Dim lngHighLight	Long	Farbwert f. Hervorhebungen
Dim lngLoop	Long	Schleifenzähler
Dim lngRetVal	Long	Rückgabewert d. MsgBox
Dim objRng	Range	Textbereich
Dim rngDoc	Range	Suchbereich
Dim strFindTxt	String	Suchbegriff
Dim strLast *)	String * 1	Letztes Zeichen einer Fußnote
Dim strListSep *)	String * 1	Listentrennzeichen
Dim strReplTxt	String	Ersatzbegriff
Dim StyleNormal	Style	Normaler Schreibstil
Dim varFind	Variant	Suchbegriff
Dim varFindList	Variant	Suchliste
Dim varRepl	Variant	Ersatzbegriff
*) Zeichenfolgevariable der Länge 1 Byte.		

4.2 Stetup

In diesem Modul werden mit der Subprozedur *UserForm_Initialize* die Anfangswerte des Benutzerformulars mit dem Namen *frmSetFile* gesetzt. Die übrigen Ereignisprozeduren werden durch die Schaltflächen des Benutzerformulars ausgelöst.

4.3 Tools

Das Modul *Tools* (dt. Werkzeuge) enthält nur zwei Prozeduren:

- Mit der Subprozedur *OpenWordFile* wird das WORD-Zieldokument durch den Benutzer interaktiv ausgewählt.
- Mit der Funktionsprozedur *ProcExists* wird das Vorkommen einer Subprozedur im Modul *Cleaning* überprüft (s. Tabelle 2).

5 Quellcode

5.1 Modul *Cleaning*

```

Option Explicit
1 Sub ChangeUnderlingToItalic(docTgt As Document, strCaption As String)
2   ' 1. Unterstreichung in Kursivschrift ändern u. Unterstreichung aufheben.
3   Dim intChanges As Integer ' Zähler f. Ersetzungen
4   Dim intHits As Integer ' Trefferzähler
5   Dim lngRetVal As Long ' Rückgabewert d. MsgBox
6   Dim objRng As Range ' Suchbereich
7   Application.ScreenUpdating = True
8   Set objRng = docTgt.Range(Start:=0, End:=0)
9   With objRng.Find
10    .Text = ""
11    .ClearFormatting
12    With .Font
13      .Underline = wdUnderlineSingle
14      .Italic = False
15    End With
16    .Forward = True
17    .Wrap = wdFindContinue
18    .Format = True
19    .MatchCase = False
20    .MatchWholeWord = False
21    .MatchWildcards = False
22    .MatchSoundsLike = False
23    .MatchAllWordForms = False
24    .Execute
25    Do While .Found = True
26      intHits = intHits + 1
27      With objRng
28        .HighlightColorIndex = wdTurquoise
29        .Duplicate.Select
30      End With
31      lngRetVal = MsgBox("Ersetzen?" & vbCrLf & _
32        objRng.Text, vbYesNo + vbQuestion, strCaption)
33      If lngRetVal = vbYes Then
34        intChanges = intChanges + 1
35        With objRng.Font
36          .Underline = wdUnderlineNone
37          .Italic = True
38        End With
39      End If
40      objRng.Collapse direction:=wdCollapseEnd
41    .Execute
42  Loop
43 End With

```

```

44     MsgBox CStr(intHits) & " Fundstelle(n), davon " & CStr(intChanges) & " ersetzt!",
45 vbExclamation, strCaption
46     Application.ScreenUpdating = True
47 End Sub

1  Sub RaisedTextToSuperscript(docTgt As Document, strCaption As String)
2  ' 2. Drei Punkte höher gestellten Text in Hochstellung umwandeln u. Schriftgrad erhöhen.
3  Dim intChanges As Integer ' Zähler f. Ersetzungen
4  Dim intHits As Integer ' Trefferzähler
5  Dim lngRetVal As Long ' Rückkehrcode d. MsgBox
6  Dim objRng As Range ' Suchbereich
7  Application.ScreenUpdating = True
8  Set objRng = docTgt.Range(Start:=0, End:=0)
9  With objRng.Find
10     .ClearFormatting
11     .Replacement.ClearFormatting
12     .Text = ""
13     With .Font
14         .Position = 3
15     End With
16     .Forward = True
17     .Wrap = wdFindContinue
18     .Format = True
19     .MatchCase = False
20     .MatchWholeWord = False
21     .MatchWildcards = False
22     .MatchSoundsLike = False
23     .MatchAllWordForms = False
24     .Execute
25     Do While .Found = True
26         intHits = intHits + 1
27         With objRng
28             .HighlightColorIndex = wdTurquoise
29             .Duplicate.Select
30         End With
31         lngRetVal = MsgBox("Ersetzen?" & vbCr & _
32             objRng.Text, vbYesNo + vbQuestion, strCaption)
33         If lngRetVal = vbYes Then
34             intChanges = intChanges + 1
35             With objRng.Font
36                 .Size = 14
37                 .Position = 0
38                 .Superscript = True
39             End With
40         End If
41         objRng.Collapse direction:=wdCollapseEnd
42     .Execute
43     Loop
44 End With
45 Application.ScreenUpdating = True
46 MsgBox CStr(intHits) & " Fundstelle(n), davon " & CStr(intChanges) & " ersetzt!",
47 vbExclamation, strCaption
48 End Sub

1  Sub RemoveExtraneousTabs(docTgt As Document, strCaption As String)
2  ' 3. Zusätzliche Tabulatoren entfernen
3  Dim intChanges As Integer ' Zähler f. Ersetzungen
4  Dim intHits As Integer ' Trefferzähler
5  Dim lngRetVal As Long ' Rückkehrcode d. MsgBox
6  Dim objRng As Range ' Suchbereich
7  Application.ScreenUpdating = True
8  Set objRng = docTgt.Range(Start:=0, End:=0)
9  With objRng.Find
10     .ClearFormatting
11     .Text = "^t{2;}"
12     .Forward = True
13     .Wrap = wdFindContinue

```

```

14     .Format = True
15     .MatchCase = False
16     .MatchWholeWord = False
17     .MatchWildcards = True
18     .MatchSoundsLike = False
19     .MatchAllWordForms = False
20     .Execute
21     Do While .Found = True
22         intHits = intHits + 1
23         With objRng
24             .HighlightColorIndex = wdTurquoise
25             .Duplicate.Select
26         End With
27         lngRetVal = MsgBox("Ersetzen?" & vbCrLf & _
28             objRng.Text, vbYesNo + vbQuestion, strCaption)
29         If lngRetVal = vbYes Then
30             intChanges = intChanges + 1
31             objRng.Text = Chr(9)
32         End If
33         objRng.Collapse direction:=wdCollapseEnd
34         .Execute
35     Loop
36     End With
37     MsgBox CStr(intHits) & " Fundstelle(n), davon " & CStr(intChanges) & " ersetzt!",
38     vbExclamation, strCaption
39     Application.ScreenUpdating = True
40 End Sub

1  Sub ReplaceMultipleSpaces(docTgt As Document, strCaption As String)
2  ' 4. Multiple Leerzeichen durch 1 Leerzeichen ersetzen.
3  Dim intChanges As Integer      ' Zähler f. Ersetzungen
4  Dim intHits As Integer        ' Trefferzähler
5  Dim lngRetVal As Long         ' Rückkehrcode d. MsgBox
6  Dim objRng As Range           ' Suchbereich
7  Dim strListSep As String * 1  ' Listentrennzeichen
8  strListSep = Application.International(wdListSeparator) 'Listentrennzeichen ermitteln
9  Application.ScreenUpdating = True
10 Set objRng = docTgt.Range(Start:=0, End:=0)
11 With objRng
12     With .Find
13         .ClearFormatting
14         .Text = "[ ]{2}" & strListSep & "]"
15         .Forward = True
16         .Wrap = wdFindContinue
17         .Format = True
18         .MatchCase = False
19         .MatchWholeWord = False
20         .MatchWildcards = True
21         .MatchSoundsLike = False
22         .MatchAllWordForms = False
23         .Execute
24     Do While .Found = True
25         intHits = intHits + 1
26         With objRng
27             .HighlightColorIndex = wdTurquoise
28             .Duplicate.Select
29         End With
30         lngRetVal = MsgBox("Ersetzen?" & vbCrLf & _
31             objRng.Text, vbYesNo + vbQuestion, strCaption)
32         If lngRetVal = vbYes Then
33             intChanges = intChanges + 1
34             objRng.Text = Space(1)
35         End If
36         objRng.Collapse direction:=wdCollapseEnd
37         .Execute
38     Loop
39 End With

```

```

40     End With
41     MsgBox CStr(intHits) & " Fundstelle(n), davon " & CStr(intChanges) & " ersetzt!",
42     vbExclamation, strCaption
43     Application.ScreenUpdating = True
44 End Sub

1  Sub DeleteWhiteSpacesAroundReturns(docTgt As Document, strCaption As String)
2  ' 5. Leerstelle(n) vor/nach Absatzmarken entfernen.
3  Dim intChanges      As Integer ' Zähler f. Ersetzungen
4  Dim intHits         As Integer ' Trefferzähler
5  Dim intLoop         As Integer ' Schleifenzähler
6  Dim lngHighLight   As Long    ' Farbwert f. Ersetzungen
7  Dim lngRetVal       As Long    ' Rückgabewert d. MsgBox
8  Dim objRng          As Range   ' Suchbereich
9  Dim varFindList     As Variant ' Suchliste
10 Application.ScreenUpdating = True
11 ' Leerstelle(n) vor/nach Absatzmarken entfernen.
12 varFindList = Array("^w^13", "^13^w") ' Suchbegriffe
13 lngHighLight = Options.DefaultHighlightColorIndex
14 For intLoop = LBound(varFindList) To UBound(varFindList)
15     Set objRng = docTgt.Range(Start:=0, End:=0)
16     With objRng.Find
17         .ClearFormatting
18         .Text = varFindList(intLoop)
19         .Forward = True
20         .Wrap = wdFindContinue
21         .Format = True
22         .MatchCase = False
23         .MatchWholeWord = False
24         .MatchWildcards = False
25         .MatchSoundsLike = False
26         .MatchAllWordForms = False
27         .Execute
28     Do While .Found = True
29         intHits = intHits + 1
30         With objRng
31             .HighlightColorIndex = wdYellow
32             .Duplicate.Select
33         End With
34         lngRetVal = MsgBox("Ersetzen?" & vbCr & _
35         objRng.Text, vbYesNo + vbQuestion, strCaption)
36         If lngRetVal = vbYes Then
37             intChanges = intChanges + 1
38             objRng.Text = Chr(13)
39         End If
40         With objRng
41             .HighlightColorIndex = wdNoHighlight
42             .Collapse direction:=wdCollapseEnd
43         End With
44         .Execute
45     Loop
46 End With
47 Next intLoop
48 Options.DefaultHighlightColorIndex = lngHighLight
49 MsgBox CStr(intHits) & " Fundstelle(n), davon " & _
50     CStr(intChanges) & " ersetzt!", vbExclamation, strCaption
51
52 ' Zwei weitere Prozeduren aufrufen.
53 If ProcExists(docTgt, "Cleaning", "RemoveSpacesBeforeParagraphBreak") Then
54     RemoveSpacesBeforeParagraphBreak ' Leerzeichen VOR Absatzumbruch entfernen
55 End If
56
57 If ProcExists(docTgt, "Cleaning", "RemoveSpacesAfterParagraphBreak") Then
58     RemoveSpacesAfterParagraphBreak ' Leerzeichen NACH Absatzumbruch entfernen
59 End If
60 Application.ScreenUpdating = True
61 End Sub

```



```

1 Sub RemoveSpacesBeforeParagraphBreak()
2 ' 5.1 Leerzeichen VOR Absatzumbruch entfernen
3 Dim lngChrCnt As Long ' anfängliche Zeichenzahl im Dokument
4 Dim lngCnt As Long ' Zähler f. entfernte Zeichen
5 lngChrCnt = ActiveDocument.Characters.Count
6 Application.ScreenUpdating = False
7 With ActiveDocument.Content.Find
8 .ClearFormatting
9 .Text = "^w^p"
10 With .Replacement
11 .ClearFormatting
12 .Text = "^p"
13 End With
14 .MatchCase = False
15 .MatchWholeWord = False
16 .MatchWildcards = False
17 .Execute Replace:=wdReplaceAll
18 End With
19 Application.ScreenUpdating = True
20 MsgBox ActiveDocument.Characters.Count
21 lngCnt = ActiveDocument.Characters.Count - lngChrCnt
22 MsgBox "Fertig!", vbExclamation, "RemoveSpacesAfterParagraphBreak: " & CStr(lngCnt)
23 End Sub

```

```

1 Sub RemoveSpacesAfterParagraphBreak()
2 ' 5.2 Leerzeichen NACH Absatzumbruch entfernen
3 Dim lngChrCnt As Long ' anfängliche Zeichenzahl im Dokument
4 Dim lngCnt As Long ' Zähler f. entfernte Zeichen
5 lngChrCnt = ActiveDocument.Characters.Count
6 Application.ScreenUpdating = False
7 With ActiveDocument.Content.Find
8 .ClearFormatting
9 .Text = "^p^w"
10 .Style = ActiveDocument.Styles(wdStyleNormal)
11 .Replacement.ClearFormatting
12 .Replacement.Text = "^p"
13 .MatchCase = False
14 .MatchWholeWord = False
15 .MatchWildcards = False
16 .Execute Replace:=wdReplaceAll
17 End With
18 Application.ScreenUpdating = True
19 lngCnt = ActiveDocument.Characters.Count - lngChrCnt
20 MsgBox "Fertig!", vbExclamation, "RemoveSpacesAfterParagraphBreak: " & CStr(lngCnt)
21 End Sub

```

```

1 Sub RemoveSpacesAroundHyphens(docTgt As Document, strCaption As String)
2 ' 6. Ein oder mehr Leerzeichen vor/hinter Bindestrich entfernen.
3 Dim CurrentColor As WdColorIndex ' Farbindex
4 Dim intChanges as Integer ' Zähler f. Ersetzungen
5 Dim intHits As Integer ' Trefferzähler
6 Dim lngRetVal As Long ' Rückgabewert d. MsgBox
7 Dim objRng As Range ' Suchbereich
8 Dim strListSep As String * 1 ' Listentrennzeichen
9 On Error GoTo ErrorHandler
10 With Application
11 .ScreenUpdating = True
12 strListSep = .International(wdListSeparator) ' Listentrennzeichen ermitteln
13 CurrentColor = .Options.DefaultHighlightColorIndex
14 .Options.DefaultHighlightColorIndex = wdYellow
15 End With
16 Set objRng = docTgt.Range(0, 0)
17 With objRng
18 With .Find
19 .ClearFormatting
20 .Text = "(*)[ ]{1} & strListSep & "})" ' vor Bindestrich

```

```

21         With .Replacement
22             .ClearFormatting
23             .Text = "\1\3"
24         End With
25         .Wrap = wdFindStop
26         .Forward = True
27         .Format = False
28         .MatchCase = False
29         .MatchWholeWord = False
30         .MatchAllWordForms = False
31         .MatchSoundsLike = False
32         .MatchWildcards = True
33         .Execute
34         Do While .Found = True
35             intHits = intHits + 1
36             objRng.Duplicate.Select
37             lngRetVal = MsgBox("Leerstelle(n) VOR Bindestrich ersetzen?", vbYesNo +
38 vbQuestion, strCaption)
39             If lngRetVal = vbYes Then
40                 intChanges = intChanges + 1
41                 .Execute Replace:=wdReplaceOne
42                 objRng.Collapse direction:=wdCollapseEnd
43             End If
44             .Execute
45         Loop
46     End With
47 End With
48 Set objRng = docTgt.Range(0, 0)
49 With objRng
50     With .Find
51         .ClearFormatting
52         .Text = "(*)[ ]{1} & strListSep & "}" ' nach Bindestrich
53     With .Replacement
54         .ClearFormatting
55         .Text = "\1"
56     End With
57     .Wrap = wdFindStop
58     .Forward = True
59     .Format = False
60     .MatchCase = False
61     .MatchWholeWord = False
62     .MatchAllWordForms = False
63     .MatchSoundsLike = False
64     .MatchWildcards = True
65     .Execute
66     Do While .Found = True
67         intHits = intHits + 1
68         objRng.Duplicate.Select
69         lngRetVal = MsgBox("Leerstelle(n) NACH Bindestrich ersetzen?", vbYesNo +
70 vbQuestion, strCaption)
71         If lngRetVal = vbYes Then
72             intChanges = intChanges + 1
73             .Execute Replace:=wdReplaceOne
74             objRng.Collapse direction:=wdCollapseEnd
75         End If
76         .Execute
77     Loop
78 End With
79 End With
80 MsgBox "Fertig! " & CStr(intHits) & " Fundstelle(n), davon " & _
81 CStr(intChanges) & " ersetzt!", vbExclamation, strCaption
82 ExitPoint:
83 On Error Resume Next
84 With Application
85     .ScreenUpdating = True
86     .Options.DefaultHighlightColorIndex = CurrentColor
87 End With

```

```

88     Set objRng = Nothing
89     Exit Sub
90 ErrorHandler:
91     MsgBox "Laufzeitfehler: " & Err.Number & ", " & Err.Description, vbExclamation,
92     "Prozedur: 'RemoveSpacesAroundHyphens'"
93     Resume ExitPoint
94 End Sub

1  Sub RemoveSpacingAroundEllipses(docTgt As Document, strCaption As String)
2  ' 7. Leerstelle(n) vor/nach Auslassungszeichen entfernen
3  Dim intChanges As Integer ' Zähler f. Ersetzungen
4  Dim intHits As Integer ' Trefferzähler
5  Dim intLoop As Integer ' Schleifenzähler
6  Dim lngHighLight As Long ' Farbwert f. Hervorhebungen
7  Dim lngRetVal As Long ' Rückgabewert d. MsgBox
8  Dim objRng As Range ' Suchbereich
9  Dim strListSep As String * 1 ' Listentrennzeichen
10 Dim varFindList As Variant ' Suchliste
11 Application.ScreenUpdating = True
12 strListSep = Application.International(wdListSeparator)
13 varFindList = Array("[ ]{1" & strListSep & "}^0133", "^0133[ ]{1" & strListSep & "}") '
14 Suchbegriffe
15 lngHighLight = Options.DefaultHighlightColorIndex
16 For intLoop = LBound(varFindList) To UBound(varFindList)
17     Set objRng = docTgt.Range(Start:=0, End:=0)
18     With objRng.Find
19         .ClearFormatting
20         .Text = varFindList(intLoop)
21         .Forward = True
22         .Wrap = wdFindStop ' wdFindContinue
23         .Format = True
24         .MatchCase = False
25         .MatchWholeWord = False
26         .MatchWildcards = True
27         .MatchSoundsLike = False
28         .MatchAllWordForms = False
29         .Execute
30     Do While .Found = True
31         intHits = intHits + 1
32         With objRng
33             .HighlightColorIndex = wdYellow
34             .Duplicate.Select
35         End With
36         lngRetVal = MsgBox("Ersetzen?" & vbCr & _
37         objRng.Text, vbYesNo + vbQuestion, strCaption)
38         If lngRetVal = vbYes Then
39             intChanges = intChanges + 1
40             With objRng
41                 If intLoop = LBound(varFindList) Then
42                     .Text = LTrim(.Text) ' vor
43                 Else
44                     .Text = RTrim(.Text) ' nach
45                 End If
46             End With
47         End If
48         With objRng
49             .HighlightColorIndex = wdNoHighlight
50             .Collapse direction:=wdCollapseEnd
51         End With
52         .Execute
53     Loop
54 End With
55 Next intLoop
56 With Application
57     .Options.DefaultHighlightColorIndex = lngHighLight
58     .ScreenUpdating = True
59 End With

```

```

60     MsgBox CStr(intHits) & " Fundstelle(n), davon " & _
61           CStr(intChanges) & " ersetzt!", vbExclamation, strCaption
62 End Sub

1  Sub ConvertQuotes(docTgt As Document, strCaption As String)
2  ' 8. Gerade in typografische Anführungsstriche umwandeln oder umgekehrt.
3  Dim bolOption As Boolean ' Optionswert
4  Dim lngLoop As Long ' Scheifenzähler
5  Dim lngRetVal As Long ' Rückgabewert d. MsgBox
6  Dim objRng As Range ' Suchbereich
7  Dim varFind As Variant ' Suchbegriff
8  Dim varRepl As Variant ' Ersatzbegriff
9  Application.ScreenUpdating = False
10 lngRetVal = MsgBox("Auf 'Ja' klicken, um typografische in gerade Anführungszeichen
11 umzuwandeln." & vbCr & _
12 "Auf 'Nein' klicken, um gerade in typografische Anführungszeichen
13 umzuwandeln.", _
14 vbYesNoCancel, strCaption)
15 If lngRetVal = vbCancel Then
16 GoTo ExitPoint
17 Else
18 bolOption = Options.AutoFormatAsYouTypeReplaceQuotes
19 If lngRetVal = vbYes Then
20 varFind = Array(ChrW(132), ChrW(147), ChrW(148), ChrW(171), ChrW(130),
21 ChrW(145), ChrW(146))
22 varRepl = Array(Chr(34), Chr(34), Chr(34), Chr(34), Chr(39), Chr(39), Chr(39))
23 Options.AutoFormatAsYouTypeReplaceQuotes = False
24 For lngLoop = LBound(varFind) To UBound(varFind)
25 Set objRng = docTgt.Range
26 With objRng.Find
27 Do While .Execute(varFind(lngLoop))
28 With objRng
29 .Text = varRepl(lngLoop)
30 .Collapse wdCollapseEnd
31 End With
32 Loop
33 End With
34 Next lngLoop
35 ElseIf lngRetVal = vbNo Then
36 Options.AutoFormatReplaceQuotes = True
37 docTgt.Range.AutoFormat
38 End If
39 End If
40 With Application
41 .Options.AutoFormatAsYouTypeReplaceQuotes = bolOption
42 .ScreenUpdating = True
43 End With
44 MsgBox "ConvertQuotes: Normales Ende!", vbExclamation, strCaption
45 ExitPoint:
46 End Sub

1  Sub ProperPunctuation(docTgt As Document, strCaption As String)
2  ' 9. Zeichensetzung prüfen
3  Dim intUpdCnt As Integer ' Zähler f. Änderungen
4  Dim lngAsk As Long ' Rückgabewert d. MsgBox
5  Dim lngHighLight As Long ' Hervorhebungsfarbe f. Änderungen
6  Dim rngDoc As Range ' Suchbereich
7  Dim strFindTxt As String ' Suchbegriff
8  Dim strReplTxt As String ' Ersatztext
9  On Error GoTo ErrorHandler
10 lngHighLight = Options.DefaultHighlightColorIndex
11 Options.DefaultHighlightColorIndex = wdTurquoise
12 ' Kein Leerzeichen vor Satzzeichen!
13 strFindTxt = "[ ][,;:\!\\?]"
14 strReplTxt = vbNullString
15 Set rngDoc = docTgt.Range
16 With rngDoc.Find

```

```

17     .ClearFormatting
18     .Replacement.ClearFormatting
19     .Style = wdStyleNormal           ' nur bei Formatvorlage "Standard"
20     .Replacement.Highlight = True    ' Ersetzung hervorheben
21     .Font.Underline = wdUnderlineNone ' Hyperlinks (unterstrichene Textstellen) ausnehmen
22     Do While .Execute( _
23         FindText:=strFindTxt, _
24         MatchCase:=True, _
25         MatchWholeWord:=True, _
26         MatchWildcards:=True, _
27         Forward:=True, _
28         Wrap:=wdFindStop) = True
29         rngDoc.Select
30         lngAsk = MsgBox("Leerzeichen vor Satzzeichen gefunden! Entfernen?", vbYesNo,
31 strCaption)
32         If lngAsk = vbYes Then
33             With rngDoc
34                 .Text = LTrim(.Text) ' Leerzeichen davor entfernen.
35             End With
36             intUpdCnt = intUpdCnt + 1
37         End If
38         rngDoc.Collapse direction:=wdCollapseEnd
39     Loop
40 End With
41 ' Hinter Satzzeichen ist 1 Leerzeichen erforderlich!
42 strFindTxt = "[,;.:!\|?][! " & ChrW(34) & ChrW(13) & "]"
43 strReplTxt = Space(1)
44 Set rngDoc = ActiveDocument.Range
45 With rngDoc.Find
46     .ClearFormatting
47     .Replacement.ClearFormatting
48     .Style = wdStyleNormal           ' nur bei Formatvorlage "Standard"
49     .Replacement.Highlight = True    ' Ersetzung hervorheben
50     .Font.Underline = wdUnderlineNone ' Hyperlinks (unterstrichene Textstellen) ausnehmen
51     Do While .Execute( _
52         FindText:=strFindTxt, _
53         MatchCase:=True, _
54         MatchWholeWord:=True, _
55         MatchWildcards:=True, _
56         Forward:=True, _
57         Wrap:=wdFindStop) = True
58         rngDoc.Select
59         lngAsk = MsgBox("Kein Leerzeichen hinter einem Satzzeichen gefunden! Einfügen?",
60 vbYesNo, strCaption)
61         If lngAsk = vbYes Then
62             With Selection
63                 .Collapse direction:=wdCollapseStart
64                 .MoveRight Unit:=wdCharacter, Count:=1
65                 .InsertAfter Text:=strReplTxt ' Leerzeichen dahinter einfügen.
66             End With
67         End If
68         intUpdCnt = intUpdCnt + 1
69         rngDoc.Collapse direction:=wdCollapseEnd
70     Loop
71 End With
72 MsgBox CStr(intUpdCnt) & " Ersetzung(en) vor/hinter Satzzeichen durchgeführt.", _
73     vbInformation, strCaption
74 ExitPoint:
75     On Error Resume Next
76     Set rngDoc = Nothing
77     Options.DefaultHighlightColorIndex = lngHighLight
78     Exit Sub
79 ErrorHandler:
80     MsgBox "Laufzeitfehler: " & Err.Number & ", " & vbCrLf & _
81     Err.Description & ", " & Err.Source, vbCritical, "ProperPunctuation"
82     Resume ExitPoint
83 End Sub

```

```

1 Sub Satzzeichen_LZ(docTgt As Document, strCaption As String)
2 ' 9.2 Überflüssige Leerzeichen vor und hinter Satzzeichen entfernen.
3 ' Hinter Satzzeichen in der Markierung stets Leerzeichen, außer bei Gänsefüßchen.
4 ' Unterstrichene Textstellen (Hyperlinks) sind ausgenommen.
5 ' Ausgenommen ist auch, wenn hinter dem Satzzeichen eine Zahl folgt.
6 Dim CurrentColor As WdColorIndex ' Farbindex
7 Dim intChoice As Integer ' Auswahl d. Benutzer
8 Dim intUpdCnt As Integer ' Zähler f. Änderungen
9 Dim objRng As Range ' Suchbereich
10 Dim strFindTxt As String ' Suchbegriff
11 Dim strReplTxt As String ' Ersatzbegriff
12 Dim StyleNormal As Style ' Schreibstil
13 On Error GoTo ErrorHandler
14 Set objRng = ActiveDocument.Content
15 If Len(objRng.Text) = 0 Then
16 MsgBox "Nichts markiert!", vbExclamation, strCaption
17 GoTo ExitPoint
18 End If
19 objRng.Collapse direction:=wdCollapseStart
20 With Application
21 .ScreenUpdating = True
22 ' Aktuelle Hervorhebungsfarbe zwischenspeichern u. setzen.
23 CurrentColor = .Options.DefaultHighlightColorIndex
24 .Options.DefaultHighlightColorIndex = wdYellow
25 End With
26 ' Normalen Schreibstil vorgeben.
27 Set StyleNormal = ActiveDocument.Styles(wdStyleNormal)
28 ' Fehlendes Leerzeichen hinter Satzzeichen einfügen, OK!
29 strFindTxt = "([,;.:!\?])([A-Za-z])" ' finden
30 strReplTxt = "\1" & Space(1) & "\2" ' ersetzen
31 With objRng
32 With .Find
33 .ClearFormatting
34 .Text = strFindTxt
35 With .Replacement
36 .ClearFormatting
37 .Text = strReplTxt
38 .Highlight = True
39 End With
40 .Style = StyleNormal
41 .Forward = True
42 .Wrap = wdFindStop ' wdFindContinue ' wdFindAsk
43 .Format = True
44 .Font.Underline = wdUnderlineNone ' keine Hyperlinks
45 .MatchCase = False
46 .MatchWholeWord = False
47 .MatchAllWordForms = False
48 .MatchSoundsLike = False
49 .MatchWildcards = True
50 .Execute
51 End With ' Find
52 Do While .Find.Found
53 .Duplicate.Select
54 Select Case MsgBox("Fehlendes Leerzeichen hinter Satzzeichen einfügen?", _
55 vbYesNoCancel, strCaption)
56 Case vbCancel
57 MsgBox "Abbruch durch den Benutzer!", vbExclamation, _
58 "Fehlendes Leerzeichen hinter Satzzeichen einfügen!"
59 GoTo ExitPoint
60 Case vbYes
61 intUpdCnt = intUpdCnt + 1
62 With Selection
63 .MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
64 .Text = .Text & Space(1)
65 End With
66 End Select

```

```

67         .Collapse wdCollapseEnd
68         .Find.Execute
69     Loop
70 End With
71 ' Vor Satzzeichen oder schließender Klammer kein Leerzeichen.
72 objRng.Collapse direction:=wdCollapseStart
73 strFindTxt = "[ ]{1;}{[.,:;!\\?\\)}\\]}]" ' finden
74 strReplTxt = "\1" ' ersetzen
75 With objRng
76     .Collapse direction:=wdCollapseStart
77     With .Find
78         .ClearFormatting
79         .Text = strFindTxt
80         With .Replacement
81             .ClearFormatting
82             .Text = strReplTxt
83             .Highlight = True
84         End With
85         .Style = StyleNormal
86         .Forward = True
87         .Wrap = wdFindContinue
88         .Font.Underline = wdUnderlineNone
89         .Format = False
90         .MatchCase = False
91         .MatchWholeWord = False
92         .MatchAllWordForms = False
93         .MatchSoundsLike = False
94         .MatchWildcards = True
95     End With
96 End With
97 Do While .Find.Found
98     .Duplicate.Select
99     Select Case MsgBox("Leerzeichen vor Satzzeichen/schließender Klammer
100 entfernen?", vbYesNoCancel + vbDefaultButton1, "Ersetzen")
101     Case vbCancel
102         MsgBox "Abbruch durch den Benutzer!", vbExclamation, "Vor
103 Satzzeichen/schließender Klammer keine Leerzeichen!"
104         GoTo ExitPoint
105     Case vbYes
106         intUpdCnt = intUpdCnt + 1
107         .Text = Trim(Selection.Text)
108         Selection.MoveRight Unit:=wdCharacter, Count:=2
109     End Select
110     .Collapse direction:=wdCollapseEnd
111     .Find.Execute
112 Loop
113 End With
114 MsgBox "Fertig! " & CStr(intUpdCnt) & " überflüssige Leerzeichen entfernt!",
115 vbInformation, strCaption
116 ExitPoint:
117 On Error Resume Next
118 With Application
119     .ScreenUpdating = True
120     .Options.DefaultHighlightColorIndex = CurrentColor ' <--- Farbe zurücksetzen
121 End With
122 Set objRng = Nothing
123 Exit Sub
124 ErrorHandler:
125 MsgBox "Laufzeitfehler: " & Err.Number & ", " & Err.Description & ", Prozedur:
126 Satzzeichen_LZ"
127 Resume ExitPoint
128 End Sub

1 Sub ChangelsToOnes(docTgt As Document, strCaption As String)
2     ' 10. 1-Zeichen, die keine 'eins' darstellen, in '1' umwandeln.
3     Dim intChanges As Integer ' Zähler f. Ersetzungen
4     Dim intHits As Integer ' Trefferzähler

```

```

5 Dim objRng As Range ' Suchbereich
6 Set objRng = docTgt.Range(0, 0)
7 With objRng.Find
8     .ClearFormatting
9     .Text = "[0-9]{1;}1[0-9]{1;}"
10 With .Replacement
11     .Text = ""
12     .Highlight = True
13 End With
14 .Forward = True
15 .Wrap = wdFindContinue
16 .MatchWildcards = True
17 .Format = True
18 .MatchCase = False
19 .MatchWholeWord = False
20 .MatchAllWordForms = False
21 .MatchSoundsLike = False
22 .Execute
23 Do While .Found = True
24     intHits = intHits + 1
25     objRng.Duplicate.Select
26     If MsgBox("Ersetzen? " & objRng.Text, vbQuestion + vbYesNo, strCaption) = vbYes Then
27         objRng.Text = Replace(objRng.Text, "1", "1", Compare:=vbTextCompare)
28         intChanges = intChanges + 1
29     End If
30     Selection.Collapse direction:=wdCollapseEnd
31     .Execute
32 Loop
33 End With
34 Set objRng = ActiveDocument.Range(0, 0)
35 With objRng.Find
36     .Text = "1[0-9]{1;}"
37     .Replacement.Text = ""
38     .Forward = True
39     .Wrap = wdFindContinue
40     .MatchWildcards = True
41     .Format = False
42     .Execute
43 Do While .Found = True
44     intHits = intHits + 1
45     objRng.Duplicate.Select
46     If MsgBox("Ersetzen? " & objRng.Text, vbQuestion + vbYesNo, strCaption) = vbYes Then
47         objRng.Text = Replace(objRng.Text, "1", "1", Compare:=vbTextCompare)
48         intChanges = intChanges + 1
49     End If
50     Selection.Collapse direction:=wdCollapseEnd
51     .Execute
52 Loop
53 End With
54 Set objRng = ActiveDocument.Range(0, 0)
55 With objRng.Find
56     .Text = "[0-9]{1;}1"
57     With .Replacement
58         .Text = ""
59         .Highlight = True
60     End With
61     .Forward = True
62     .Wrap = wdFindContinue
63     .MatchWildcards = True
64     .Format = True
65     .Execute
66 Do While .Found = True
67     intHits = intHits + 1
68     objRng.Duplicate.Select
69     If MsgBox("Ersetzen? " & objRng.Text, vbQuestion + vbYesNo, strCaption) = vbYes Then
70         objRng.Text = Replace(objRng.Text, "1", "1", Compare:=vbTextCompare)
71         intChanges = intChanges + 1
72     End If

```



```

73         Selection.Collapse direction:=wdCollapseEnd
74         .Execute
75     Loop
76 End With
77     MsgBox CStr(intHits) & " Fundstelle(n), davon " & CStr(intChanges) & " ersetzt!",
78 vbExclamation, strCaption
79 End Sub

1  Sub ChangeOsToZeros(docTgt As Document, strCaption As String)
2  ' 11. 0-Zeichen, die keine null darstellen, in 0 umwandeln.
3  Dim intChanges As Integer ' Zähler f. Ersetzungen
4  Dim intHits As Integer ' Trefferzähler
5  Dim objRng As Range ' Suchbereich
6  Set objRng = docTgt.Range(0, 0)
7  With objRng.Find
8      .ClearFormatting
9      .Text = "[0-9]{1;}0[0-9]{1;}"
10     With .Replacement
11         .Text = ""
12         .Highlight = True
13     End With
14     .Forward = True
15     .Wrap = wdFindContinue
16     .MatchWildcards = True
17     .Format = True
18     .MatchCase = False
19     .MatchWholeWord = False
20     .MatchAllWordForms = False
21     .MatchSoundsLike = False
22     .Execute
23     Do While .Found = True
24         intHits = intHits + 1
25         objRng.Duplicate.Select
26         If MsgBox("Ersetzen? " & objRng.Text, vbQuestion + vbYesNo, strCaption) = vbYes Then
27             objRng.Text = Replace(objRng.Text, "0", "0", Compare:=vbTextCompare)
28             intChanges = intChanges + 1
29         End If
30         Selection.Collapse direction:=wdCollapseEnd
31         .Execute
32     Loop
33 End With
34 Set objRng = ActiveDocument.Range(0, 0)
35 With objRng.Find
36     .Text = "0[0-9]{1;}"
37     With .Replacement
38         .Text = ""
39         .Highlight = True
40     End With
41     .Forward = True
42     .Wrap = wdFindContinue
43     .MatchWildcards = True
44     .Format = True
45     .Execute
46     Do While .Found = True
47         intHits = intHits + 1
48         objRng.Duplicate.Select
49         If MsgBox("Ersetzen? " & objRng.Text, vbQuestion + vbYesNo, strCaption) = vbYes Then
50             objRng.Text = Replace(objRng.Text, "0", "0", Compare:=vbTextCompare)
51             intChanges = intChanges + 1
52         End If
53         Selection.Collapse direction:=wdCollapseEnd
54         .Execute
55     Loop
56 End With
57 Set objRng = ActiveDocument.Range(0, 0)
58 With objRng.Find
59     .Text = "[0-9]{1;}0"
60     With .Replacement

```

```

61         .Text = ""
62         .Highlight = True
63     End With
64     .Forward = True
65     .Wrap = wdFindContinue
66     .MatchWildcards = True
67     .Format = True
68     .Execute
69     Do While .Found = True
70         intHits = intHits + 1
71         objRng.Duplicate.Select
72         If MsgBox("Ersetzen? " & objRng.Text, vbQuestion + vbYesNo, strCaption) = vbYes Then
73             objRng.Text = Replace(objRng.Text, "0", "Ø", Compare:=vbTextCompare)
74             intChanges = intChanges + 1
75         End If
76         Selection.Collapse direction:=wdCollapseEnd
77     .Execute
78     Loop
79 End With
80 MsgBox CStr(intHits) & " Fundstelle(n), davon " & CStr(intChanges) & " ersetzt!",
81 vbExclamation, strCaption
82 End Sub

1 Sub FindReplaceSearchTerm(docTgt As Document, strCaption As String)
2     ' 12. Best. Zeichenfolge finden und ersetzen
3     Dim intChanges As Integer ' Zähler f. Ersetzungen
4     Dim intHits As Integer ' Trefferzähler
5     Dim objRng As Range ' Suchbereich
6     Dim strFindTxt As String ' Suchbegriff
7     Dim strReplTxt As String ' Ersatzbegriff
8     Set objRng = docTgt.Content
9     If Len(objRng) < 1 Then
10        MsgBox "Abbruch: Das aktuelle Dokument ist leer!", vbExclamation, strCaption
11        GoTo ExitPoint
12    End If
13    Do While Len(strFindTxt) < 1
14        strFindTxt = InputBox("Gesuchte Zeichenfolge", strCaption)
15        If strFindTxt = vbNullString Then
16            MsgBox "Abbruch: Eingabe des Suchbegriffs ist Pflicht!", vbExclamation, strCaption
17            GoTo ExitPoint
18        End If
19    Loop
20
21    Do While Len(strReplTxt) < 1
22        strReplTxt = InputBox("Ersatzbegriff", strCaption)
23        If strReplTxt = vbNullString Then
24            MsgBox "Abbruch: Eingabe d. Ersatzbegriffs ist Pflicht!", vbExclamation, strCaption
25            GoTo ExitPoint
26        End If
27    Loop
28    With objRng
29        With .Find
30            .ClearFormatting
31            .Text = strFindTxt
32            .Style = ActiveDocument.Styles(wdStyleNormal)
33            With .Replacement
34                .ClearFormatting
35                .Text = ""
36                .Highlight = True
37            End With
38            .MatchWholeWord = True
39            .Forward = True
40            .Format = False
41            .MatchCase = True
42            .MatchWildcards = False
43            .Wrap = wdFindStop
44            .Execute

```

```

45     End With
46     Do While .Find.Found
47         ' Die Auswahl auf den gesamten Absatz ausweiten: => objRng.Expand Unit:=wdParagraph
48         ' Fensterinhalt d. Dokumentfensters verschieben, bis der angegebene Suchbereich darin angezeigt wird:
49         ' ActiveWindow.ScrollIntoView obj:=Selection.Range, Start:=True
50         .Duplicate.Select ' Fundstelle markieren
51         intHits = intHits + 1
52         Select Case MsgBox("Ersetzen:" & _
53             vbCr & vbTab & strFindTxt & vbCr & "durch:" & _
54             vbCr & vbTab & strReplTxt, vbYesNoCancel, strCaption)
55             Case vbCancel
56                 GoTo ExitPoint
57             Case vbYes
58                 .Text = strReplTxt
59                 intChanges = intChanges + 1
60             Case Else
61                 ' nichts tun
62             End Select
63         .Collapse direction:=wdCollapseEnd
64         .Find.Execute
65     Loop
66     End With
67     MsgBox CStr(intHits) & " Fundstelle(n), davon " & CStr(intChanges) & " ersetzt!",
68     vbExclamation, strCaption
69     ExitPoint:
70     On Error Resume Next
71     Exit Sub
72     ErrorHandler:
73     MsgBox Err.Number & ", " & Err.Description & " in Prozedur 'FindReplaceSearchTerm'"
74     Resume ExitPoint
75     End Sub

```

```

1  Sub AddPeriodToFootnotes(docTgt As Document, strCaption As String)
2  ' 13. Fehlendes Satzendezeichen in Fußnote einfügen
3  Const conLast As String = ".?!" ' Zeichen für Satzende
4  Dim ftIdx As Integer ' Schleifenzähler f. Fußnoten
5  Dim ftRng As Range ' Fußnotenbereich
6  Dim intHits As Integer ' Trefferzählen (f. fehlendes Satzendezeichen)
7  Dim strLast As String * 1 ' Letztes Zeichen einer Fußnote
8  On Error GoTo ErrHandler
9  intHits = 0
10 For ftIdx = 1 To ActiveDocument.Footnotes.Count
11     Set ftRng = ActiveDocument.Footnotes(ftIdx).Range
12     With ftRng
13         strLast = Right(.Text, 1)
14         If InStr(1, conLast, strLast, vbTextCompare) = 0 Then
15             ' Satzendezeichen
16             .Collapse direction:=wdCollapseEnd
17             .Text = .Text & "."
18             intHits = intHits + 1
19         End If
20     End With
21 Next ftIdx
22 If intHits = 1 Then
23     MsgBox "1 fehlendes Zeichen am Ende einer Fußnote eingefügt.", vbInformation,
24     strCaption
25 Else
26     MsgBox CStr(intHits) & " fehlende Zeichen am Ende von Fußnoten eingefügt.",
27     vbInformation, strCaption
28 End If
29 ExitPoint:
30 On Error Resume Next
31 Exit Sub
32 ErrorHandler:
33 If Err.Number = 5941 Then
34     MsgBox "Keine Fußnoten im aktuellen Dokument gefunden", vbExclamation, strCaption
35 End If
36 Resume ExitPoint

```


5.2 Modul Setup

```
1 Option Base 1
2 Option Explicit
3 Dim strTgtFileNm As String

1 Private Sub UserForm_Initialize()
2     With frmGetFile
3         .Left = 10
4         .Top = 10
5         .StartupPosition = 0
6         .lblFileNm.Caption = ""
7         .cmdCancel.Enabled = True
8         .cmdStart.Enabled = False
9         .Show
10    End With
11 End Sub

1 Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
2     If CloseMode = vbFormControlMenu Then
3         Cancel = True
4         Call cmdCancel_Click
5     End If
6 End Sub

1 Private Sub cmdCancel_MouseMove(ByVal Button As Integer, _
2     ByVal Shift As Integer, ByVal x As Single, ByVal y As Single)
3     frmGetFile.cmdCancel.SetFocus
4 End Sub

1 Private Sub cmdStart_MouseMove(ByVal Button As Integer, _
2     ByVal Shift As Integer, ByVal x As Single, ByVal y As Single)
3     frmGetFile.cmdStart.SetFocus
4 End Sub

1 Private Sub cmdCancel_Click() ' Schaltfläche "Abbruch"
2     Dim intRetVal As Integer, strMsg As String
3     strMsg = "Sind Sie sicher, dass Sie abbrechen und beenden möchten?"
4     intRetVal = MsgBox(strMsg, vbQuestion + vbYesNo, "Abbruch?")
5     Select Case intRetVal
6         Case vbYes
7             frmGetFile.Hide
8             Unload frmGetFile
9         Case Else
10            Exit Sub
11    End Select
12 End Sub
```

```

1
1 Private Sub cmdStart_Click()      ' Schaltfläche "Start"
2     Dim ctrl      As Control      ' Steuerelement
3     Dim docTgt    As Document     ' Zieldokument
4     Dim lngRtnCode As Long        ' Rückgabewert der MsgBox
5     Dim strFileNm As String       ' Dateiname
6     Dim strMdlNm  As String       ' Modulname
7     Dim strProcNm As String       ' Prozedurname
8     strTgtFileNm = frmGetFile.lblFileNm.Caption
9     For Each ctrl In frmGetFile.Controls
10        If TypeOf ctrl Is MSForms.CheckBox Then
11            If ctrl.Value = True Then
12                MsgBox ctrl.Name & " & ctrl.tag"
13            End If
14        End If
15    Next ctrl
16    On Error GoTo ErrorHandler
17    strMdlNm = "Cleaning"          ' Modulname
18    If ProcExists(ActiveDocument, strMdlNm, strProcNm) = True Then
19        strFileNm = frmGetFile.lblFileNm      ' Dateiname aus Benutzerformular
20        If Len(strFileNm) > 0 And Dir(strFileNm) <> vbNullString Then
21            lngRtnCode = MsgBox("'" & Dir(strFileNm) & "' öffnen und bereinigen?",
22 vbQuestion + vbYesNo, "Word-Datei öffnen und bereinigen?")
23            If lngRtnCode = vbYes Then
24                Set docTgt = Documents.Open(FileName:=strFileNm)          , Zieldokument
25                docTgt.Activate
26                Application.Run Macroname:=strProcNm, varg1:=docTgt      , Prozedur ausführen
27            Else
28                MsgBox „Es wurde ‚Abbruch‘ wurde gewählt!“, vbOKOnly, „Word-Datei öffnen“
29            End If
30        Else
31            MsgBox „Zieldokument konnte nicht geöffnet werden!“, vbExclamation,
32 „GetWordFile“
33        End If
34    Else
35        MsgBox „Die ausgewählte VBA-Prozedur existiert nicht!“, vbExclamation, strProcNm
36    End If
37 ExitPoint:
38 On Error Resume Next
39 Exit Sub
40 ErrorHandler:
41 MsgBox „Laufzeitfehler # “ & Err.Number & “, “ & Err.Description, vbCritical, “cmdStart”
42 Resume ExitPoint
43 End Sub

```

5.3 Modul *Tools*

```
1 Sub OpenWordFile()  
2   ' Aufgabe: Word-Datei auswählen, öffnen und interaktiv bereinigen.  
3   Dim strFullNm As String   ' vollst. Dateiname d. ausgewählten Zieldokuments  
4   Dim lngRtnCode As Long   ' Rückgabewert d. MsgBox  
5   On Error GoTo ErrorHandler ' Fehlerroutine  
6   With Application.FileDialog(msoFileDialogOpen)  
7     .Title = "Dateiauswahl"  
8     .ButtonName = "Auswählen"  
9     .AllowMultiSelect = False  
10    .InitialFileName = ThisDocument.Path  
11    With .Filters  
12      .Clear  
13      .Add "Word-Dateien (*.docx)", "*.docx", 1  
14      .Add "Word-Dateien (*.docm)", "*.docm", 2  
15      .Add "Alle Dateien (*.*)", "*.*", 3  
16    End With  
17    .FilterIndex = 1  
18    If .Show = True Then  
19      strFullNm = CStr(.SelectedItems(1)) ' Pfadname + Dateiname  
20      If Len(strFullNm) > 0 And Dir(strFullNm) <> vbNullString Then  
21        With frmGetFile  
22          .lblFileNm = strFullNm  
23          .cmdStart.Enabled = True  
24        End With  
25      Else  
26        MsgBox "Zieldokument kann nicht geöffnet werden", vbExclamation, "OpenWordFile"  
27      End If  
28    Else  
29      MsgBox "Es wurde nichts ausgewählt!", vbOKOnly, "Dateiauswahl"  
30    End If  
31  End With  
32 ExitPoint:  
33 Exit Sub  
34 ErrorHandler:  
35   MsgBox "Laufzeitfehler: " & Err.Description, vbCritical, "OpenWordFile"  
36   Resume ExitPoint  
37 End Sub
```

```
1 Public Function ProcExists(docSrc As Document, strMdlNm As String, strProcNm As String) As
2 Boolean
3     ' Aufgabe: Existenzprüfung einer Prozedur.
4     ' Verweis erforderlich auf: 'Microsoft Visual Basic for Applications Extensibility 5.3'
5     Dim CodeMod As VBIDE.CodeModule, lngLine As Long
6     ProcExists = False
7     Err.Clear
8     On Error Resume Next
9     Set CodeMod = docSrc.VBProject.VBComponents(strMdlNm).CodeModule
10    If Err.Number = 0 Then
11        lngLine = CodeMod.ProcStartLine(strProcNm, vbext_pk_Proc)
12        If Err.Number = 0 Then ProcExists = True
13    End If
14 End Function
```