

Word-Dokument säubern (CleanUp_Document)

Dr. Volker Thormählen, 10. Nov. 2016

Inhaltsverzeichnis

Inhaltsverzeichnis.....	I
Abbildungsverzeichnis.....	II
Tabellenverzeichnis.....	II
Listings	II
1 Aufgabenstellung	1
2 Lösungsansatz	1
3 Projekt-Explorer	2
4 ThisDocument	2
5 Benutzerformular	3
6 Ereignisprozeduren	3
6.1 Zieldokument auswählen	4
6.2 Kopie des Zieldokuments erstellen.....	4
6.3 Kopie säubern.....	4
7 Standardprozeduren	5
8 Nachbearbeitung.....	6
9 Verfeinerungen	6
10 Anhang 1: Prozeduren	III
10.1 Ereignisprozeduren.....	III
10.2 Standardprozeduren.....	V
11 Anhang 2: Leerraum	X
12 Literaturverzeichnis.....	XI

Abbildungsverzeichnis

Abb. 1: Baumstruktur des Projekt-Explorers für das Projekt „CleanUp_Document“	2
Abb. 2: Benutzerformular (CleanUp_Form) mit den Namen der Steuerelemente	3

Tabellenverzeichnis

Tabelle 1: Steuerelements im Benutzerformular (CleanUp_Form)	3
Tabelle 2: Ereignisprozeduren des Benutzerformulars.....	4
Tabelle 3: Standardprozeduren zur Säuberung eines WORD-Dokuments.....	4
Tabelle 4: Alphabetische Liste der Standardprozeduren zur Säuberung eines WORD-Dokuments.....	5
Tabelle 5: Leerraum (engl. white space).....	X

Listings

Listing 1: Alternative Ereignisprozeduren zum Aufruf des Benutzerformulars.....	2
Listing 2: Prozedur zum Erstellen einer Befehlsschaltfläche in „ThisDocument“	2
Listing 3: Prozedur zum Aufruf des Benutzerformulars mit Namen „CleanUp_Form“	2
Listing 4: Text-Hervorhebungen zurücksetzen	6
Listing 5: Suchbereich begrenzen mit der Style-Eigenschaft des Find-Objekts.....	6
Listing 6: Ereignisprozeduren des Benutzerformulars	V
Listing 7: Leerzeichen am Anfang und/oder Ende eines Absatzes entfernen	VI
Listing 8: Leerzeichen erkennen	VII
Listing 9: Leere Absätze entfernen	VIII
Listing 10: Leerzeichen vor (nach) best. Satzzeichen löschen (einfügen)	X

1 Aufgabenstellung

In diesem Beitrag wird gezeigt, wie formale Fehler in einem bestehenden *Word*-Dokument durch den Einsatz von VBA¹-Prozeduren (sog. Makros) gefunden und berichtigt werden können, die gewöhnlich durch die in *Word* eingebaute Rechtschreibprüfung nicht gefunden werden. Denn was als formaler Fehler angesehen wird, richtet sich nach u. a. nach den persönlichen oder betrieblichen Standards für den jeweiligen Schreibstil.

Die vorgeschlagene Lösung zum Säubern eines vorhandenen *Word*-Dokuments beruht auf der Annahme, dass nach der üblichen Überprüfung der Rechtschreibung und Grammatik mit *WORD* weitere formale Überprüfungen erforderlich sind, um die oben erwähnten Standards einzuhalten.

Auch für die Bereinigung von Texten, die von Webseiten, PDF²-Dateien oder E-Mails in ein *Word*-Dokument eingefügt werden sollen, eignet sich die vorgeschlagene Lösung. Sie ist im Übrigen unverzichtbar, wenn ein Manuskript die formalen Regeln eines Verlags genau erfüllen muss, vgl. dazu [1].

2 Lösungsansatz

Um eine benutzerfreundliche Lösung zu erreichen, werden VBA-Prozeduren (sog. Makros³) eingesetzt, die mit einem Benutzerformular (engl. Userform) gestartet werden können. Das Benutzerformular kann alternativ wie folgt aufgerufen werden:

- Durch eine Schaltfläche mit der Beschriftung „*Formular*“ am Anfang dieses Dokuments
- Durch das Öffnen dieses Dokuments (*CleanUp_Document.docm*).

¹ VBA ist das Kürzel für **V**isual **B**asic for **A**pplications, eine Skriptsprache für die Steuerung von Abläufen der Microsoft-Office-Programmfamilie.

² Das Kürzel **PDF** steht für "**P**ortable **D**ocument **F**ormat".

³ Makros sind kleine Programme, welche mittels VBA erstellt bzw. mittels in *Word* integriertem Makro-Recorder aufgezeichnet werden können.

3 Projekt-Explorer

Die weitere Beschreibung des Lösungsansatzes folgt der in Abb. 1 dargestellten Baumstruktur für das Projekt mit dem Namen „CleanUp_Document“.

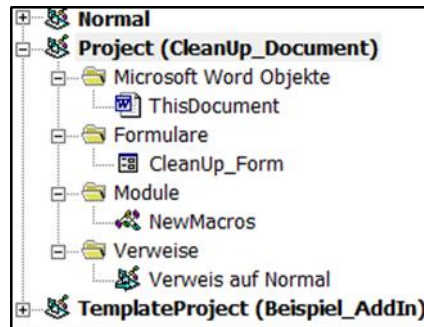


Abb. 1: Baumstruktur des Projekt-Explorers für das Projekt „CleanUp_Document“

4 ThisDocument

Aus technischer Sicht repräsentiert „ThisDocument“ ein Code-Modul, siehe Abb. 1. Listing 1 beinhaltet die dort gespeicherten Ereignisprozeduren zum Starten des Benutzerformulars mit dem Namen „CleanUp_Form“.

```
Private Sub Document_Open()
    Application.Run "AutoOpenMyForm"
End Sub
Private Sub CommandButton1_Click()
    Application.Run "AutoOpenMyForm"
End Sub
```

Listing 1: Alternative Ereignisprozeduren zum Aufruf des Benutzerformulars

Das in Listing 1 enthaltene Unterprogramm „CommandButton1_Click()“ zum Aufruf des erwähnten Benutzerformulars wurde mit der Prozedur „AddCommandButton()“ automatisch erzeugt, siehe Listing 2.

```
Sub AddCommandButton()
    ' VBA-Code zum Einfügen einer Befehlsschaltfläche mit der
    ' der Beschriftung "Formular" in "ThisDocument"
    Dim docSrc As Document ' aktuelles Dokument
    Dim objShp As InlineShape ' Objekt in der Textebene des aktuellen Dokuments
    Dim strCode As String ' VBA-Code für die die Befehlsschaltfläche
    Set docSrc = ThisDocument
    Set objShp = docSrc.Content.InlineShapes.AddOLEControl(ClassType:="Forms.CommandButton.1")
    objShp.OLEFormat.Object.Caption = "Formular"
    strCode = "Private Sub " & objShp.OLEFormat.Object.Name & "_Click()" & vbCrLf &
    " Application.Run ""AutoOpenMyForm"" & vbCrLf & "End Sub"
    docSrc.VBProject.VBComponents("ThisDocument").CodeModule.AddFromString strCode
End Sub
```

Listing 2: Prozedur zum Erstellen einer Befehlsschaltfläche in „ThisDocument“

Die Anweisungen `Application.Run „AutoOpenMyForm“`, siehe Listing 1, rufen jeweils die Prozedur `AutoOpenMyForm()` auf, siehe Listing 3. Damit wird jeweils das Benutzerformular mit dem Namen „CleanUp_Form“ angezeigt.

```
Sub AutoOpenMyForm()
    ' ActiveWindow.Visible = False
    CleanUp_Form.Show vbModeless
End Sub
```

Listing 3: Prozedur zum Aufruf des Benutzerformulars mit Namen „CleanUp_Form“

5 Benutzerformular

Das aufgerufene Benutzerformular mit dem Namen „*CleanUp_Form*“ enthält drei Bezeichnungsfelder, zwei Textfelder, drei Befehlsschaltflächen und ein Kontrollkästchen, siehe Tabelle 1.

Steuerelement		
Name	Art	Inhalt
lblTitle	Bezeichnungsfeld	Kopie des Zieldokuments säubern
lblTgtNm	Bezeichnungsfeld	Name des Zieldokuments
lblTgtCpy	Bezeichnungsfeld	Name der Kopie des Zieldokuments
txtTgtNm	Textfeld	Vollständiger Name des Zieldokuments
txtTgtCpy	Textfeld	Vollständiger Name der Kopie des Zieldokuments
cmdGetFile	Befehlsschaltfläche	Zieldokument auswählen
cmdCleanUp	Befehlsschaltfläche	Kopie säubern
cmdCancel	Befehlsschaltfläche	Abbruch/Ende
chkNonStop	Kontrollkästchen	Säubern ohne Benutzerdialog

Tabelle 1: Steuerelements im Benutzerformular (CleanUp_Form)

Die Namen der neun Steuerelemente sind auch in der Grafik (sog. ScreenShot) des Benutzerformulars verzeichnet, siehe Abb. 2. Gezeigt wird dort das Aussehen des Benutzerformulars unmittelbar nach dessen Aufruf. Der Benutzer hat nur die Wahl zwischen zwei Befehlsschaltflächen mit den Beschriftungen „Zieldokument wählen“ bzw. „Abbruch/Ende“

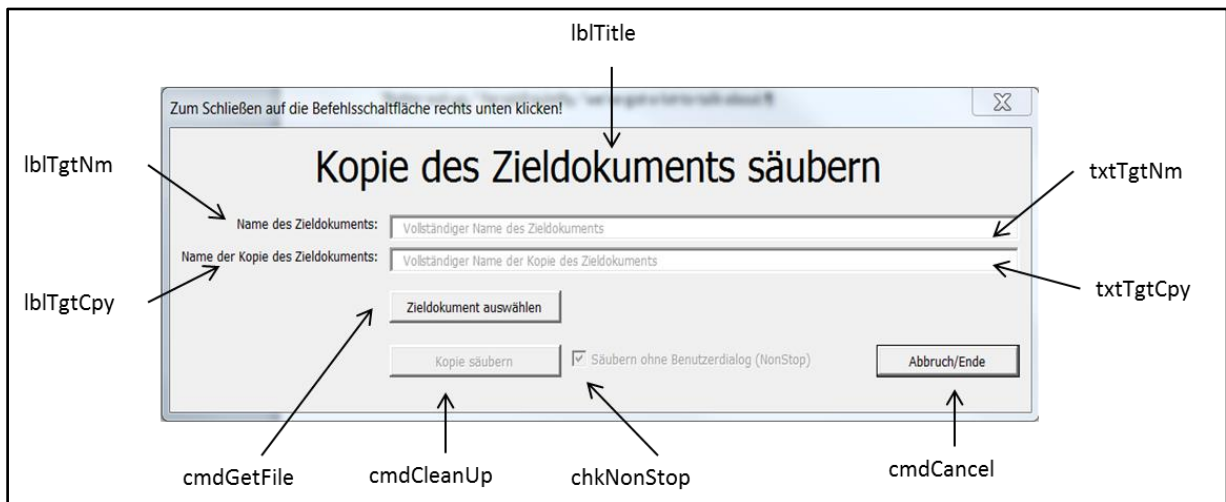


Abb. 2: Benutzerformular (CleanUp_Form) mit den Namen der Steuerelemente

6 Ereignisprozeduren

Zum Benutzerformular (*CleanUp_Form*) gehören 6 Ereignisprozeduren, siehe Tabelle 2.

- Die ersten drei werden automatisch von Word ausgelöst.
- Die letzten drei werden durch das Klicken auf die entsprechende Schaltfläche gestartet.

Private Sub ...	Aufgabe
UserForm_Initialize()	Anfangswerte für das Benutzerformular setzen
UserForm_Activate()	Hinweistext im Formulkopf anzeigen, siehe Abb. 2
UserForm_QueryClose(Cancel As	Das ungeplante Schließen des Benutzerformulars mit dem roten

Integer, CloseMode As Integer)	Kreuz rechts oben verhindern
cmdGetFile_Click()	Zieldokument bestimmen und eine Kopie davon erstellen
cmdCleanup_Click()	Kopie des gewählten Zieldokuments säubern
cmdCancel_Click()	Benutzerformular schließen

Tabelle 2: Ereignisprozeduren des Benutzerformulars

Auf die Schaltfläche mit dem Namen *cmdCleanup*, mit der die Ereignisprozedur *cmdCleanup_Click()* ausgelöst wird, kann erst dann wirksam geklickt werden, wenn eine Kopie des gewählten Zieldokuments erfolgreich erstellt wurde.

6.1 Zieldokument auswählen

Zum Auswählen des Zieldokuments mit der Ereignisprozedur *cmdGetFile_Click()* wird der in Word eingebaute Dialog zum Öffnen von Dokumenten eingesetzt.

6.2 Kopie des Zieldokuments erstellen

Die Kopie des Zieldokuments wird mit dem in Word eingebauten Speichern-Dialog erstellt. Der Dateiname der Kopie wird gebildet aus dem Namen des Zieldokuments ergänzt durch dessen Erstellungsdatum mit dem Format *yyyy_mm_dd*. Das Erstellen einer Kopie ist wichtig, damit das jeweilige Zieldokument nicht durch Bedienungs- und/oder Programmfehler unbrauchbar wird.

Nach dem Erstellen der Kopie wird das gewählte Zieldokument unverändert geschlossen.

6.3 Kopie säubern

Die Schaltfläche mit der Beschriftung „Kopie säubern“, siehe Abb. 2, ruft bestimmte Standardprozeduren zum Säubern der Kopie des Zieldokuments auf. Zurzeit sind folgende 14 Prozeduren zum Säubern der Kopie des Zieldokuments eingerichtet:

Name der Standardprozedur	Aufgabe
RemoveLeadingAndTrailingSpaces	Leerzeichen (sog. White Spaces) am Anfang und/oder Ende eines Absatzes entfernen
DeleteEmptyParas	Leere Absätze entfernen
FindAndReplaceStraightQuotes	Doppelte/halbe gerade Anführungszeichen durch typografische ersetzen
CheckNumbersAndUnitsOfMeasurement	Geschütztes Leerzeichen zwischen Zahl und zugehöriger Maßeinheit einfügen
EnsureSpaceBeforeNumber	Geschütztes Leerzeichen vor einer Zahl durch ein übliches ersetzen.
FindSpaceInsideNumber	Leerzeichen in einer Nummer durch ein geschütztes ersetzen
CleanDocument	Suchen/ersetzen mit/ohne Mustervergleich
ProperPunctuation	Leerzeichen vor (nach) best. Satzzeichen löschen (einfügen)
ProperDocument	5 formale Prüfungen/Säuberungen durchführen
ProperAbbreviations	Abkürzungen finden und ggf. ersetzen
CapAfterColons	Großschreibung nach Doppelpunkt erzwingen
CleanReturnsInNotes	Leere Absätze in Fußnoten entfernen
InsertMissingDots	Punkt nach Fußnote einfügen, falls notwendig
RemoveTrailingSpacesInCells	Nachfolgende Leerzeichen in Tabellen-Zellen entfernen

Tabelle 3: Standardprozeduren zur Säuberung eines WORD-Dokuments

7 Standardprozeduren

Tabelle 4 enthält nochmals die Namen der 14 Standardprozeduren, allerdings in alphabetischer Reihenfolge:

Name der Standardprozedur	Reihenfolge	LoC	Interaktiv	Element
CapAfterColons	11	48	ja	
CheckNumbersAndUnitsOfMeasurement	4	45	ja	
CleanDocument	7	72	ja	
CleanReturnsInNotes	12	73	ja	Fußnoten
DeleteEmptyParas	2	59	nein	
EnsureSpaceBeforeNumber	5	51	ja	
FindAndReplaceStraightQuotes	3	70	nein	
FindSpaceInsideNumber	6	46	ja	
InsertMissingDots	13	64	ja	Fußnoten
ProperAbbreviations	10	117	ja	
ProperDocument	9	46	ja	
ProperPunctuation	8	93	ja	
RemoveLeadingAndTrailingSpaces	1	82	ja	
RemoveTrailingSpacesInCells	14	35	nein	Tabellen
Legende: <i>Reihenfolge:</i> Logische Reihenfolge der Bereinigungs-schritte <i>LoC:</i> Zahl der Programmzeilen in der auf Standardprozedur <i>Interaktiv:</i> Ja-/Nein-Entscheidung des Benutzers bei Berichtigungsvorschlägen <i>Element:</i> Objekte in MS WORD, die nur bei Existenz geprüft bzw. berichtigt werden.				

Tabelle 4: Alphabetische Liste der Standardprozeduren zur Säuberung eines WORD-Dokuments

Diese Prozeduren und weitere befinden sich im Modul „NewMaros“ des Projekts mit dem Namen „CleanUp_Document“, siehe Abb. 1. In Listing 6 sind die entsprechenden Aufrufe grau markiert.

Einige der in Tabelle 4 genannten Prozeduren ignorieren den Inhalt des Kontrollkästchen „Säubern ohne Benutzerdialog“ (siehe Abb. 1), mit anderen Worten, sie laufen ohne Unterbrechung ab. Auch werden mögliche Änderungen in der Kopie des Zieldokuments nicht farblich hervorgehoben.

Die übrigen Prozeduren reagieren auf den Inhalt des Kontrollkästchens, d. h. der Benutzer kann vorgeschlagene Änderungen ablehnen oder akzeptieren. Bei Akzeptanz werden sie in der Kopie des Zieldokuments vorübergehend farblich hervorgehoben, sonst nicht.

Aufgrund der strukturierten und modularen Programmierung können neue Prozeduren zur Säuberung der Kopie des Zieldokuments nach dem Muster der vorhandenen geradewegs hinzugefügt werden. Dabei muss allerdings auf die logische Reihenfolge der Aufrufe der Prozeduren im Benutzerformular *CleanUp_Form* geachtet werden.

8 Nachbearbeitung

Nach der Säuberung kann die bereits automatisch geschlossene Zieldatei gelöscht und deren gesäuberte Kopie beliebig umbenannt werden.

Das Hervorheben der Änderungen in der gesäuberten Kopie muss rückgängig gemacht werden. Dafür kann folgende Standardprozedur eingesetzt werden:

```
Sub ResetHighlights()
' Aufgabe: Hervorhebungen im Text und in Fußnoten zurücksetzen
Dim rngTemp As Range
Set rngTemp = ActiveDocument.Range(Start:=0, End:=0)
With rngTemp.Find
.ClearFormatting
.Highlight = True
With .Replacement
.ClearFormatting
.Highlight = False
End With
.Execute Replace:=wdReplaceAll, Forward:=True, FindText:="", ReplaceWith:"", Format:=True
End With
ActiveDocument.StoryRanges(wdFootnotesStory).HighlightColorIndex = wdNoHighlight
Set rngTemp = Nothing
End Sub
```

Listing 4: Text-Hervorhebungen zurücksetzen

9 Verfeinerungen

Die beschriebenen Lösungen zum Säubern eines Word-Dokuments können in mehrfacher Hinsicht verfeinert werden:

- Der Suchbereich für die eingebaute Word-Funktionalität „Suchen und Ersetzen“ kann auf bestimmte Abschnitte (engl. *sections*) eines Word-Dokuments begrenzt werden, beispielsweise durch folgende Code-Zeile:
With ActiveDocument.Sections(3).Range.Find
Dadurch wird der Suchbereich auf den dritten Abschnitt eines Word-Dokuments eingegrenzt.
- Der Suchbereich in einem Word-Dokument kann auch durch sog. *StoryRanges* zu begrenzt werden, beispielsweise durch folgende Code-Zeile:
With ActiveDocument.StoryRanges(wdMainTextStory).Find
Dadurch wird das Suchen und Ersetzen auf den Hauptdokumentbereich *wdMainTextStory* begrenzt. In Word 2010 stehen 17 *StoryRanges* zur Auswahl.
- Mit der *Selection.Range*-Eigenschaft kann ein Suchbereich wie folgt eingegrenzt werden:
With Selection.Range.Find
- Mit der *Find.Style*-Eigenschaft kann der Suchbereich auf eine bestimmte Formatvorlage begrenzt werden, siehe dazu das Code-Beispiel in Listing 5. In Word 2010 gibt es eine Vielzahl eingebauter Formatvorlagen als auch beliebig viele benutzerdefinierte.

```
Sub FindWithStyleProperty()
' Aufgabe: 2 und mehr Leerzeichen löschen in allen Textbereichen,
' die mit der Formatvorlage "Standard" formatiert sind.
Dim objRng As Range
Set objRng = ActiveDocument.Content
With objRng.Find
.Text = " {2;}"
.Replacement.Text = " "
.MatchWildcards = True
.Style = "Standard"
End With
objRng.Find.Execute Replace:=wdReplaceAll
End Sub
```

Listing 5: Suchbereich begrenzen mit der Style-Eigenschaft des Find-Objekts

10 Anhang 1: Prozeduren

10.1 Ereignisprozeduren

```

Private Sub UserForm_Initialize()
' Aufgabe: Anfangswerte für das Benutzerformular setzen
With Me
.txtTgtNm = "Vollständiger Name des Zieldokuments"
.txtTgtNm.Enabled = False
.txtTgtCpy = "Vollständiger Name der Kopie des Zieldokuments"
.txtTgtCpy.Enabled = False
.cmdCleanUp.Enabled = False
.chkNonStop = True
.chkNonStop.Enabled = False
End With
End Sub
Private Sub UserForm_Activate()
' Hinweistext im Formulkopf anzeigen.
CleanUp_Form.Caption = "Zum Schließen auf die Befehlsschaltfläche rechts unten klicken!"
End Sub
Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
' Das Schließen des Benutzerformulars mit dem roten Kreuz verhindern.
If CloseMode = vbFormControlMenu Then Cancel = True
End Sub
Private Sub cmdGetFile_Click()
' Aufgabe: Zieldokument bestimmen und eine Kopie davon erstellen.
Dim docTgt As Document ' Zieldokument
Dim docTgtCpy As Document ' Kopie des Zieldokuments
Dim strTgtCpyNm As String ' Name der Kopie des Zieldokuments
' Zieldokument mit eingebautem Öffnen-Dialog auswählen
If Not GetDocTgt Then
' Wenn der Öffnen-Dialog abgebrochen wird, dann Ende.
End
End If
' Den Namen des gewählten Zieldokuments in das Benutzerformular eintragen.
Set docTgt = ActiveDocument
docTgt.Activate
Me.txtTgtNm = docTgt.FullName
' Eine Kopie des gewählten Zieldokuments erstellen.
docTgt.Select
Selection.Copy
Documents.Add Template:="Normal", NewTemplate:=False, _
DocumentType:=wdNewBlankDocument, Visible:=False
Selection.PasteAndFormat (wdFormatOriginalFormatting)
Set docTgtCpy = ActiveDocument
' Den Namen der Kopie des gewählten Zieldokuments zusammenbauen.
strTgtCpyNm = MakeDocNewNm(docTgtCpy)
' Falls bereits eine Kopie des Zieldokuments existiert, diese Kopie löschen.
If DocExists(strTgtCpyNm) Then
Kill strTgtCpyNm
End If
' Kopie speichern mit dem eingebauten Speichern-Dialog.
With Dialogs(wdDialogFileSaveAs)
.Name = strTgtCpyNm
If .Show = False Then
MsgBox "Speichern der Kopie abgebrochen", vbInformation
GoTo Exit_Point
End If
End With
' Den Namen der Kopie des Zieldokuments in das Benutzerformular eintragen.
Me.txtTgtCpy = ActiveDocument.FullName
Me.txtTgtCpy.Enabled = False

' Zieldokument schließen ohne zu speichern.
docTgt.Close SaveChanges:=wdDoNotSaveChanges

' Diese Befehlsschaltfläche stilllegen ...
Me.cmdGetFile.Enabled = False
' und folgende zwei Befehlsschaltflächen zulassen:
Me.cmdCleanUp.Enabled = True
Me.chkNonStop.Enabled = True
Exit_Point:
On Error Resume Next
Set docTgt = Nothing
Set docTgtCpy = Nothing
Exit Sub
Err_Point:
MsgBox "Laufzeitfehler: " & Err.Description, vbCritical
Resume Exit_Point
End Sub

```

```

Private Sub cmdCleanup_Click()
' Aufgabe: Kopie des gewählten Zieldokuments säubern
Dim bolErr As Boolean ' Fehlerschalter
Dim docTgtCpy As Document ' Kopie des gewählten Zieldokuments
Dim bolShowRev As Boolean ' Kommentare ein-/ausblenden
Dim bolHiddenTxt As Boolean ' Versteckten Text ein-/ausblenden
bolErr = False
On Error GoTo Err_Point
' Falls die Kopie des gewählten Zieldokuments existiert, diese öffnen, sonst abbrechen.
If DocExists(Me.txtTgtCpy.Text) Then
Documents.Open FileName:=Me.txtTgtCpy.Text, Format:=wdOpenFormatAuto
Set docTgtCpy = ActiveDocument
docTgtCpy.Activate
With docTgtCpy
' Änderungen im aktiven Dokument nicht nachverfolgen/anzeigen
.TrackRevisions = False
.ShowRevisions = False
If .ProtectionType <> wdNoProtection Then
' Dokumentschutz aufheben
.Unprotect Password:=""
End If
With .ActiveWindow.View
' Kommentare im aktiven Dokument ausgeblenden
bolShowRev = .ShowRevisionsAndComments
.ShowRevisionsAndComments = False
' Versteckten Text im Aktiven Document ein-/ausblenden
bolShowRev = .ShowHiddenText
.ShowHiddenText = False
End With
End With
' <<< Kopie (docTgtCpy) des Zieldokuments säubern >>>
' Leerzeichen am Anfang und/oder Ende eines Absatzes entfernen
Call RemoveLeadingAndTrailingSpaces(docTgtCpy, Me.chkNonStop.Value)
' Leere Absätze entfernen
Call DeleteEmptyParas(docTgtCpy)
' Doppelte/halbe gerade Anführungszeichen durch typografische ersetzen
Call FindAndReplaceStraightQuotes(docTgtCpy)
' Geschütztes Leerzeichen zwischen Zahl und zugehöriger Maßeinheit einfügen
Call CheckNumbersAndUnitsOfMeasurement(docTgtCpy, Me.chkNonStop.Value)
' Geschütztes Leerzeichen vor einer Zahl durch ein übliches ersetzen.
Call EnsureSpaceBeforeNumber(docTgtCpy, Me.chkNonStop.Value)
' Zwischenraum in einer Nummer durch ein geschütztes ersetzen
Call FindSpaceInsideNumber(docTgtCpy, Me.chkNonStop.Value)
' Suchen und Ersetzen mit/ohne Mustervergleich (engl. regular expression)
Call CleanDocument(docTgtCpy, Me.chkNonStop.Value)
' Leerzeichen vor/nach Satzzeichen prüfen
Call ProperPunctuation(docTgtCpy, Me.chkNonStop.Value)
' Fünf formale Prüfungen/Säuberungen durchführen
Call ProperDocument(docTgtCpy, Me.chkNonStop.Value)
' Abkürzungen finden und ggf. ersetzen
Call ProperAbbreviations(docTgtCpy, Me.chkNonStop.Value)
' Großschreibung nach Doppelpunkt erzwingen
Call CapAfterColons(docTgtCpy, Me.chkNonStop.Value)
If docTgtCpy.Footnotes.Count > 0 Then
' Leere Absätze in Fußnoten entfernen
Call CleanReturnsInNotes(docTgtCpy, Me.chkNonStop.Value)
' Punkt nach Fußnote einfügen
Call InsertMissingDots(docTgtCpy, Me.chkNonStop.Value)
End If
If docTgtCpy.Tables.Count > 0 Then
' Nachfolgende Leerzeichen in Tabellen-Zellen entfernen
Call RemoveTrailingSpacesInCells(docTgtCpy)
End If
docTgtCpy.Close SaveChanges:=wdSaveChanges ' Änderungen speichern.
Else
MsgBox "Der Name der Kopie des Zieldokuments ist ungültig!", vbCritical
bolErr = True
GoTo Exit_Point
End If
' Folgende zwei Steuerelemente stilllegen:
Me.cmdCleanUp.Enabled = False
Me.chkNonStop.Enabled = False
Exit_Point:
On Error Resume Next
With docTgtCpy.ActiveWindow.View
.ShowRevisionsAndComments = bolShowRev ' Kommentare ein-/ausblenden.
.ShowHiddenText = bolHiddenTxt ' versteckten Text ein-/ausblenden.
End With
Set docTgtCpy = Nothing
If bolErr Then
End
Else
Exit Sub
End If
Err_Point:
MsgBox "Laufzeitfehler: " & Err.Number & ", " & _
Err.Description & ", " & Err.Source, vbCritical, "cmdCleanup_Click()"
Resume Exit_Point
End Sub

```

```

Private Sub cmdCancel_Click()
' Aufgabe: Benutzerformular schließen
If Documents.Count > 1 Then
Documents.Close SaveChanges:=wdDoNotSaveChanges
Application.Quit ' Word-Anwendung verlassen
End If.
Unload Me ' Benutzerformular schließen.
End Sub

```

Listing 6: Ereignisprozeduren des Benutzerformulars

10.2 Standardprozeduren

Nur der VBA-Quellcode der drei wichtigsten Standardprozeduren wird hier gezeigt:

- **RemoveLeadingAndTrailingSpaces** Leerzeichen am Anfang und/oder Ende aller Absätze entfernen
- **DeleteEmptyParas** Leere Absätze entfernen
- **ProperPunctuation** Leerzeichen vor/nach best. Satzzeichen löschen/einfügen

```

Sub RemoveLeadingAndTrailingSpaces(ByVal docTgtCpy As Document, Optional bolNonStop As Boolean = False)
' Aufgabe: Leerzeichen am Anfang und/oder Ende eines Absatzes entfernen
' Argumente: docTgtCpy: aktuelles Word-Dokument
' bolNonStop: Schalter für 'Ohne Unterbrechnung' (J/N)
' Benötigt: Funktion "IsWhiteSpace"
Dim objPara As Paragraph ' Absatz im Dokument
Dim lngAsk As Long ' Antwort auf eine Abfrage
Dim lngSpaceCnt As Long ' Zähler für Leerzeichen am Anfang/Ende eines Absatzes
Dim lngTotalDel As Long ' Insgesamt entfernte Leerzeichen
Dim rngChar As Range ' Suchbereich
Dim rngSpaceDel As Range ' Löchbereich
On Error GoTo Err_Point
docTgtCpy.Activate
With Selection
.HomeKey Unit:=wdStory, Extend:=wdMove
Set objPara = .Range.Paragraphs(1) ' erster Absatz im aktuellen Dokument
End With
Do
objPara.Range.Select
' Absätze in Tabellen ignorieren
If Not Selection.Information(wdWithInTable) Then
' Leerzeichen am Anfang eines Absatzes entfernen, falls vorhanden.
lngSpaceCnt = 0
Set rngChar = objPara.Range.Characters.First
If rngChar.Text <> vbCr Then
Do While IsWhiteSpace(rngChar.Text)
lngSpaceCnt = lngSpaceCnt + 1
Set rngChar = rngChar.Next
If rngChar Is Nothing Or rngChar.Text = vbCr Then Exit Do
Loop
End If
If lngSpaceCnt > 0 Then
' Leerzeichen am Absatzanfang markieren
Set rngSpaceDel = ActiveDocument.Range(Start:=objPara.Range.Start, _
End:=objPara.Range.Start + lngSpaceCnt)
rngSpaceDel.Select
rngSpaceDel.HighlightColorIndex = wdTurquoise
If bolNonStop Then
lngAsk = vbYes
Else
lngAsk = MsgBox("Diese" & IIf(lngSpaceCnt < 2, "s", "") & Str(lngSpaceCnt) & _
" Leerzeichen entfernen?", vbYesNoCancel + vbQuestion, "RemoveLeadingAndTrailingSpaces")
End If
If lngAsk = vbYes Then
rngSpaceDel.Delete
lngTotalDel = lngTotalDel + lngSpaceCnt
ElseIf lngAsk = vbCancel Then
GoTo Exit_Point
End If
End If
' Leerzeichen am Ende eines Absatzes entfernen, falls vorhanden.
If objPara.Range.Characters.Count > 1 Then
' Gehe vor die Absatzmarke
Set rngChar = objPara.Range.Characters.Last.Previous
Else
Set rngChar = Nothing
End If
lngSpaceCnt = 0
If Not rngChar Is Nothing Then
Do While IsWhiteSpace(rngChar.Text)
lngSpaceCnt = lngSpaceCnt + 1
Set rngChar = rngChar.Previous

```

```

    Loop
End If
If lngSpaceCnt > 0 Then
    ' Leerzeichen am Absatzende markieren
    Set rngSpaceDel = ActiveDocument.Range(Start:=objPara.Range.End - lngSpaceCnt - 1, _
        End:=objPara.Range.End - 1)
    rngSpaceDel.Select
    rngSpaceDel.HighlightColorIndex = wdTurquoise
    If bolNonStop Then
        lngAsk = vbYes
    Else
        lngAsk = MsgBox("Diese " & IIf(lngSpaceCnt < 2, "s", "") & Str(lngSpaceCnt) & _
            " Leerzeichen entfernen?", vbYesNoCancel + vbQuestion, "RemoveLeadingAndTrailingSpaces")
    End If
    If lngAsk = vbYes Then
        rngSpaceDel.Delete
        ' Gehe vor die Absatzmarke
        Set rngSpaceDel = objPara.Range.Characters.Last.Previous
        rngSpaceDel.Select
        If rngSpaceDel.Text = Space(1) Then
            rngSpaceDel.Delete
            rngSpaceDel.Delete ' 2-mal wg. Bug in Word
        End If
        lngTotalDel = lngTotalDel + lngSpaceCnt
    ElseIf lngAsk = vbCancel Then
        GoTo Exit_Point
    End If
End If
End If
End If
' Nächster Absatz
Set objPara = objPara.Next
Loop Until objPara Is Nothing
MsgBox Str(lngTotalDel) & " Leerzeichen am Anfang und/oder Ende eines Absatzes " & _
    IIf(bolNonStop, "automatisch", "interaktiv") & " entfernt.", _
    vbInformation, "RemoveLeadingAndTrailingSpaces"
Exit_Point:
On Error Resume Next
Set objPara = Nothing
Exit Sub
Err_Point:
MsgBox "Laufzeitfehler: " & Err.Number & ", " & vbCrLf & _
    Err.Description & ", " & Err.Source, vbCritical, "RemoveLeadingAndTrailingSpaces"
Resume Exit_Point
End Sub

```

Listing 7: Leerzeichen am Anfang und/oder Ende eines Absatzes entfernen

```

Function IsWhiteSpace(ByVal strChar As String) As Boolean
' Aufgabe: Gibt 'True' zurück, wenn das übergebene Zeichen (strTmp) ein Leerzeichen ist, sonst 'False'
' Wird aufgerufen von: RemoveLeadingAndTrailingSpaces
' Quelle:
'   o. V., Whitespace character
'   URL: https://en.wikipedia.org/wiki/Whitespace_character#Unicode
'-----
' Die 24 Hexadezimalzahlen in der unten stehenden Select Case Anweisung bedeuten:
' Horizontal Tab, HT, vbKeyTab
' LineFeed, LF
' Vertical Tab, VT
' Form Feed, FF, vkKeyClear
' Carriage Return, CR, vbKeyReturn
' Space, vbKeySpace
' Non-Breaking Space, Hard Space
' OGHAM Space Mark
' En Quad
' Em Quad (Mutton Quad)
' En Space (Nut)
' Em Space (Mutton)
' Tree-Per-Em-Space (Thick Space)
' Four-Per-Em-Space (Mid Space)
' Six-Per-Em-Space
' Figure Space
' Punctuation Space
' Thin Space
' Hair Space
' Zero-Width Space
' Narrow No-Break Space
' Medium Mathematical Space
' Ideographic Space
' Zero-Width Non-Breaking Space
'-----
Select Case AscW(strChar)
Case &H9, &HA, &HB, &HC, &HD, &H20, &HA0, &H1680, _
    &H2000, &H2001, &H2002, &H2003, &H2004, &H2005, &H2006, &H2007, &H2008, &H2009, _
    &H200A, &H200B, &H202F, &H205F, &H3000, &HFEFF
    IsWhiteSpace = True
Case Else
    IsWhiteSpace = False
End Select
End Function

```

Listing 8: Leerzeichen erkennen

```

Sub DeleteEmptyParas(ByRef docTgtCpy As Document)
' Aufgabe: Alle leeren Absätze in einem Word-Dokument löschen.
' Argumente: docTgtCpy - aktuelles Word-Dokument
' In Anlehnung an: [2]
Dim objRng As Range, objTbl As Table, lngAsk As Long
Dim strListSep As String ' Listentrennzeichen
On Error GoTo Err_Point
strListSep = Application.International(wdListSeparator) ' Listentrennzeichen ermitteln
' Leere Absätze löschen
With Selection
.HomeKey Unit:=wdStory, Extend:=wdMove
With .Find
.Text = "^13{2} & strListSep & ")"
.Replacement.Text = "^p"
.Forward = True
.Wrap = wdFindContinue
.Format = False
.MatchCase = False
.MatchWholeWord = False
.MatchAllWordForms = False
.MatchSoundsLike = False
.MatchWildcards = True
.Execute Replace:=wdReplaceAll
End With
End With
Set objRng = docTgtCpy.Paragraphs(1).Range ' Erster Absatz im Dokument
If objRng.Text = vbCr Then objRng.Delete
Set objRng = docTgtCpy.Paragraphs.Last.Range ' Letzter Absatz im Dokument
If objRng.Text = vbCr Then objRng.Delete
For Each objTbl In docTgtCpy.Tables
' Leeren Absatz NACH einer Tabelle löschen
Set objRng = objTbl.Range
objRng.Collapse wdCollapseEnd
With objRng.Paragraphs(1).Range
.Select
If Not Selection.Information(wdWithInTable) Then
.HighlightColorIndex = wdTurquoise
If .Text = vbCr Then
lngAsk = MsgBox("Leeren Absatz NACH Tabelle entfernen?", _
vbYesNoCancel + vbQuestion, "Leerer Absatz")
If lngAsk = vbYes Then
.Delete
ElseIf lngAsk = vbCancel Then
GoTo Exit_Point
End If
End If
End If
' Leeren Absatz VOR einer Tabelle löschen
Set objRng = objTbl.Range
objRng.Collapse wdCollapseStart
objRng.Move Unit:=wdParagraph, Count:=-1
With objRng.Paragraphs(1).Range
.Select
If Not Selection.Information(wdWithInTable) Then
.HighlightColorIndex = wdTurquoise
If .Text = vbCr Then
lngAsk = MsgBox("Leeren Absatz VOR Tabelle entfernen?", _
vbYesNoCancel + vbQuestion, "Leerer Absatz")
If lngAsk = vbYes Then
.Delete
ElseIf lngAsk = vbCancel Then
GoTo Exit_Point
End If
End If
End If
End With
Next objTbl
MsgBox "Das Entfernen leerer Absätze im Word-Dokument ist beendet.", _
vbInformation, "DeleteEmptyParas"
Exit_Point:
On Error Resume Next
Exit Sub
Err_Point:
Select Case Err.Number
Case 5904 ' Bereich kann nicht bearbeitet werden.
Resume Next
Case Else
MsgBox "Laufzeitfehler: " & Err.Number & ", " & vbCrLf & _
Err.Description & ", " & Err.Source, vbCritical, "DeleteEmptyParas"
Resume Exit_Point
End Select
End Sub

```

Listing 9: Leere Absätze entfernen

```

Sub ProperPunctuation(ByRef docTgtCpy As Document, ByVal bolNonStop As Boolean)
' Aufgabe: Leerzeichen vor/nach best. Satzzeichen löschen/einfügen.
' Argumente: docTgtCpy - aktuelles Word-Dokument
' bolNonStop - Schalter für 'Ohne Unterbrechung' (J/N)
' In Anlehnung an: [3]
Dim intUpdCnt As Integer ' Zähler f. Änderungen
Dim lngAsk As Long ' Antwort auf Abfrage
Dim lngColorIdx As Long ' Hervorhebungsfarbe f. Änderungen
Dim rngDoc As Range ' Suchbereich
Dim strFindTxt As String ' Suchbegriff
Dim strReplaceTxt As String ' Ersatztext
On Error GoTo Err_Point
lngColorIdx = Options.DefaultHighlightColorIndex
Options.DefaultHighlightColorIndex = wdTurquoise
' Vor Satzzeichen ist kein Leerzeichen zulässig!
strFindTxt = "[ ][,;.:!\|?]"
strReplaceTxt = vbNullString
Set rngDoc = docTgtCpy.Range
With rngDoc.Find
.ClearFormatting
.Replacement.ClearFormatting
.Style = wdStyleNormal ' nur bei Formatvorlage "Standard"
.Replacement.Highlight = True ' Ersetzung hervorheben
.Font.Underline = wdUnderlineNone ' Hyperlinks (unterstrichene Textstellen) ausnehmen
Do While .Execute(FindText:=strFindTxt, _
MatchCase:=True, _
MatchWholeWord:=True, _
MatchWildcards:=True, _
Forward:=True, _
Wrap:=wdFindStop) = True

rngDoc.Select
If bolNonStop Then
lngAsk = vbYes
Else
lngAsk = MsgBox("Leerzeichen vor einem Satzzeichen gefunden.", _
vbYesNo, "Entfernen?")
End If
If lngAsk = vbYes Then
With rngDoc
.Text = LTrim(.Text) ' Leerzeichen davor entfernen
End With
intUpdCnt = intUpdCnt + 1
End If
rngDoc.Collapse Direction:=wdCollapseEnd
Loop
End With
' Hinter Satzzeichen ist 1 Leerzeichen erforderlich!
strFindTxt = "[,;.:!\|?][! " & ChrW(34) & ChrW(13) & "]"
strReplaceTxt = Space(1)
Set rngDoc = ActiveDocument.Range
With rngDoc.Find
.ClearFormatting
.Replacement.ClearFormatting
.Style = wdStyleNormal ' nur bei Formatvorlage "Standard"
.Replacement.Highlight = True ' Ersetzung hervorheben
.Font.Underline = wdUnderlineNone ' Hyperlinks (unterstrichene Textstellen) ausnehmen
Do While .Execute(FindText:=strFindTxt, _
MatchCase:=True, _
MatchWholeWord:=True, _
MatchWildcards:=True, _
Forward:=True, _
Wrap:=wdFindStop) = True

rngDoc.Select
If bolNonStop Then
lngAsk = vbYes
Else
lngAsk = MsgBox("Kein Leerzeichen hinter einem Satzzeichen gefunden.", _
vbYesNo, "Einfügen?")
End If
If lngAsk = vbYes Then
With Selection
.Collapse Direction:=wdCollapseStart
.MoveRight Unit:=wdCharacter, Count:=1
.InsertAfter Text:=strReplaceTxt ' Leerzeichen dahinter einfügen
End With
intUpdCnt = intUpdCnt + 1
End If
rngDoc.Collapse Direction:=wdCollapseEnd
Loop
End With
MsgBox Str(intUpdCnt) & " Ersetzungen vor/hinter Satzzeichen " & _
IIf(bolNonStop, "automatisch", "interaktiv") & " durchgeführt.", _
vbInformation, "ProperPunctuation"
Exit Point:

```

```

On Error Resume Next
Set rngDoc = Nothing
Options.DefaultHighlightColorIndex = lngColorIdx
Exit Sub
Err_Point:
MsgBox "Laufzeitfehler: " & Err.Number & ", " & vbCrLf & _
    Err.Description & ", " & Err.Source, vbCritical, "ProperPunctuation"
Resume Exit_Point
End Sub

```

Listing 10: Leerzeichen vor/nach best. Satzzeichen löschen/einfügen

Der Quellcode der übrigen 14 Standardprozeduren zur Bereinigung eines WORD-Dokuments wird hier *nicht* aufgelistet. Stattdessen wird verwiesen auf das umfangreiche Quellenverzeichnis am Ende dieses Beitrags.

11 Anhang 2: Leerraum

Zum sog. Leerraum werden 25 Zeichen gezählt, die mit der Eigenschaft „white space“ gekennzeichnet sind, siehe Tabelle 5:

Leerraum (engl.: white space characters)			
Hex	Dezimal	engl. Bezeichnung	Gruppe
&H9	ChrW(9)	Horizontal Tab, <i>HT</i>	Steuerzeichen
&HA	ChrW(10)	LineFeed, LF	
&HB	ChrW(11)	Vertical Tab, <i>VT</i>	
&HC	ChrW(12)	Form Feed, <i>FF</i>	
&HD	ChrW(13)	Carriage Return, <i>CR</i>	
&H85	ChrW(133)	Next Line, <i>NEL</i>	
&H20	ChrW(32)	Space	normales Leerzeichen
&HA0	ChrW(160)	No Break Space (<i>Hard Space</i>)	geschütztes Leerzeichen
&H1680	ChrW(5760)	OGHAM Space Mark	bes. Leerzeichen
&H2000	ChrW(8192)	En Quad	schmale Leerzeichen
&H2001	ChrW(8193)	Em Quad (<i>Mutton Quad</i>)	
&H2002	ChrW(8194)	En Space (<i>Nut</i>)	
&H2003	ChrW(8195)	Em Space (<i>Mutton</i>)	
&H2004	ChrW(8196)	Tree-Per-Em-Space (<i>Thick Space</i>)	
&H2005	ChrW(8197)	Four-Per-Em-Space (<i>Mid Space</i>)	
&H2006	ChrW(8198)	Six-Per-Em-Space	
&H2007	ChrW(8199)	Figure Space	
&H2008	ChrW(8200)	Punctuation Space	
&H2009	ChrW(8201)	Thin Space	
&H200A	ChrW(8202)	Hair Space	Zeilen- und Absatztrenner
&H2028	ChrW(8232)	Line Separator	
&H2029	ChrW(8233)	Paragraph Separator	schmales geschütztes Leerz.
&H202F	ChrW(8239)	Narrow No-Break Space	
&H205F	ChrW(8287)	Medium Mathematical Space	
&H3000	ChrW(12288)	Ideographic Space	ideographisches Leerzeichen
Quelle: o. V., <i>Whitespace character</i> , URL: https://en.wikipedia.org/wiki/Whitespace_character , gefunden am 26.10.2016			

Tabelle 5: Leerraum (engl. white space)

Das Sonderformat **^w** in WORD steht für Leerraum oder jede Kombination von regulären und geschützten Leer- und Tabulatorzeichen.

12 Literaturverzeichnis

- [1] A. Prunkl, „How to professionally format your manuscript for editors, agents, and publishers,“ 24 10 2013. [Online]. Available: <http://penultimateword.com/editing-blogs/how-to-professionally-format-your-manuscript-for-editors-agents-publishers/>. [Zugriff am 28 08 2016].
- [2] D. Rado, „Remove all empty paragraphs from a document,“ 08 05 2016. [Online]. Available: <http://word.mvps.org/faqs/macrosvba/DeleteEmptyParas.htm>. [Zugriff am 05 08 2016].
- [3] C. H. Gast, „Einige Makros für Word (und deren Installation), Sub Satzzeichen_Leerzeichen(),“ 09 02 2015. [Online]. Available: <http://www.siebener-kurier.de/chris-aufsaeetze/Word-Makros.docx>. [Zugriff am 12 08 2016].
- [4] G. Mayor, „Finding and replacing characters using wildcards,“ [Online]. Available: <http://word.mvps.org/faqs/general/usingwildcards.htm>. [Zugriff am 11 08 2016].
- [5] A. Wyatt, „Strip Trailing Spaces,“ 28 02 2015. [Online]. Available: http://wordribbon.tips.net/T013376_Strip_Trailing_Spaces.html. [Zugriff am 12 08 2016].
- [6] A. Wyatt, „ An Automatic Two Spaces after a Period,“ 13 07 2015. [Online]. Available: http://wordribbon.tips.net/T010775_An_Automatic_Two_Spaces_after_a_Period.html. [Zugriff am 26 07 2016].
- [7] A. Wyatt, „Capitals After Colons,“ 10 01 2015. [Online]. Available: http://wordribbon.tips.net/T013352_Capitals_After_Colons.html. [Zugriff am 24 07 2016].
- [8] G. Mayor, „Is there any way to replace single quote marks with double quote marks?,“ 07 05 2014. [Online]. Available: http://answers.microsoft.com/en-us/office/forum/office_2013_release-word/is-there-any-way-to-replace-single-quote-marks/8af55925-6472-41b8-87ea-7ace3f6b6596?auth=1. [Zugriff am 28 07 2016].
- [9] T. Riem, „FussnotenPunkteEinfuegen,“ 12 02 2001. [Online]. Available: <http://support.citavi.com/forum/viewtopic.php?t=2851>. [Zugriff am 23 07 2016].
- [10] T. Riem, „CitaviNachbereitung Makro,“ 24 06 2009. [Online]. Available: <http://support.citavi.com/forum/viewtopic.php?t=2851>. [Zugriff am 23 07 2016].
- [11] R. Macklin, „A Word Macro for Processing Plain Text,“ 23 05 2013. [Online]. Available: <http://ryanmacklin.com/2013/05/word-process-plaintext/>. [Zugriff am 26 07 2016].
- [12] T. Barton, „Macro to replace smart quotes and smart apostrophes with straight quotes and straight apostrophes in Word,“ 23 02 2016. [Online]. Available: <http://www.anglopremier.com/blog/?p=569>. [Zugriff am 26 07 2016].
- [13] E. John, „ Widerborstige Anführungszeichen in Word,“ 25 06 2012. [Online]. Available: <http://ue-wie-uebersetzen.de/widerborstige-anfuehrungszeichen-in-word/>. [Zugriff am 02 11

2016].

- [14] J. Lyon, „Deleting Extraneous Carriage Returns in Footnotes and Endnotes,“ 22 05 2014. [Online]. Available: <http://editorium.com/archive/deleting-extraneous-carriage-returns-in-footnotes-and-endnotes/>. [Zugriff am 28 08 2016].
- [15] J. Nugent, „Useful Word Macros, General Text Cleanup,“ 11 02 2016. [Online]. Available: <http://rhetoricity.com/resources/macros.html>. [Zugriff am 28 08 2016].
- [16] G. Maxey, „Have an easy way to remove extra line breaks in Word,“ 02 01 2005. [Online]. Available: <http://www.wordbanter.com/showthread.php?t=2402>. [Zugriff am 30 08 2016].
- [17] G. Maxey, „How to strip lots of carriage returns (paragraph marker) in Word?,“ 09 09 2004. [Online]. Available: http://www.yqcomputer.com/1050_76126_1.htm. [Zugriff am 04 09 2016].
- [18] G. Mayor, „Replace a list of words from an array,“ 11 02 2016. [Online]. Available: http://www.gmayor.com/word_vba_examples_4.htm. [Zugriff am 07 09 2016].
- [19] F. Hookham, „Sub TidyUp(),“ 22 04 2004. [Online]. Available: <https://groups.google.com/forum/#!msg/microsoft.public.mac.office.word/t7Xq9jBEQOw/bdlqEAxv354J>. [Zugriff am 13 09 2016].
- [20] o. V., „Remove leading whitespace from a Word document,“ 11 03 2013. [Online]. Available: <http://stackoverflow.com/questions/15561633/remove-leading-whitespace-from-a-word-document>. [Zugriff am 22 07 2016].
- [21] o. V., „looping through a String Array,“ 01 12 2015. [Online]. Available: <http://www.vbaexpress.com/forum/showthread.php?51515-looping-through-a-String-Array>. [Zugriff am 18 08 2016].
- [22] o. V., „How to change straight quotes to curly quotes in word?,“ [Online]. Available: <https://www.extendoffice.com/documents/word/982-word-change-straight-quotes-to-curly-quotes.html>. [Zugriff am 03 11 2016].
- [23] o. V., „Some M/S Visual Basic Macros, Sub Spaces(),“ 29 11 2014. [Online]. Available: <https://theedixieflatline.wordpress.com/2014/11/29/some-ms-visual-basic-macros/>. [Zugriff am 13 09 2016].
- [24] o. V., „Some M/S Visual Basic Macros, Sub DeleteHangingParagraph(),“ 01 12 2015. [Online]. Available: <https://theedixieflatline.wordpress.com/2014/11/29/some-ms-visual-basic-macros/>. [Zugriff am 18 08 2016].
- [25] L. Laughsalot, „Format-cleanup document macro in Word 2010,“ 08 2015. [Online]. Available: <http://windowssecrets.com/forums/showthread.php/175068-Format-cleanup-document-macro-in-Word-2010>. [Zugriff am 12 08 2016].
- [26] A. Wyatt, „Removing Extra Paragraph Marks,“ 20 08 2016. [Online]. Available: http://wordribbon.tips.net/T000998_Removing_Extra_Paragraph_Marks.html. [Zugriff am 10

11 2016].

- [27] J. Smith, „Document Clean Up Tutorial,“ 21 01 2014. [Online]. Available: <https://www.youtube.com/watch?v=HoWj2itlv8c>. [Zugriff am 12 01 2017].