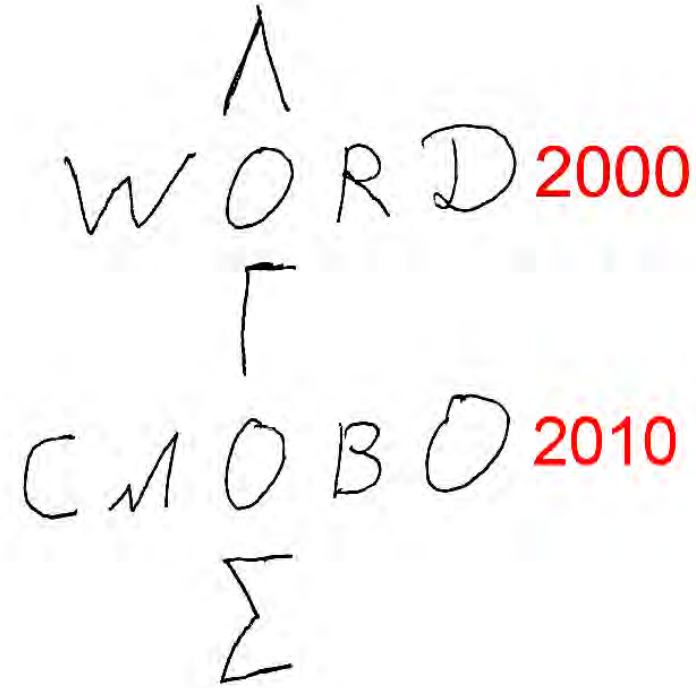
Hanna-Chris Gast

Einige Makros für Word (und deren Installation)



Inhalt

Seite

Seite

Seite

Vorwort

Für meine häufigsten Makros verwende ich Tastenkürzel, nicht ganz so häufig gebrauchte Makros lasse ich mit dem Buchstaben "a" + Ziffer beginnen, damit ich sie schneller aufrufen kann, weil sie dann als oberstes in der Liste unter Alt + F8 stehen. Für einige Makros habe ich auch im Menüband Icons zum Aufrufen eingefügt.

Die nachträgliche Zuordnung von Tastenkürzeln und Icons habe ich in meinem Aufsatz "**Tastenkombinationen (Tastenkürzel) für Word, Windows, Internet-Explorer und Adobe-Pdf"** beschrieben, siehe [1].

Für das Übertragen von eigenen Tastenkürzeln und (deutschen) AutoKorrekturen auf einen neuen Computer oder auf eine neue Word-Version biete ich ein **Formblatt mit Anklickbuttons** (und Makros) als Download ab, siehe [4]. Eine Beschreibung des Formblatts findet sich in [3].

Nach "leeren Kästchen" (Platzhalter) kann man in Word nicht suchen. Aber das Makro in Abschnitt 20 färbt alle Zeichen ein, die **nicht mit Cambria** darstellbar sind. Und nach Farbmarkierungen ("Hervorhebung", "Highlight") kann man bekanntlich sehr gut suchen!

Ein recht aufwendiges Makro (März 2017) ist ein Makro zur Farbanalyse an der Cursorposition mit Ausgabe von RGB-, CMYK-, RGB- und HSL-Farbwerten. Es funktioniert auch bei "Theme colours". **Es wird ferner angegeben, ob der Kontrast ausreichend ist,** siehe Abschnitt 15.2**.**

Ein Makro zur Umwandlung von Farben in Grautöne für den Schwarz-Weiß-Druck ist vielleicht auch ganz praktisch. Es wird dabei das ganze Dokument auf die Einhaltung von Mindestkontrast geprüft, und ggf. wird der Kontrast erhöht (15.3).

Am wichtigsten sind wohl die Makros für die Querverweis-Prüfung (a2) und die Hyperlink-Prüfung (a3).

Tipp: Ich benutze gern den Makro-Rekorder (siehe 1.1), um auf diese Weise an die Namen von Makro-Befehlen zu kommen. Ferner "google" ich gerne, um Ideen für ein benötigtes Makro zu erhalten. Natürlich muss ich dann auch die Quelle angeben.

Chris, 2018

Nachtrag (16.Dezember 2020 und 25. April 2023):

Einige der Makros funktionieren möglicherweise bei einer neuen Windows- oder Word-Version nicht mehr richtig. Einige Makros habe ich in der Vergangenheit auch schnell angepasst, ohne daran zu denken, die Änderungen hier zu notieren.

Über eine Rückmeldung würde ich mich freuen. Ich trage dann die Änderungen dann hier nach.

Ich arbeite weiterhin mit Word 2010 und bin nicht bereit, mir dauernd eine neue Word-Version kaufen zu müssen. Außerdem musste ich bis jetzt bei jeder neuen Word-Version die Makros teilweise anpassen. Aber auch bei Office-Updates passiert es immer wieder, dass ein Makro plötzlich nicht mehr funktioniert, und ich es umschreiben muss.

Chris

1 Erstellen und Starten von Makros

Makros bestehen aus einer oder mehreren Prozeduren. Die meisten davon fangen mit der Zeile "**Sub** xxxx()" an und enden mit der Zeile "**End Sub**". Dazwischen liegen die einzelnen Befehle.

1.1 Erstellen von Makros für Word mit dem Makro-Rekorder

Die Erstellung von Makros in Word ist relativ einfach. Man startet den Makro-Rekorder und führt den Ablauf aus, wie man es sonst von Hand machen würde. Nicht alles wird richtig aufgezeichnet, aber für einfache Makros reicht dies aus. Meist nehme ich die so erzeugten Makros als Basis für kompliziertere Makros. Wichtig: Der Name eines Makros muss mit einem Buchstaben beginnen. Leerzeichen und Bindestriche sind verboten. Umlaute werden akzeptiert, schränken den Anwendungsbereich aber auf Computer mit deutschsprachigem Windows ein.

Wer kompliziertere Makros schreiben will, lese eine Anleitung für VBA ("Visual Basic for Applications"). Für Anfänger ist eine Broschüre aus dem Internet (Pdf-Datei) oder ein kleines Taschenbuch aus der Buchhandlung sinnvoller als ein "dicker Wälzer".

a) Makrorekorder in Word 2000 bis Word 2003:
Extras → Makros → Aufzeichnen → Name vergeben → Tastenkombination wählen → Aufzeichnen
→ Aufzeichnen stoppen

(*dann prüfen, ob es geht und solange wiederholen, bis es funktioniert* ☺, *oder aufgeben* ☹)

b) Makrorekorder in Word 2010:
Entwicklertools → "Makroaufzchn." → "Makroname" → "Tastatur" (= Tastenkürzel wählen) → OK → "Aufzeichnung beenden"
oder unten in der Statusleiste von Word das Icon für Makroaufzeichnung anklicken, wenn dieses dort enthalten ist.

****

Problem:
Ist der Makrorekorder eingeschaltet, funktioniert die rechte Maustaste nicht.

Abhilfe:
Die Kontextmenü der rechten Maustaste kann man mit Shift + F10 aufrufen. Dann lässt sich der mit der Maus ausgewählte Befehl aus dem Kontextmenü anklicken.

1.2 Starten eines Makros

Wer für ein Makro kein Tastenkürzel vergeben hat oder es vergessen hat, kann eingeben:
Alt + F8
Dann kann man das Makro per Liste auswählen, vorausgesetzt, man hat dem Makro einen sinnvollen Namen gegeben, siehe das folgende Bild.

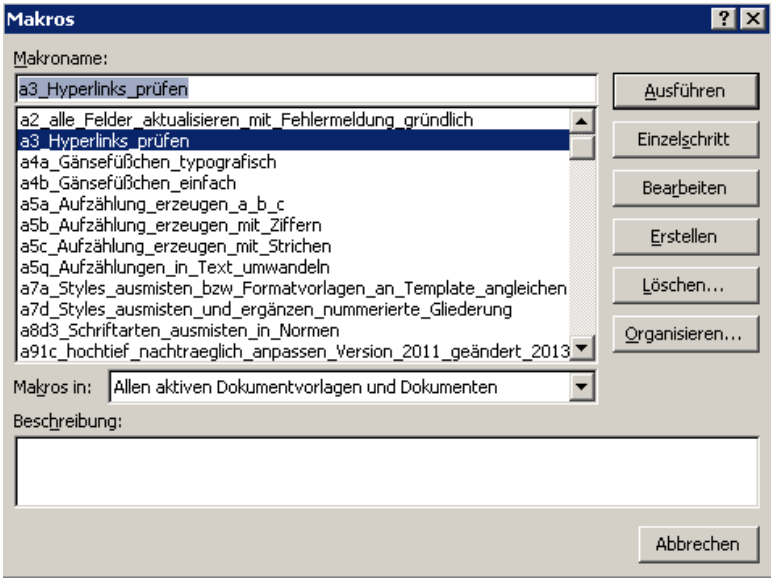


Bild 1.1: Fenster mit Makro-Liste

Dann ist "Ausführen" zu drücken.
Will man ein Makro bearbeiten, drücke man stattdessen den Button "Bearbeiten".

Zur vereinfachten Anwendung von häufig verwendeten Makros kann man diesen auch nachträglich Tastenkürzel zuordnen oder nachträglich Buttons oben in die Menüleiste von Word 2000 und Word 2003 legen (oberste Reihe), bzw. Buttons in die Schnellstartleiste von Word 2010 einfügen. Eine Anleitung dazu siehe [1] [1].

Man kann auch Makros über "Sub AutoStart()", "Sub AutoNew()" usw. oder mittels Formularfelder starten.

Wenig bekannt ist das "Macrobutton"-Feld, deshalb sei es hier noch erklärt:

— Einfügen in Word 2000 und Word 2003 über "*Einfügen → Feld*"
(in Word 2010 über "*Einfügen →Schnellbausteine → Feld*"),
— "MacroButton" auswählen, und dann rechts ein Makro auswählen und darüber einen sichtbaren Text eingeben, etwa "*Hier doppelklicken!*".

Ansicht im Feldcode: { MACROBUTTON Testmakro1 "Hier doppelklicken!" }
Die Gänsefüßchen sind hier überflüssig.

1 Dort zurzeit (Jan. 2013) Abschnitt 2.7.

1.3 Das Direktfenster ("Direktbereich")

Über das Pulldown-Menü "Ansicht" im Makro-Editor ("Visual Basic Editor") kann man verschiedene Fenster öffnen. Siehe folgendes Bild.

Links sieht man den "Makro-Explorer" mit den einzelnen Makromodulen,
rechts die Programmzeilen der jeweiligen Prozeduren (Makros) des links gewählten Makromoduls.

Nützlich ist zum Beispiel das Direktfenster ("Direktbereich"), in dem man einzelne Makro-Zeilen ausprobieren kann (im folgenden Bild unten rechts). Mit dem Befehl Alt + F11 gelangt man auch direkt in das Direktfenster.

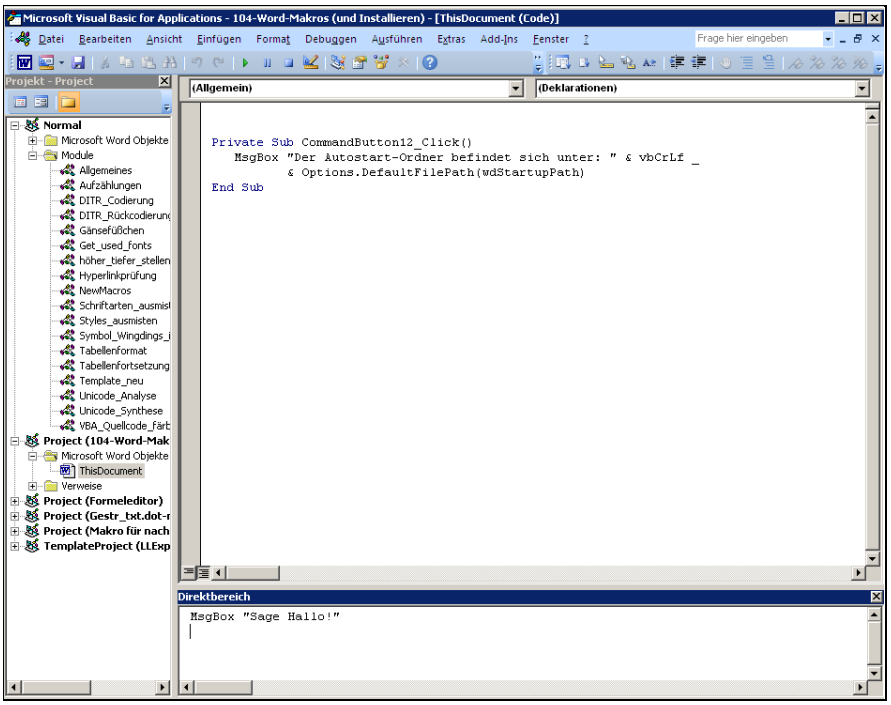


Bild 1.2: Der Makro-Editor (englisch: "Visual basic editor", "VBE")

Beispiele für die Anwendung des Direktfensters:
Pfad des STARTUP-Ordners:

```
MsgBox Options.DefaultFilePath(wdStartupPath)
```

Pfad für die Dokumentvorlagen ("dot-Dateien"):

```
MsgBox Options.DefaultFilePath(wdUserTemplatesPath)
```

oder

```
Debug.Print Options.DefaultFilePath(wdUserTemplatesPath)
```

Mittels "Debug.Print" wird das Ergebnis in das Direktfenster geschrieben.
Hat man damit einen Pfad gesucht, kann diesen dann über die Zwischenablage oben in die Adressleiste des Windows-Explorers kopieren und ihn damit leicht aufsuchen.

2 Installation von Makros

Es gibt in Word zwei unterschiedliche Prinzipien zur Installation eines Makros:

a) Einfügen eines einzelnen Makros über die Zwischenablage (siehe Abschnitt 2.1);

b) Einfügen eines Makromoduls von Hand (siehe Abschnitt 2.2).

Die Installation erfolgt von Hand. Ein Makro zum automatischen Installieren eines Makros und zur Zuordnung eines Tastenkürzels konnte ich nicht verwenden, weil bei mir zuhause der Virenschutz sofort Alarm meldete und das Öffnen der Datei nicht mehr zuließ.

2.1 Installieren von einzelnen Makros über die Zwischenablage

— Das gewünschte Makro in die Zwischenablage kopieren.

— Alt + F8 drücken

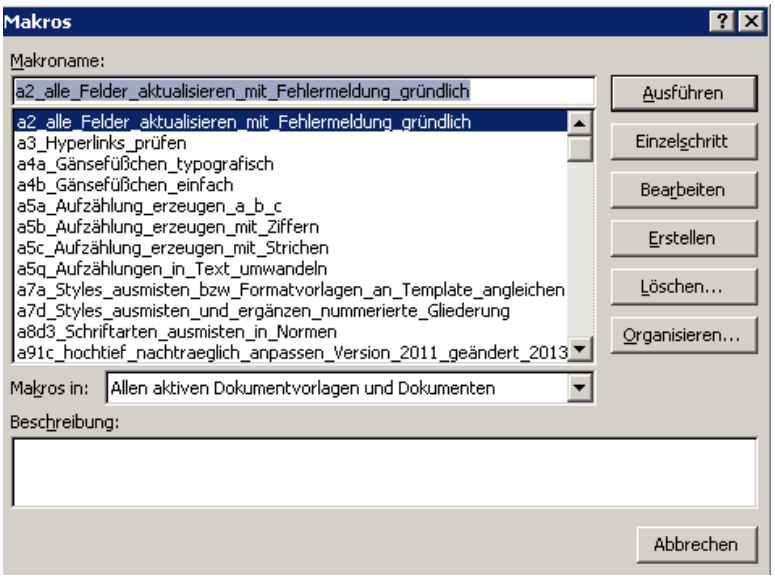


Bild 2.1: Fenster mit Makro-Liste

— Oben im sich öffnenden Fenster einige beliebige Buchstaben eingeben.

— Dann rechts den Button "**Erstellen**" drücken. Es öffnet sich das Makro-Fenster, siehe folgendes Bild:

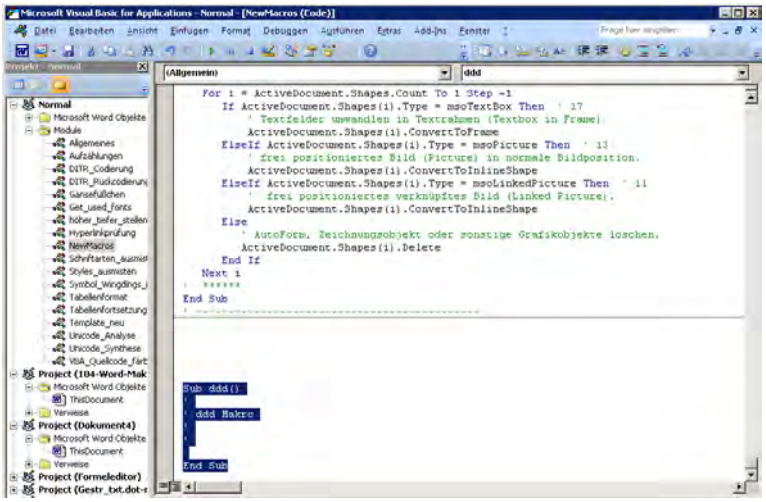


Bild 2.2: Fenster mit Makro-Liste

— Das neue Makro anstelle des erzeugten Makro-Torsos aus der Zwischenablage hineinkopieren. Dabei ist darauf zu achten, dass auch die erste Zeile " **Sub** xxxx()" mitkopiert wird (dieser Fehler kommt seltsamerweise bei mir öfters vor).

2.2 Installieren von kompletten Makromodulen

Fertige Makromodule können entweder als spezielle Datei (".bas") vorliegen, oder sie werden von einer Datei mit Makros per "Organisieren" in eine andere Datei, vorzugsweise in die "Normal.dot" bzw. "Normal.dotm" kopiert.

2.2.1 Installieren von Makromodulen, die als Textdatei vorliegen

Die zu installierenden Makros sind als Textdatei zu speichern bzw. aus dem Internet herunterzuladen ("downzuloaden"). Dabei ist nach dem Download ggf. die Textdatei-Endung "**.txt**" in die Makromodul-Endung "**.bas**" zu ändern.

In Word ist mit Alt + F11 oder über den Menüpfad das Makro-Fenster zu öffnen. Dabei ist es egal, welche Worddatei gerade geöffnet ist. Achtung: Im "Projekt-Explorer" (links oben im folgenden Bild, hier durch das Pull-Down-Menü zum größten Teil verdeckt) muss die "Normal.dot" gewählt sein.

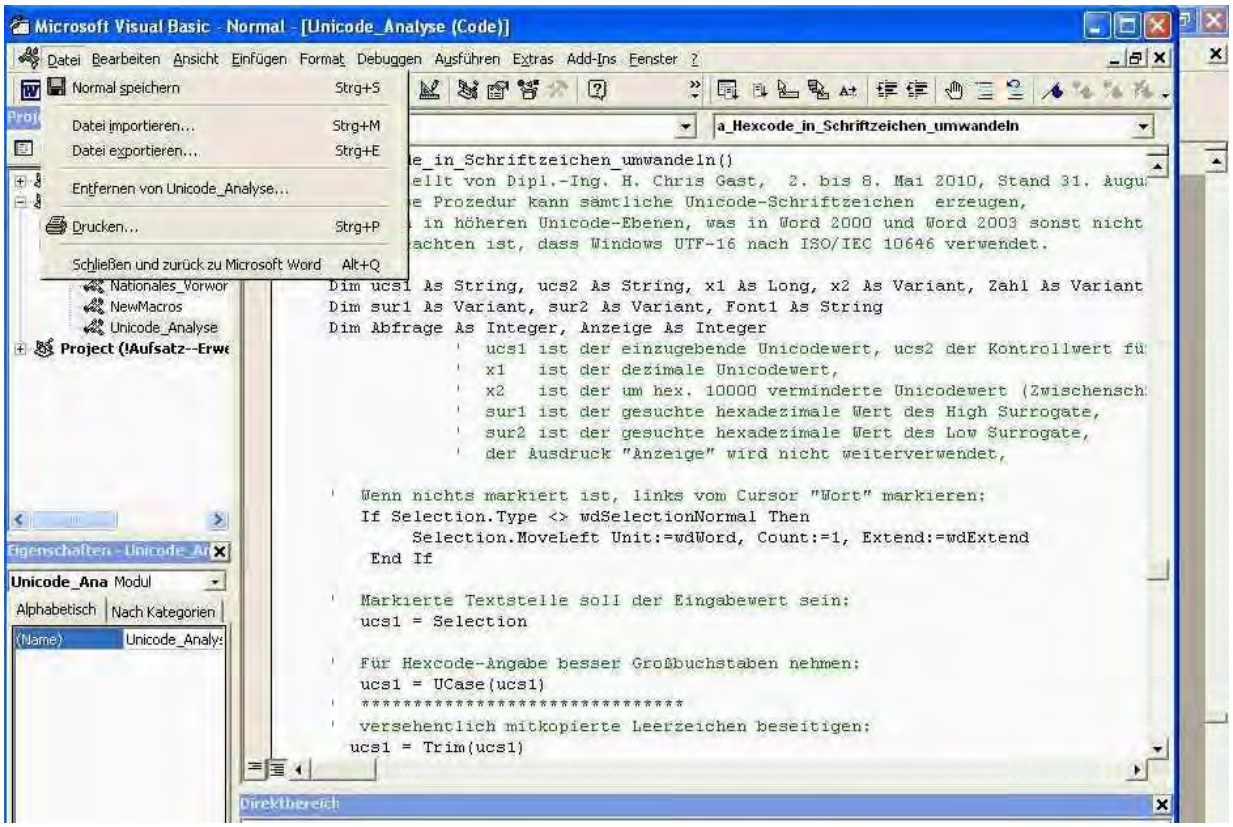


Bild 2.3: VBA-Fenster mit Makrozeilen

Oben links ist das Pulldown-Menü "Datei" anklicken, dann "**Datei importieren**". Damit wird dann das gewünschte Makromodul installiert.

2.2.2 Übertragen eines Makromoduls aus einer Word-Datei in eine Dot-Datei

Das Makromodul mit dem gewünschten Makro oder den Makros kann entweder in der Normal.dot (bzw. Normal.dotm bei Word 2010) oder in einer Dot-Datei im STARTUP-Ordner (siehe Abschnitt 2.2.3) untergebracht werden.

Die Datei mit den zu übertragenden Makros muss geöffnet sein.

Die Makro-Dialogbox ist mit Alt + F8 oder über den Menüpfad zu öffnen:

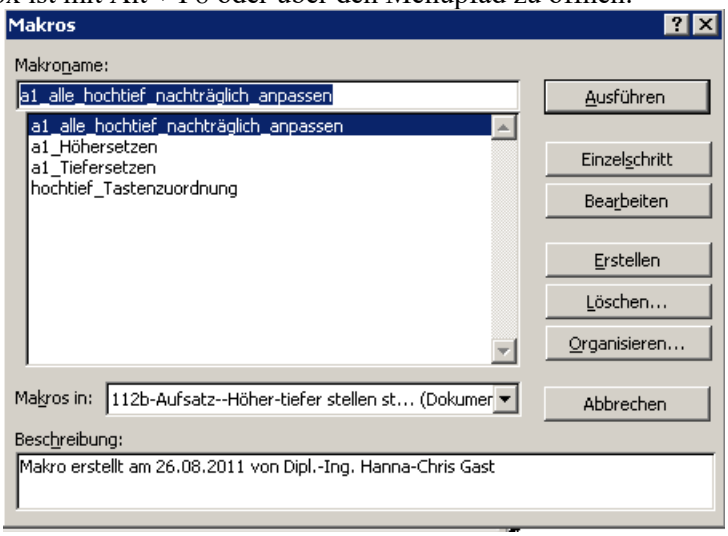


Bild 2.4: Makro-Dialogbox (Aufruf mit Alt + F8)

Hier ist rechts der Knopf "Organisieren" zu drücken und das Fenster ändert sich wie folgt:

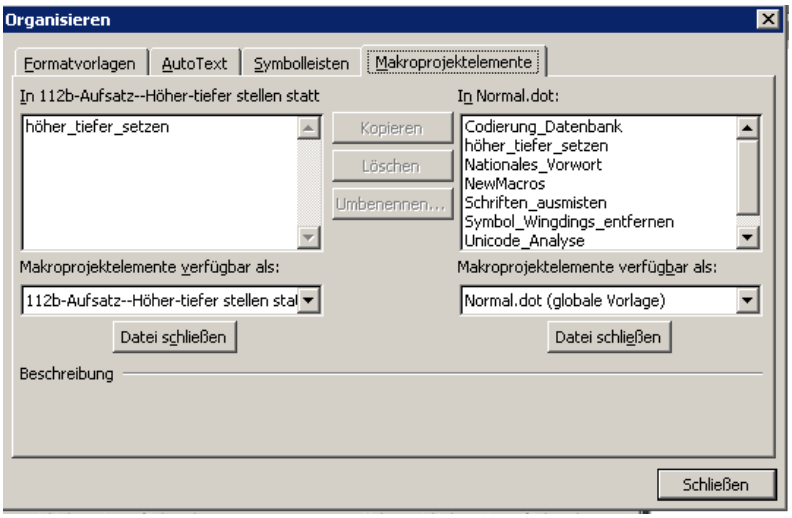


Bild 2.5: Makro-Dialogbox (Alt + F8, dann "Organisieren...")[2]

Links ist hier (im Reiter "Makroprojektelemente") das betreffende Makromodul anzuklicken und nach rechts in die "Normal.dot" bzw. "Normal.dotm" durch Druck des betreffenden Buttons zu kopieren (siehe Bild). Wenn eine andere Dot-Datei gewünscht wird, ggf. mit Anklicken von "Datei schließen" und anschließendem "Datei öffnen" diese andere dot-Dateien wählen.

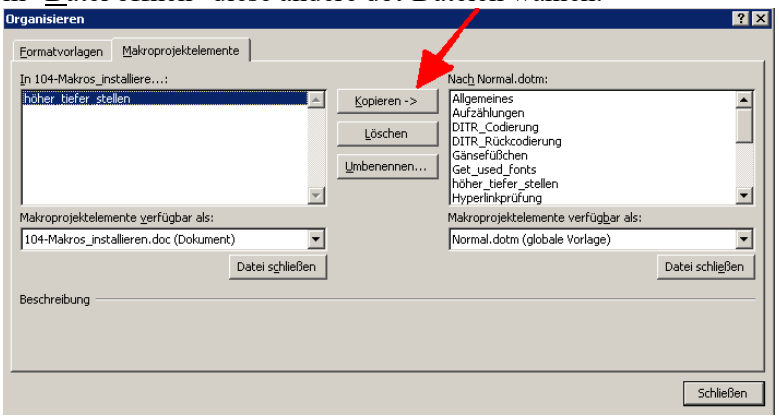


Bild 2.6: Makro-Dialogbox ("Kopieren" drücken)

2 Wie man im Bild sieht, kann man in Word 2000 und Word 2003 mittels "Organisieren" auch noch AutoText und eigene Symbolleisten übertragen.

Wenn das zu übertragende Makromodul "NewMacros" heißt oder den gleichen Namen wie eines der anderen Makromodule in der Normal.dot hat, muss es erst umbenannt werden. Enthält die Normal.dot eine veraltete Version des gleichen Makromoduls, ist sie vorher zu löschen, bevor der Button "Kopieren->" gedrückt werden kann.

Nach erfolgreichem Kopieren kann die Dialogbox wieder geschlossen werden. Achtung: Wenn Word beim Beenden fragt, ob die Änderungen in der Normal.dot ebenfalls gespeichert werden sollen, ist "ja" anzuklicken. Daran sind bei mir schon einige Makro-Installationen gescheitert.

2.2.3 Unterbringung einer Dot-Datei mit dem Makro im STARTUP-Ordner

Das Modul mit den Makros kann auch in einer dot-Datei im STARTUP -Ordner untergebracht werden. Der Pfad zum STARTUP-Ordner ist mittels Direktfenster ("Direktbereich") des Makro-Explorers
(Alt + F11) wie folgt zu finden:

```
MsgBox Options.DefaultFilePath(wdStartupPath)
```

Oder:

Pfad zum StartUp-Order unter Windows XP:

.....C:\Dokumente und Einstellungen\Benutzernamen\Anwendungsdaten\Microsoft\Word\StartUp

Pfad zum StartUp-Order unter Windows 7:

c:\users\Benutzernamenkürzel\appdata\roaming\microsoft\word\STARTUP

Exkurs zur Symbolleisten-Übertragung:
Man kann eine Dot-Datei auch benutzen, um selbstgemachte **Symbolleisten** aus Word 2000 und Word 2003 auch in Word 2010 weiterzubenutzen. Die alten zusätzlichen Symbol-Leisten aus Word 2000 bzw. Word 2003 wird in die Dot-Datei übertragen, die in Word 2010 in dem STARTUP -Ordner untergebracht wird. In Word 2010 wird sie mit einem Mausklick auf das Ribbon-Tab 3 "Add-Ins" erreicht, siehe folgendes Bild. In diesem sieht man eine Symbolleiste, bei der ein Test-Makro durch Anklicken des (hier testweise gewählten) Herz-Symbols gestartet wird.

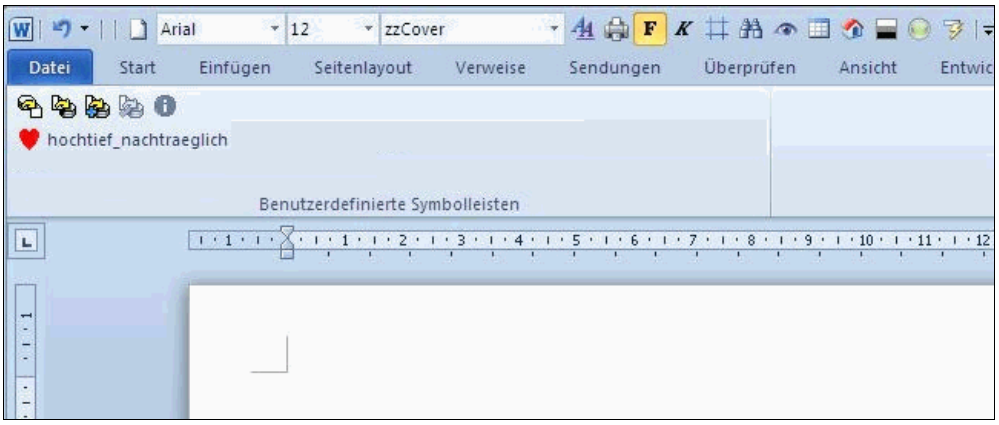


Bild 2.7: Symbolleiste aus einer alten Dot-Datei im STARTUP-Ordner von Word 2010

3 Eigentlich heißt "Ribbon-Tab" auf Deutsch "Menüband-Registerkarte".

2.3 Makros im Makrodialog "Alt + F8" unsichtbar machen

Werden Makros nur per Icons in einer Symbolleiste (in Word 2000 bis Word 2003) oder in der Schnellstartleiste bzw. in der Menüleiste (Ribbon) bei Word 2007 und folgende aufgerufen, kann man sie aus der Liste entfernen, die mit "Alt + F8" erscheint, weil man sie ja dort nicht mehr braucht. Dann kann man sie allerdings nicht mehr mit Tastenkürzeln aufrufen!

a) **Das jeweilige Makro mit einem Zusatz "Optional…" versehen**

```
Sub aaaa(Optional Irgendetwas As Boolean)
```

Statt "Irgendwas" kann man auch ein anderes Wort wählen, solange es keine Sonderzeichen enthält.[4]

b) **Das ganze Modul als "Privat" kennzeichnen.**

Damit die Makros eines Moduls nicht im Makrodialog erscheinen, kann oben im Makro-Modul vor dem ersten Makro die Zeile "**Option Private Module**" ergänzt werden (bei mir hinter "**Option Explicit**). Das Makro braucht nicht als "Privat" deklariert werden, es darf sogar "Public" vor dem "Sub" stehen.

Dann ist das Makro zwar immer noch öffentlich, aber nicht im Makrodialog sichtbar.

```
Option Explicit
Option Private Module   ' Dies soll die Makros aus der Liste mit Alt + F8 entfernen.

Sub Test()
   MsgBox "Test"
End Sub
```

2.4 Anleitung: Makromodul zwischen Dateien übertragen

Drücken Sie Alt + F8, dann drücken Sie rechts den Button Organisieren…
Dann erscheint folgendes Menü (siehe Bild):

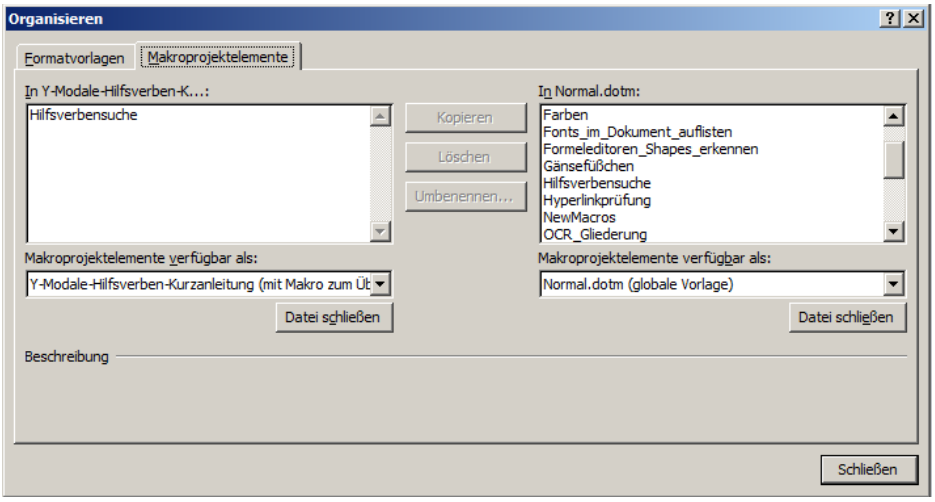


Bild 8: Menü "Organisieren"

Markieren Sie das gewünschte "Makroprojektelement" und drücken Sie den Button für Kopieren ->. Wollen Sie eine andere Datei statt der "Normal.dot", drücken Sie "Datei schließen, anschließend den neuen Button "Datei öffnen, usw.

[4] Im Makro eines Bretonen fand ich hierfür "Optional bennak", und ein Lateinfan würde vielleicht "aliquid" wählen.

2.5 Auflisten aller Makros

Eine Auflistung aller Makros per Makro (aus dem Internet) funktionierte bei mir nicht in Word 2010!

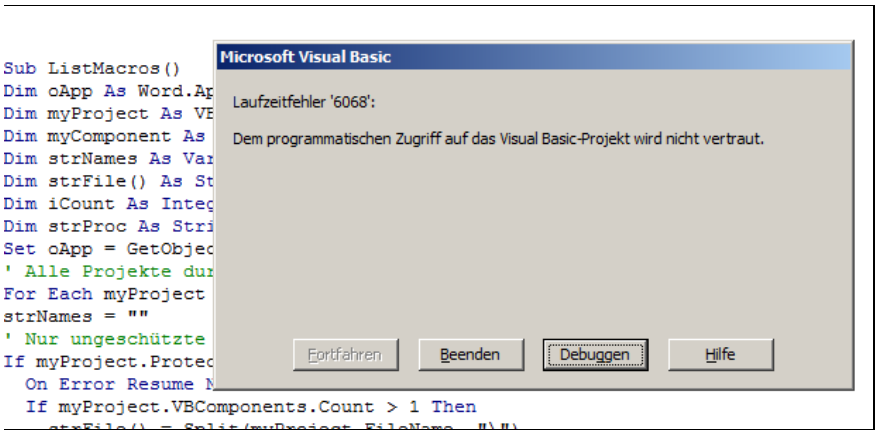


**Bild 2.9: Bild: Fehlermeldung "*Laufzeitfehler '6068':*
Dem programmatischen Zugriff auf das VisualBasic-Projekt wird nicht vertraut"**

Abhilfe *(theoretisch)*: Sicherheitseinstellungen lockern. Ein Herabsetzen der Sicherheitseinstellungen kommt für mich aber nicht in Frage!

Elegantere Abhilfe ohne Makro:
"Alt F8"→"Word-Befehlen" → "BefehleAuflisten" → "Alle Word_Befehle".

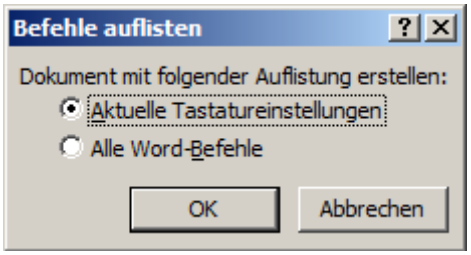


Bild 2.10: Bild: Screenshot

Dann in der damit erzeugten Tabelle alle Zeilen löschen, außer den Zeilen, die mit "Normal." anfangen.

Dann bleibt die gewünschte Liste übrig, die bei mir wie folgt beginnt:

Normal.Allgemeines.a8a_Seitenzahlen_oben_mittig_mit_Strichen_davor_und_danach
Normal.Allgemeines.a8b_Seitenzahlen_unten_außen
… usw.

3 Mehrere Bilder mit Dateiname als Bildtitel gleichzeitig in eine Word-Datei einfügen

Mit dem folgenden Makro füge ich ausgewählte Bilder eines Ordners mitsamt Dateinamen als Bildtitel in eine Word-Datei ein. Ferner wird auf maximal 12,5 Zentimeter skaliert.

```
Sub ausgewählte_Bilder_einfügen_mit_Titel_per_Filepicker()
' H. Chris Gast, neu 3. + 4. April 2019, 28. + 30. Oktober 2019.
' Dieses Makro fügt die ausgewählten Bilder aus dem gewählten Ordner
' mit Dateiname als Bildtitel an der Markierung in die aktuelle Datei ein.
' Funktioniert in Word 2010, nicht in Word 2000 (wegen "Filepicker").
' Filepicker-Anwendung nach "http://www.0711office.de/vba/FileDialog/default.htm".
' Gelöst: Bild und Bildtitel müssen stets zusammenbleiben.
'
' Neu: Es können Bilder ausgewählt werden.
'   Die Bilder werden auf 12 Zentimeter Breite verkleinert.
'   Es wird auch die Höhe begrenzt.
'   Im Fehlerfall wird bei der nächsten Datei weitergemacht.
' *****************
Dim Rng0 As Range
Dim oFileDialog As FileDialog
Dim vItem As Variant
Dim strPfad As String
Dim BlRänder As Boolean: BlRänder = False
'  ****************
Application. screenUpdating = False
'  Formateinstellungen merken:
'  Neuen Absatz erzeugen:
  Selection.Paragraphs(1).Range.InsertParagraphAfter
  Selection.MoveDown Unit:=wdParagraph
  Selection.Style = wdStyleNormal
'  Anfangsstelle für die Überschrift merken:
  Set Rng0 = Selection.Range
'  Anfangswert für Pfad
  If strPfad = "" Then
    '' strPfad = CurDir
    strPfad = Options.DefaultFilePath(wdDocumentsPath)
  Else
    strPfad = ActiveDocument.Path ' geht nicht bei neuen Dateien ohne Namen!
  End If
'  ****************
'' Options.DefaultFilePath(Path:=wdPicturesPath) = ActiveDocument.Path
''  Bilder per File-Picker wählen:
  Set oFileDialog = Application.FileDialog(msoFileDialogFilePicker)
  With oFileDialog
    .Title = "Grafiken auswählen"
    .ButtonName = "Bilder einfügen"
    .Filters.Clear
    .InitialFileName = strPfad & "\*"
    ''.Filters.Add "Alles", "*.*"
    .Filters.Add "Nur Grafiken", "*.jpg;*.gif;*.tif;*.tiff;*.png", 1
    .AllowMultiSelect = True
    If .Show = True Then
      Selection.TypeParagraph
      For Each vItem In .SelectedItems
        Call Einzelbild_einfügen(vItem)
      Next
    Else
      MsgBox "Es wurde kein Bild ausgewählt.", vbInformation
```

```
      Exit Sub
    End If
  End With
'  ****************
'  Überschrift darübersetzen mit Pfad des ersten Bildes
'  ****************
'  Pfadname:
  strPfad = CStr(oFileDialog.SelectedItems(1))
  strPfad = Left(strPfad, InStrRev(strPfad, "\") - 1)
Anfangsstelle_aufsuchen:
  Rng0.Select
  Selection.Text = "Bilder (Grafiken) aus dem Pfad " & strPfad
'  Als Überschrift 2, zentriert und fett:
  Selection.Style = wdStyleHeading2
    Selection.Font.Bold = True ' (ist wahrscheinlich überflüssig)
    Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
'  Cursor an den Anfang:
  Selection.Collapse Direction:=wdCollapseStart
'  Bildschirm aktualisieren:
  Application.ScreenUpdating = True
End Sub
' ----------------------------------------------------------------------------------------------------------


Private Sub Einzelbild_einfügen(vItem As Variant)
' H. Chris Gast, 3. + 4. April 2019, 28. + 30. Oktober 2019.
' Fügt Einzelbild und dessen Dateiname (ohne Pfad) ein.
' Die Breite bzw. Höhe wird beschränkt auf 12,3 cm, wenn nicht Spaltensatz ist.
' Es ist (zumindest bei mir) eine Warteschleife nach dem Einfügen jedes Bildes nötig.
' Im Fehlerfall wird die Unterprozedur abgebrochen, etwa wenn es keine Bild-Datei ist.
' *****************
Dim strFullname As String
  strFullname = CStr(vItem)   ' einzufügende Bilddatei
Dim strBilddateiname As String   ' Name für den Bildtitel
  ''strBilddateiname = Mid(strFullname, InStrRev(strFullname, "\") + 1)
  strBilddateiname = Dir(vItem)
Dim sgHöhe As Single
Dim sgBreite As Single
Dim sgFaktor As Single
Dim sgWunschbreite As Single
Dim sgWunschhöhe As Single
'  ****************
'  Bild einfügen (als InlineShape)
On Error GoTo Zeile_Fehlerbehandlung
  Selection.Style = wdStyleNormal
  Selection.InlineShapes.AddPicture FileName:=strFullname, _
    LinkToFile:=False, SaveWithDocument:=True
  With Selection.ParagraphFormat
    .SpaceBefore = 0
    .SpaceBeforeAuto = False
    .SpaceAfter = 0
    .SpaceAfterAuto = False
    .Alignment = wdAlignParagraphCenter  ' Bild zentrieren
    .KeepWithNext = True                 ' Wichtig!
  End With
  Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter ' wdAlignLeft
'  Auf 12 Zentimeter Breite begrenzen (außer bei Spaltensatz):
  Selection.Paragraphs(1).Range.Select
  ''aBild = Selection.InlineShapes(1)
  If Selection.PageSetup.TextColumns.Count < 2 Then
    sgHöhe = Selection.InlineShapes(1).Height  '>= 150 Then
    sgBreite = Selection.InlineShapes(1).Width
```

```
      ' 12,3 Zentimeter ergeben 348,66 Points:
      ''''''''
      If sgBreite > sgHöhe Then
        sgWunschbreite = CentimetersToPoints(12.3)
        ' zum Beispiel 12.3 cm entsprechen 348,6614 Points
        sgFaktor = sgWunschbreite / sgBreite
      Else
        sgWunschhöhe = CentimetersToPoints(12.3)
        sgFaktor = sgWunschhöhe / sgHöhe
      End If
      ' Neues Format:
      Selection.InlineShapes(1).Height = sgFaktor * sgHöhe
      Selection.InlineShapes(1).Width = sgFaktor * sgBreite
    End If
'  ****************
    ' Warteschleife nach Bild-Einfügen:
    Dim tt As Single
    tt = Timer
    Do While Timer < tt + 0.05  'Sekunden
      DoEvents
    Loop
    DoEvents
'  ****************
'  Dateiname unter das Bild setzen:
    Selection.Paragraphs(1).Range.InsertParagraphAfter
    Selection.MoveDown Unit:=wdLine
    ' ***
    With Selection
      .Font.Size = 10  ' Die Bildtitel passen sonst oft nicht in eine Zeile.
      .Font.Bold = True
      .Font.Name = "Arial"
      .Text = strBilddateiname
    End With
    With Selection.ParagraphFormat
      .SpaceBefore = 6
      .SpaceBeforeAuto = False
      .SpaceAfter = 12
      .SpaceAfterAuto = False
      .Alignment = wdAlignParagraphCenter
      .KeepWithNext = False
    End With
'  ****************
'  Neuen Absatz daruntersetzen:
    Selection.Paragraphs(1).Range.InsertParagraphAfter
    Selection.MoveDown Unit:=wdParagraph
    Selection.Style = wdStyleNormal
Exit Sub
' *****************
Zeile_Fehlerbehandlung:
    Selection.Paragraphs(1).Range.InsertParagraphAfter
      Selection.MoveDown Unit:=wdLine
      Selection.Text = "Die Datei """ & strBilddateiname & """ lässt sich nicht als Bild einfügen!"
      Selection.Font.ColorIndex = wdRed
    Selection.Paragraphs(1).Range.InsertParagraphAfter
      Selection.MoveDown Unit:=wdLine
    Err.Clear
End Sub
' ---------------------------------------------------------------------------------------------------
```

4 Alle Querverweisungs-Felder und Inhaltsverzeichnisse aktualisieren

Gelegentlich begegnen mir in einem Text Formulierungen wie die folgende:

"(siehe **Fehler! Verweisquelle konnte nicht gefunden werden**)…".

bzw. bei englischen Manuskripten:

"(see **Error! Reference source not found**) ..."

Die dazugehörige Abfrage in VBA lautet:

```
If rng.Fields(i).Update = False Then
```

Manche fehlerhaften Querverweise zeigen jedoch nicht "Fehler! ..." an, sondern "**0**" an oder "**[0]**". Diese Anzeige "Null" entsteht dadurch, dass die Nummerierung der Zieladresse verloren ging. Nur in sehr seltenen Fällen gibt es wirklich einen Abschnitt "0".5

Leider wird bei der Abfrage aller Felder per Makro auch das Ergebnis "0" gemeldet, wenn das Feldergebnis leer ist und nichts enthält. Deshalb frage ich auch den Feldcode (bzw. "Feldtyp") ab. Nur wenn dieser mit "REF" beginnt wie zum Beispiel { REF _Ref312063926 \r \h }, handelt es sich um einen echten Querverweis. StyleRef-Felder { STYLEREF "Überschrift 1" \n } werden vom Makro nicht abgefragt. Dies hat sich aber bis jetzt nicht als nötig erwiesen.

Fehlerhafte Querverweise werden rot markiert. Es werden auch Querverweise erfasst, die als Ergebnis "0" ("Null") anzeigen, was Word sonst nicht als Fehler erkennen würde. Dies geschieht bei Aufsätzen mit Verweisungen auf Nummerierungen, wenn diese Nummerierungen entfernt wurden oder gar die betreffenden nummerierten Abschnitte entfernt wurden.

Der extrem seltene Fall einer Verweisung auf einen "Abschnitt 0" wird dann allerdings fälschlich als Fehler gesehen. Dies ist mir aber noch nie begegnet, dagegen ein fehlerhaftes Feldergebnis "0" ziemlich oft.

Die eigentliche Abfrage ist dann:

```
If aField.Update = False _
Or aField.Result = "0" _
Or aField.Result = "[0]" Then
```

Dies müsste meines Erachtens auch in nicht-deutschsprachigen Word-Versionen funktionieren.

Gegenüber der Fassung von 2010 habe ich Befehlszeilen ergänzt, welche vorübergehend die **Änderungsverfolgung ausschalten**. Stattdessen wird der zu untersuchende Text vor der Aktualisierung der Felder gespeichert, so dass man notfalls über die gespeichert Fassung feststellen kann, wie die fehlerhaften Querverweise vorher aussahen. Ferner werden in der neuen Version auch **Textfelder in Kopfzeilen** erfasst, obwohl dieser Fall in der Praxis wohl nie vorkommen wird. Es werden auch Kopf- und Fußzeilen erfasst, die nicht sichtbar sind, wenn ein "Section" nur eine einzige Seite hat (jedes "Section" hat sonst bis zu drei verschiedene Kopf- und Fußzeilen).

Das folgende Makro aktualisiert alle Felder in allen Komponenten einer Word-Datei sowie das Inhaltsverzeichnis und gibt fehlerhafte aktive Querverweise in einer Liste aus.

```
Option Explicit

Private Fehlermeldung As String
Private Fehlernummer As Integer
Private iSection As Integer      ' ("Section" = "Abschnitt")
Private Rng2 As Range            ' (Bereich des 1. gefundenen Fehlers)
```

5 Als Querverweis ist mir dieser Fall noch nie begegnet.

```
' ******************************


Sub a2_alle_Felder_aktualisieren_mit_Fehlermeldung()
'  Dipl.-Ing. Hanna-Chris Gast, 26. April 2014, Stand 13. August 2014, 30. August 2015.
'  Dieses Makro aktualisiert alle Felder in der Datei, auch in Tabellen, Fußnoten, Fußzeilen und
'  Kopfzeilen, Textfeldern;  Felder in Kopf- und Fußzeilen weiterer Sections werden auch erfasst
'  (jedoch keine Felder in Textfeldern von Kopf- oder Fußzeilen, was kaum vorkommt).
'  Es wird von dem Makro ein Fehler gemeldet, wenn auf einen Abschnitt "0" verwiesen wird.
'  Meistens liegt dann ein Fehler vor. Nur in seltenen Fällen gibt es wirklich einen Abschnitt "0".
'
'  Das Makro meldet fehlerhafte Querverweise und markiert sie rot.
'  Man kann diese Fehler anschließend anhand  der Seitenangaben in der Fehlermeldung finden.
'  Ausdruck der Fehlermeldung in separate Datei auf Wunsch oder bei zu vielen Fehlern.
'
'  Dieses Makro aktualisiert auch Inhaltsverzeichnisse.
'
'  Offenes Problem: Es werden auch Fehler in nicht anzeigbaren Kopf- und Fußzeilen aufgelistet.
'  ******************************
Dim Änderungsverfolgung As Boolean
Dim Ansichtstyp1 As Integer
Dim Rng0 As Range  ' Markierung bei Start des Makros
Dim Rng1 As Range  ' Bereich der jeweiligen Sections
Dim Zeit1 As Single
Dim Zeit2 As Single
Dim Zeitdauer As Single
Dim i As Integer
Fehlernummer = 0
Fehlermeldung = ""
Dim Abfrage As Integer
Dim Anzeigetext As String
Dim Dateiname As String
Dim Seitenzahlanfang As Integer
Dim Seitenzahlende As Integer
Dim rngStory As Range        ' Dokumentkomponente
Dim rngSection As Section
Dim rngHeaderFooter As HeaderFooter
Dim shpShape As Shape        ' Textfelder und anderes Shapes in den Kopf- und Fußzeilen.
Dim BL_Textfeld As Boolean   ' Für Suche nach Textfeldern in Kopf- oder Fußzeile
'  *******************************
'  Beim Aktualisieren von Feldern und Inhaltsverzeichnissen sollte die Änderungsverfolgung
'  abgestellt sein:
  Änderungsverfolgung = ActiveDocument.TrackRevisions
  ' MsgBox "Die Änderungsverfolgung ist " & Änderungsverfolgung
  ActiveDocument.TrackRevisions = False
'  ******************************
'  Zeitmessen für die Rechendauer:
  Zeit1 = Timer
'  ******************************
'  Vorab Dokument speichern, dann wird vom Makro auch das (Speicher-)Datum aktualisiert.
  ' Lästige Nachfrage(Warnmeldung) in Word 2000 beim Aktualisieren aller Felder
  ' unterdrücken (durch das Speichern ist eine Rückstellung ja möglich):
  Application.DisplayAlerts = wdAlertsNone
'  ******************************
'  Zuerst die vorhandene Bildschirmansicht speichern:
  Ansichtstyp1 = (ActiveWindow.View.Type)
'  Bildschirm zur Beschleunigung auf "Normalansicht" stellen, wenn dies nicht schon der Fall ist:
  If ActiveWindow.View.Type = wdNormalView = False Then
    ActiveWindow.View.Type = wdNormalView
  End If
'  Bildschirmaktualisierung während des Programms zur Beschleunigung abstellen.
  StatusBar = "Bitte warten - das Makro läuft."   '  Geht nicht in Word 2010.
```

```
  Application. screenUpdating = False
"   Options.Pagination = False
'  ******************************
'  Anfangsmakierung speichern:
  Set Rng0 = Selection.Range
' ******************************************************************************
' Alle Komponenten des Dokuments durchsuchen:
' ******************************************************************************
' Alle Dokumentkomponenten außer Fuß- und Kopfzeilen:
 For Each rngStory In ActiveDocument.StoryRanges
   If rngStory.StoryType < 6 Then
     '(die Trennlinie über Fußnoten soll nicht erfasst werden)
     Call Querverweisprüfen_In_Range(rngStory)
   End If
 Next
' ****************************************
' Kopf und Fußzeilen (in allen Sections):
 For Each rngSection In ActiveDocument.Sections
  rngSection.Range.Select
  iSection = Selection.Sections(1).Index
  ' Kopfzeilen:
  For Each rngHeaderFooter In rngSection.Headers
   If rngHeaderFooter.LinkToPrevious = False Then
     Call Querverweisprüfen_In_Range(rngHeaderFooter.Range)
     '''''''''''''''''''''
     ' Suche nach Textfeldern in Kopf-/Fußzeilen nur ein einziges Mal durchführen!
     ' (sie werden sonst bei jedem rngHeaderFooter gefunden!)
     If BL_Textfeld = False Then
       For Each shpShape In rngHeaderFooter.Shapes
         If shpShape.Type = msoTextBox Then
           Call Querverweisprüfen_In_Range(shpShape.TextFrame.TextRange)
         End If
       Next
       BL_Textfeld = True
     End If
     '''''''''''''''''''''
   End If
  Next
  ' *********
  ' Fußzeilen:
  For Each rngHeaderFooter In rngSection.Footers
   If rngHeaderFooter.LinkToPrevious = False Then
     Call Querverweisprüfen_In_Range(rngHeaderFooter.Range)
   End If
  Next
 Next
' ************************************************************************
' Seitenzahlen der "Sections":
' ************************************************************************
' Seitenzahlenbereiche der "Sections" ermitteln, wenn dort Fehler in Kopf- oder Fußzeilen:
 If InStr(Fehlermeldung, "Kopfzeile") > 0 _
 Or InStr(Fehlermeldung, "Fußzeile") > 0 Then
  '' If Fehler2 = True Then
     For i = 1 To ActiveDocument.Sections.Count
       Set Rng1 = ActiveDocument.Sections(i).Range
       Seitenzahlende = ActiveDocument.Sections(i).Range.Information(wdActiveEndPageNumber)
       Rng1.Collapse Direction:=wdCollapseStart
       Seitenzahlanfang = Rng1.Information(wdActiveEndPageNumber)
       Fehlermeldung = Fehlermeldung & vbCrLf & "Abschnitt ('Section') " & i & " beginnt auf Seite " _
         & Seitenzahlanfang & " und endet auf Seite " & Seitenzahlende & "."
     Next i
     Fehlermeldung = Fehlermeldung & vbCrLf
```

```
    ' Gibt es keine Sections, braucht "in Section 1" nicht angegeben zu werden:
    If ActiveDocument.Sections.Count = 1 Then
      Fehlermeldung = Replace(Fehlermeldung, " in 'Section' 1", "")
    End If
  End If
' ********************************************************************
' alle Inhaltsverzeichnisse (und Bilderverzeichnisse) aktualisieren:
' ********************************************************************
Zeile_Inhaltsverzeichnisse_aktualisieren:
  ''If ActiveDocument.TablesOfContents.Count > 0 Then
  ' (das folgende gibt keinen Fehler, wenn es keine Verzeichnisse gibt.)
  For i = 1 To ActiveDocument.TablesOfContents.Count
     ActiveDocument.TablesOfContents(i).Update
   Next i
' Inhaltsverzeichnisse für Bilder und Tabellen:
  For i = 1 To ActiveDocument.TablesOfFigures.Count
     ActiveDocument.TablesOfFigures(i).Update
  Next i
' ********************************************************************
' ursprüngliche Bildschirmansicht wiederherstellen
' und markierte Stelle wieder markieren:
' ********************************************************************
  Rng0.Select    ' Funktioniert nur in der Dokument-Hauptkomponente.
  ActiveWindow.View.Type = Ansichtstyp1
  Options.Pagination = True
  StatusBar = False
  Application.ScreenUpdating = True
  Application.DisplayStatusBar = True
  Application.DisplayAlerts = wdAlertsAll
' ********************************************************************
' Zeitmessen für die Rechendauer:
      Zeit2 = Timer
      Zeitdauer = Zeit2 - Zeit1
      Zeitdauer = Round(Zeitdauer, 2)
' ********************************************************************
' Ausgabe des Ergebnisses in MsgBox und/oder in separate Datei:
' ********************************************************************
  If Fehlernummer > 0 Then
  ' Setze den Cursor an die Stelle des ersten Fehlers:
      '' If ActiveDocument.Bookmarks.Exists("Fehler1") Then
      ''    Selection.GoTo What:=wdGoToBookmark, Name:="Fehler1"
      '' gab Fehlermeldung, wenn "Fehler1" in Fußnote.
    'ActiveDocument.Bookmarks("Fehler1").Range.Select
    Rng2.Select
    '
    ' Fehlermeldung um Rechenzeit ergänzen:
    Fehlermeldung = Fehlermeldung & vbCrLf _
          & "Die Rechenzeit der Querverweisprüfung war " & Zeitdauer & " Sekunden."
    ' Bei zuviel Fehlern passen nicht mehr alle Zeilen in eine Messagbox.
    If Fehlernummer > 6 Then
      Abfrage = 6
    Else
      Anzeigetext = "Fehlerhafte Querverweise" & vbCrLf & Fehlermeldung & vbCrLf _
              & "Wollen Sie das Ergebnis in einer separaten Datei?"
      Abfrage = MsgBox(Anzeigetext, vbYesNoCancel + vbDefaultButton2, _
             "Prüfunge der Querverweise")
       ' Bei "Ja" kommt "6" heraus, bei "Nein" kommt "7" heraus, bei Abbrechen eine "2".
    End If
    If Abfrage = 6 Then
      Dateiname = ActiveDocument.Name
      ' Anzeige in separater Datei:
      Documents.Add DocumentType:=wdNewBlankDocument
```

```
      ' falls in der Normal.dot schon Name und Datum vorgesehen waren, dieses stehe lassen:
      Selection.EndKey Unit:=wdStory
      ' Satzspiegel verbreitern, längere Zeilen zulassen:
      With ActiveDocument.PageSetup
        .LineNumbering.Active = False
        .LeftMargin = CentimetersToPoints(1.5)
        .RightMargin = CentimetersToPoints(1.5)
        .Gutter = CentimetersToPoints(0)
        .PageWidth = CentimetersToPoints(21)
        .PageHeight = CentimetersToPoints(29.7)
        .OddAndEvenPagesHeaderFooter = False
        .DifferentFirstPageHeaderFooter = False
      End With
      ' Absatzabstand auf Null stellen und Einrückung:
      With Selection.ParagraphFormat
        .Alignment = wdAlignParagraphLeft
        .LeftIndent = CentimetersToPoints(0.5)
        .FirstLineIndent = CentimetersToPoints(-0.5)
        .SpaceBefore = 0
        .SpaceAfter = 0
        .SpaceAfterAuto = False
        .Alignment = wdAlignParagraphLeft
      End With
      Selection.Font.Name = "Arial"
      Selection.Font.Size = 10
      ' *********
      Selection.TypeParagraph
      Selection.TypeParagraph
      Selection.TypeParagraph
      Selection.Font.Bold = True
      Selection.TypeText Text:="Fehlerhafte Querverweise in """ & Dateiname & """:"
      Selection.Font.Bold = False
      Selection.TypeParagraph
      Selection.Font.Size = 8
      Selection.TypeText Text:=Fehlermeldung
    End If
  Else
    MsgBox "Keine fehlerhaften Links." & vbCrLf _
        & "Die Rechenzeit der Querverweisprüfung war " & Zeitdauer & " Sekunden."
  End If
'  ***************************************************************
' Änderungsverfolgung ggf. wieder einschalten:
 ActiveDocument.TrackRevisions = Änderungsverfolgung
End Sub


Private Sub Querverweisprüfen_In_Range(Rng1)
'  Alle Felder aktualisieren in der Dokumentkomponente
'  und nur Felder mit Querverweisen ("wdFieldRef") prüfen:
'  ****************************************
'Dim i As Integer  ' Nummer des Feldes in der Dokumentkomponente
Dim Fehlerseitenzahl As Integer
Dim Fehlerseitenzahlangabe As String
Dim Sectionangabe As String
Dim afield As Field
'  ****************************************
'  Beschleunigung: Es werden nur noch die "Ref-Feldern" geprüft ("wdFieldRef").
''   Options.Pagination = False
'  ****************************************
   ' Der normale Lösungsweg wäre lediglich:  "If aField.Update = False Then..."
   ' Damit wird aber kein fehlerhaftes Ergebnis mit der Anzeige "0" gefunden,
   ' wenn es eine Abschnittsnummer, auf die verwiesen wird, nicht mehr gibt.
```

```
  ' Deshalb wird hier auch nach dem Feldergebnissen "Fehler!..." sowie nach "0" bzw. "[0]" gesucht.
'  ***************************************
  Sectionangabe = " in 'Section' " & iSection & ":"
  Application.DisplayAlerts = wdAlertsNone
  Rng1.Fields.Update   ' Bei "{ SET Quantity { FILLIN "Enter number...:"  }  }" gibt es Problem!
' ***************************************
  For Each afield In Rng1.Fields
    "aField.Update
    If afield.Type = wdFieldRef Then
"       If InStr(aField.Result, "Fehler!") <> 0 _
"       Or aField.Result = "0" _
"       Or aField.Result = "[0]" Then
      If afield.Update = False _
      Or afield.Result = "0" _
      Or afield.Result = "[0]" Then
        ' Der Querverweis ist fehlerhaft!
        afield.Select
        Selection.Range.HighlightColorIndex = wdRed
        Fehlernummer = Fehlernummer + 1
        Fehlerseitenzahl = Selection.Information(wdActiveEndPageNumber)
        '''''''
        ' bei Kopf- und Fußzeilen funktioniert die Seitenangabe nicht,
        ' dort wird als Seitenzahl "-1" ermittelt:
        If Fehlerseitenzahl < 1 Then
          Select Case Rng1.StoryType
            Case 6, 7, 10
              Fehlerseitenzahlangabe = "Kopfzeile" & Sectionangabe
            Case 8, 9, 11
              Fehlerseitenzahlangabe = "Fußzeile" & Sectionangabe
            Case Else
              Fehlerseitenzahlangabe = "unbekannter Dokumentkomponente" & Sectionangabe
          End Select
        Else
          Fehlerseitenzahlangabe = "Seite " & Fehlerseitenzahl & ":"
        End If
        '''''''
        Fehlermeldung = Fehlermeldung _
           & "Fehler " & Fehlernummer & " auf " & Fehlerseitenzahlangabe _
           & " """ & afield.Result & _
            """ bzw.  ""{" & afield.Code & "}""." & vbCrLf
          If Fehlernummer = 1 Then Set Rng2 = Selection.Range
      End If
    End If
  "Next i
  Next afield
End Sub
```

5 Hyperlinkprüfung

Für die Prüfung einer Datei mit Hunderten von Hyperlinks suchte ich im Internet nach passenden Makros. Ich fand viele Beispiele, aber nur zwei brauchbare Prinzipien. Durch deren Kombination und Erweiterung entwickelte ich das folgende Makro.

Das hier entwickelte Makro prüft:
a) Hyperlinks ins Internet
b) bei E-Mails die Domainnamen hinter dem @-("ät")-Zeichen als Hyperlink,
c) Links auf den eigenen Computer bzw. auf das eigene Computer-Netzwerk.

Der Zeitaufwand bei der Prüfung von Hyperlinks ins Internet ist teilweise erheblich, wobei das Problem am Internet liegt, nicht am Makro. Der Extremwert war bei mir etwa 7 Sekunden bei Abfrage je Datei im Internet (auch bei schneller Internetverbindung).

Die fehlerhaften Links werden rot markiert und in einem Protokoll aufgelistet. Bei erkannten Weiterleitungen wird stattdessen die Fundstelle rosa markiert und die Weiterleitungsadresse im Protokoll ergänzt.

Das Protokoll erfolgt als MsgBox und auf Wunsch oder bei zu vielen Fundstellen als separate Datei.

Mit einer abgewandelten Funktion werden jetzt auch Hyperlinks geprüft, die mit "ftp:" beginnen.

Ist nichts markiert, wird **das gesamte Dokument** einschließlich Fußnoten, Kopfzeilen usw. nach Hyperlinks durchsucht (neu Mai 2018).

Sind im Dokument die Hyperlinks deaktiviert, hilft hierfür das im Abschnitt 5.2 beschriebene Makro. Damit lassen sich in allen Dokumentkomponenten die ins Internet führenden Hyperlinks aktivieren.

Offene Probleme:
Weiterleitungen werden bei fälschlicher Rückmeldung von "Status 200" nur anhand einiger Schlüsselwörter wie "*REFRESH*", "*is obsolete*", "*Seite nicht gefunden*" und "*page not found*" erkannt. Es kann also sein, dass Hyperlinks zu falschen oder veralteten Unterseiten im Internet von meinem Makro nicht als fehlerhaft bemerkt werden, wenn der Domain-Inhaber auf eine andere Unterseite umleitet und dabei trotzdem den "Status 200" sendet.

Manchmal werden auch Hyperlinks als fehlerhaft gemeldet, die sich dann als richtig erweisen, etwa bei E-Mails bei name@gmx.de, weil die Webseite den Linkstatus "0" zurückgab, obwohl sie erreichbar ist.

Fehlermeldung gab es in seltenen Fällen, wenn am Ende des Hyperlinks ein Backslash stand oder auch fehlte.

Beginnt eine Internet-Adresse inzwischen mit https://... (sichere Verbindung), gibt es manchmal eine Fehlermeldung, wenn nach "http://…" ohne das "s" gesucht wird. Hier wäre vielleicht eine Makro-Ergänzung sinnvoll mit einer zusätzlichen Markierfarbe für diese Fälle.

5.1 Das komplette Makromodul "Hyperlinkprüfung"

' Achtung: Es werden nur aktive Hyperlinks geprüft!

```
' Makromodul "Hyperlinkprüfung"
'
Option Explicit

Private strFehlermeldung As String
Private IntFehlernummer As Integer
Private strWeiterleitung As String
Private N As Integer          ' laufende Nummer für zu prüfenden Hyperlink ins Internet.
Private k As Integer          ' Gesamtanzahl aller zu prüfender Hyperlinks ins Internet.
Private IntRosamarkierung As Integer
Private L As Integer          ' Anzahl der fehlerhaften Hyperlinks.
Private j As Integer          ' Anzahl der E-Mail-Adressen.
Private Seitenzahlanfang As Integer   ' Anfang eines Sections
Private Seitenzahlende As Integer     ' Ende eines Sections
' ******************************


Sub a3_Hyperlinkprüfung()
' (Rahmenprozedur)
' H. Chris Gast, 2011, 2012, 2015, 20. Januar 2018, 5. bis 8. + 23. Mai 2018.
' http://www.siebener-kurier.de/chris-aufsaetze
'
' Mit diesem Makro werden alle markierten Hyperlinks geprüft, auch Links auf andere
' Dateien desselben Computers bzw. Netzwerkes.
'
' Ist nichts markiert, wird das gesamte Dokument einschließlich Fußnoten, Kopfzeilen usw. geprüft.
'
' Fehlerhafte Hyperlinks werden rot markiert,
' bei erkannter Weiterleitung wird rosa markiert.
' Erreichbare Hyperlinks werden (zu Testzwecken) grün markiert.
'
' Ausgabe der Fehlermeldung als MsgBox bzw. in separate Datei auf Wunsch
' oder bei zu vielen Fehlern.
' Man kann diese Fehler anschließend anhand der Seitenangaben in der Fehlermeldung finden,
' bei Kopfzeilen und Fußzeilen wird der Abschnitt ('Section') mit Seitenzahlbereich angegeben.
'
' Für Hyperlinks, die mit "ftp:" beginnen, wird eine andere Prozedur benötigt.
'
' Offenes Problem: Viele Weiterleitungen werden nicht erkannt.
' ***************************************
Dim IntAbfrage As Integer
' ***************************************
'  Wenn nichts markiert ist, im ganzen Dokument prüfen (einschließlich Fußnoten, Kopfzeilen usw.).
  If Selection.Type < 2 Then
  '  Sicherheitshalber abfragen, ob man das ganze Dokument prüfen will:
    IntAbfrage = MsgBox("Wollen Sie im ganzen Dokument Hyperlinks prüfen?", _
                 vbYesNoCancel + vbDefaultButton2, "Alles Grün?")
    ' Bei "Ja" kommt "6" heraus, bei "Nein" kommt "7" heraus, bei Abbrechen eine "2".
    If IntAbfrage <> 6 Then
      Exit Sub
    Else
      Call Hyperlinksprüfung_alle_Storyranges
    End If
  Else
    ' nur den markierten Bereich prüfen.
    Call Hyperlinksprüfung_Selection
  End If
End Sub
```

```
' ----------------------------------------------------------------------------------------------------------------

Private Sub Hyperlinksprüfung_Selection()
' H. Chris Gast, 2011, 2012, 2015, 20. Januar 2018, 6. bis 23. Mai 2018.
' Mit diesem Makro werden alle Hyperlinks in der Markierung geprüft,
' auch Links auf andere Dateien desselben Computers bzw. Netzwerkes.
Dim i As Integer  ' Section
Dim k As Integer
Dim IntAbfrage As Integer
Dim Zeitdauer As Single
Dim para As Paragraph
Dim Zeit1 As Single
Dim Zeit2 As Single
Dim strMsg As String
Dim strAnzeige As String
Dim strDateiname As String
Dim StrErgebnis As String
Dim Rng0 As Range  ' Markierung bei Start des Makros
Dim Rng1 As Range
Zeitdauer = 0
N = 0
k = 0
L = 0
j = 0
IntRosamarkierung = 0
strFehlermeldung = ""
'  ***************************************
'  Markierung merken:
  Set Rng0 = Selection.Range
'  ***************************************
'  Vorab prüfen, ob und wieviel Hyperlinks in der Markierung enthalten sind:
  '' K = Selection.Range.Hyperlinks.Count
  'Zählte fälschlich Inhaltsverzeichnis mit, wenn dort Schalter "\h" gesetzt ist.
  ' Abhilfe: In der Feldcode-Ansicht nach dem Wort "HYPERLINK" suchen.
  'Damit der Bildschirm bei der Umschaltung auf Feldcode-Ansicht und zurück nicht
  'flattert, wird die Bildschirm-Aktualisierung währenddessen unterdrückt.
  Application.ScreenUpdating = False
  ActiveWindow.View.ShowFieldCodes = True
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "HYPERLINK"
    .Replacement.Text = ""
    .Forward = True
    .Format = False
    .MatchCase = True
    .MatchWholeWord = True
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
    Do While .Execute      ' (tue, solange etwas gefunden wird)
      If Selection.Start > Rng0.End Then Exit Do
      k = k + 1
      Selection.Collapse Direction:=wdCollapseEnd
    Loop
  End With
  ActiveWindow.View.ShowFieldCodes = False
  Rng0.Select
  Application.ScreenUpdating = True
'  ***************************************
  If k = 0 Then
```

```
    MsgBox "Der Text enthält keine Hyperlinks!", vbCritical
    End
  ElseIf k > 0 Then
  '  Wenn Internet-Links in der Markierung sind, vorab mit korrektem Hyperlink prüfen,
  '  ob überhaupt eine funktionierende Internetverbindung besteht:
    If Fkt_URL_existiert("https://de.wikipedia.org") = False Then
      MsgBox "Keine Internetverbindung vorhanden!" _
      & vbCrLf _
      & "Es werden deshalb nur Hyperlinks innerhalb des Computers geprüft.", vbCritical
     '' End
      StatusBar = "     Das Makro läuft, es durchsucht aber nur Hyperlinks innerhalb des Computers."
    End If
    Select Case k
      Case 1
        strMsg = "Bitte warten. Es wird 1 Hyperlink geprüft."
      Case 1 To 5
        strMsg = "Bitte warten. Es werden " & k & " Hyperlinks geprüft."
      Case Is > 5
        strMsg = "Bitte warten. Es werden " & k & " Hyperlinks geprüft." _
        & vbCrLf & "Die Rechenzeit kann entsprechend lang dauern!"
    End Select
    Call MsgFenster(strMsg, "Makro läuft.")
    StatusBar = "     " & strMsg
  End If
'  ************************************
'  Lästige Nachfrage(Warnmeldung) unterdrücken
'  (liefert leider trotzdem Warnmeldungen):
  Application.DisplayAlerts = wdAlertsNone
'  ******************************
'  Zeitmessen für die Rechendauer:
  Zeit1 = Timer
'  ************************************
'  Die eigentliche Prüfung:
  Call Hyperlink_Prüfung_Range(Selection.Range)
'  ************************************
'  Seitenzahlenbereiche der "Sections" ermitteln, wenn dort Fehler in Kopf- oder Fußzeilen:
  If ActiveDocument.Sections.Count > 1 Then
    If InStr(strFehlermeldung, "Kopfzeile") > 0 _
    Or InStr(strFehlermeldung, "Fußzeile") > 0 Then
      For i = 1 To ActiveDocument.Sections.Count
        Set Rng1 = ActiveDocument.Sections(i).Range
        Seitenzahlende = ActiveDocument.Sections(i).Range.Information(wdActiveEndPageNumber)
        Rng1.Collapse Direction:=wdCollapseStart
        Seitenzahlanfang = Rng1.Information(wdActiveEndPageNumber)
        strFehlermeldung = strFehlermeldung & vbCrLf _
        & "Abschnitt ('Section') " & i & " beginnt auf Seite " & Seitenzahlanfang _
        & " und endet auf Seite " & Seitenzahlende & "."
      Next i
      strFehlermeldung = strFehlermeldung & vbCrLf
      ' Gibt es keine Sections, braucht "in Section 1" nicht angegeben zu werden:
      If ActiveDocument.Sections.Count = 1 Then
        strFehlermeldung = Replace(strFehlermeldung, " in 'Section' 1", "")
      End If
    End If
  End If
'  ********************************************************************
'  Zeitmessen für die Rechendauer:
    DoEvents
    Zeit2 = Timer
    Zeitdauer = Zeit2 - Zeit1
    Zeitdauer = Round(Zeitdauer, 2)
'  ************************************
```

```
  ' kurze Warteschleife  für Word 2010:
  Dim tt As Single
    tt = Timer
    Do While Timer < tt + 0.1  'Sekunden
      DoEvents
    Loop
'  ************************************
'  Ausgabe des Ergebnisses in MsgBox und/oder in separate Datei:
  StatusBar = "     Auswertung wird vorbereitet"
  If j > 0 Then StrErgebnis = "Bei E-Mail-Adressen wurde nur die Domain geprüft (" & j & "x)." _
         & vbCrLf & vbCrLf
  If L = 0 And IntRosamarkierung = 0 Then
    ' keine Fehler:
    If k = 1 Then
        StrErgebnis = StrErgebnis & "Der geprüfte Hyperlink ist o.k." ' & vbCrLf
    ElseIf k > 1 Then
        StrErgebnis = StrErgebnis & "Die " & k & " geprüften Hyperlinks sind o.k." & vbCrLf
    End If
  ElseIf L = 1 And IntRosamarkierung = 0 Then
    ' einmal rot, kein Rosa:
    StrErgebnis = StrErgebnis & "Ein Hyperlink ist nicht erreichbar, siehe rote Markierung"
  ElseIf L > 1 And IntRosamarkierung = 0 Then
    ' mehrmals rot, kein Rosa:
    StrErgebnis = StrErgebnis & L & " Hyperlinks sind nicht erreichbar, siehe rote Markierungen."
  ElseIf L = 0 And IntRosamarkierung > 0 Then
    StrErgebnis = StrErgebnis & IntRosamarkierung _
    & " Hyperlink(s) hat/haben eine Weiterleitung oder war/waren nicht prüfbar (rosa" _
    & ChrW(160) & "markiert)."
  Else
    ' rot und rosa vorhanden:
    StrErgebnis = StrErgebnis & L & " Hyperlink(s) ist/sind nicht erreichbar (rot markiert)," _
    & vbCrLf _
    & IntRosamarkierung _
    & " Hyperlink(s) hat/haben eine Weiterleitung oder war/waren nicht prüfbar (rosa" _
    & ChrW(160) & "markiert)."
  End If
  strFehlermeldung = StrErgebnis & vbCrLf & vbCrLf & strFehlermeldung
  DoEvents
'  ************************************
  '' Debug.Print strFehlermeldung
'  Fehlermeldung um Rechenzeit ergänzen:
  strFehlermeldung = strFehlermeldung & vbCrLf _
        & "Die Rechenzeit der Hyperlink-Prüfung war " & Zeitdauer & " Sekunden."
'  ************************************
  StatusBar = "     Hyperlinkprüfung ist abgeschlossen."
  Rng0.Select
  DoEvents
'  ************************************
'  Bei zuviel Fehlern passen nicht mehr alle Zeilen in eine Messagbox.
  If L > 6 Then
    ' Zuviele Fehler für eine Messagebox, Ausgabe aller Fehler als separate Datei.
    IntAbfrage = 6
  Else
     '' "Fehlerhafte Hyperlinks: " & vbCrLf &
    strAnzeige = strFehlermeldung _
            & vbCrLf _
             & "Wollen Sie das Ergebnis in einer separaten Datei?"
    IntAbfrage = MsgBox(strAnzeige, vbYesNoCancel + vbDefaultButton2, _
           "Prüfung der Hyperlinks")
     ' Bei "Ja" kommt "6" heraus, bei "Nein" kommt "7" heraus, bei Abbrechen eine "2".
  End If
'  ************************************
```

```
' Alles zurückstellen:
  ActiveWindow.View.ShowFieldCodes = False
  Rng0.Select
  Application.ScreenUpdating = True
  Application.DisplayStatusBar = True
  Application.DisplayAlerts = wdAlertsAll
' ************************************
' Ausgabe des Protokolls in einer neuen Datei:
  If IntAbfrage = 6 Then
    strDateiname = ActiveDocument.Name
    ' Anzeige in separater Datei:
    Documents.Add DocumentType:=wdNewBlankDocument
    ' falls in der Normal.dot schon Name und Datum vorgesehen sind, dieses überschreiben:
    Selection.EndKey Unit:=wdStory, Extend:=True
    Call Namen_und_Datum_ergaenzen
    ' Absatzabstand auf Null stellen:
    With Selection.ParagraphFormat
      .SpaceBefore = 0
      .SpaceAfter = 6
      .Alignment = wdAlignParagraphLeft
    End With
    Selection.Font.Size = 12
    Selection.TypeParagraph
    Selection.Font.Bold = True
    If L > 0 Then
      Selection.TypeText Text:="Fehlerhafte Hyperlinks in """ & strDateiname & """"
    Else
      Selection.TypeText Text:="Hyperlinkprüfung in """ & strDateiname & """"
    End If
    Selection.Font.Bold = False
    Selection.TypeParagraph
    With Selection.ParagraphFormat
      .SpaceBefore = 0
      .SpaceAfter = 0
      .Alignment = wdAlignParagraphLeft
      .LeftIndent = CentimetersToPoints(1.5)
      .FirstLineIndent = CentimetersToPoints(-1.5)
    End With
    Selection.Font.Size = 11
    Selection.TypeText Text:=strFehlermeldung
  End If
End Sub
' ---------------------------------------------------------------------------------------------------------


Private Sub Hyperlinksprüfung_alle_Storyranges()
' H. Chris Gast, 2011, 2012, 2015, 20. Januar 2018, 5. bis 8. + 23. Mai 2018.
' http://www.siebener-kurier.de/chris-aufsaetze
' Mit diesem Makro werden alle Hyperlinks in einem Word-Dokument geprüft, auch Links auf andere
' Dateien desselben Computers bzw. Netzwerkes.
' Dieses Makro prüft dabei die Hyperlinks in allen StoryRanges, auch in Fußnoten, Fußzeilen
' und Kopfzeilen, Textfeldern, (ausgenommen Hyperlinks in Textfeldern von  Fußzeilen,
' was wohl kaum vorkommt).
'
' Kleines Problem: Es werden auch Fehler in nicht anzeigbaren Kopf- und Fußzeilen aufgelistet.
' ****************************
Dim Änderungsverfolgung As Boolean
Dim Ansichtstyp1 As Integer
Dim Rng0 As Range  ' Markierung bei Start des Makros
Dim Rng1 As Range  ' Bereich der jeweiligen Sections
Dim Zeit1 As Single
Dim Zeit2 As Single
```

```
Dim Zeitdauer As Single
Dim i As Integer
"Dim K As Integer   ' Anzahl der Hyperlinks ins Internet
Dim strMsg As String
IntFehlernummer = 0
strFehlermeldung = ""
Dim IntAbfrage As Integer
Dim IntSection As Integer
Dim Anzeigetext As String
Dim strDateiname As String
Dim Seitenzahlanfang As Integer
Dim Seitenzahlende As Integer
Dim rngStory As Range        ' Dokumentkomponente
Dim rngSection As Section
Dim rngHeaderFooter As HeaderFooter
Dim shpShape As Shape        ' Textfelder und anderes Shapes in den Kopf- und Fußzeilen.
Dim BL_Textfeld As Boolean   ' Für Suche nach Textfeldern in Kopf- oder Fußzeile
Dim StrErgebnis As String
IntRosamarkierung = 0
N = 0
k = 0
L = 0
j = 0
IntRosamarkierung = 0
' ********************************
' Beim Aktualisieren von Feldern und Inhaltsverzeichnissen sollte die Änderungsverfolgung
' abgestellt sein:
  Änderungsverfolgung = ActiveDocument.TrackRevisions
  ' MsgBox "Die Änderungsverfolgung ist " & Änderungsverfolgung
  ActiveDocument.TrackRevisions = False
' ******************************
' Zeitmessen für die Rechendauer:
  Zeit1 = Timer
' ******************************
' Lästige Nachfrage(Warnmeldung) in Word 2000 beim Aktualisieren aller Felder unterdrücken:
  Application.DisplayAlerts = wdAlertsNone
' ******************************
' Zuerst die vorhandene Bildschirmansicht speichern:
  Ansichtstyp1 = (ActiveWindow.View.Type)
' Bildschirm zur Beschleunigung auf "Normalansicht" stellen, wenn dies nicht schon der Fall ist:
  If ActiveWindow.View.Type = wdNormalView = False Then
    ActiveWindow.View.Type = wdNormalView
  End If
' Bildschirmaktualisierung während des Programms zur Beschleunigung abstellen.
  StatusBar = "Bitte warten - das Makro läuft."   ' Geht nicht in Word 2010.
  Application.ScreenUpdating = False
  On Error Resume Next  ' 7. Jan. 2015
  Options.Pagination = False
  On Error GoTo 0
' ******************************
' Anfangsmakierung speichern:
  Set Rng0 = Selection.Range
' ********************************************************************************
  k = Fkt_Hyperlinks_zählen
  If k = 0 Then
    MsgBox "Die Datei enthält keine Hyperlinks!"
    Exit Sub
  ElseIf k > 0 Then
  ' Wenn Internet-Links in der Markierung sind, vorab mit korrektem Hyperlink prüfen,
  ' ob überhaupt eine funktionierende Internetverbindung besteht:
    If Fkt_URL_existiert("https://de.wikipedia.org") = False Then
      MsgBox "Keine Internetverbindung vorhanden!" _
```

```
        & vbCrLf _
        & "Das Makro wird abgebrochen.", vbCritical
        End
      End If
      Select Case k
        Case 1
          strMsg = "Bitte warten. Es wird 1 Hyperlink geprüft."
        Case 1 To 5
          strMsg = "Bitte warten. Es werden " & k & " Hyperlinks geprüft."
        Case Is > 5
          strMsg = "Bitte warten. Es werden " & k & " Hyperlinks geprüft." _
          & vbCrLf & "Die Rechenzeit kann entsprechend lang dauern!"
      End Select
      Call MsgFenster(strMsg, "Makro läuft.")
      StatusBar = "     " & strMsg
    End If
' ******************************************************************************
' Alle Komponenten des Dokuments durchsuchen:
' ******************************************************************************
' Alle Dokumentkomponenten außer Fuß- und Kopfzeilen:
  For Each rngStory In ActiveDocument.StoryRanges
    If rngStory.StoryType < 6 Then
      '(die Trennlinie über Fußnoten soll nicht erfasst werden)
      Call Hyperlink_Prüfung_Range(rngStory)
    End If
  Next
' ****************************************
' Kopf und Fußzeilen (in allen Sections):
  For Each rngSection In ActiveDocument.Sections
    rngSection.Range.Select
    IntSection = Selection.Sections(1).Index
    ' Kopfzeilen:
    For Each rngHeaderFooter In rngSection.Headers
      If rngHeaderFooter.LinkToPrevious = False Then
        Call Hyperlink_Prüfung_Range(rngHeaderFooter.Range)
        ' ************
        ' Suche nach Textfeldern in Kopf-/Fußzeilen nur ein einziges Mal durchführen!
        ' (sie werden sonst bei jedem rngHeaderFooter gefunden!)
        If BL_Textfeld = False Then
          For Each shpShape In rngHeaderFooter.Shapes
            If shpShape.Type = msoTextBox Then
              Call Hyperlink_Prüfung_Range(shpShape.TextFrame.TextRange)
            End If
          Next
          BL_Textfeld = True
        End If
        ' ************
      End If
    Next rngHeaderFooter
    ' *********
    ' Fußzeilen:
    For Each rngHeaderFooter In rngSection.Footers
      If rngHeaderFooter.LinkToPrevious = False Then
        Call Hyperlink_Prüfung_Range(rngHeaderFooter.Range)
      End If
    Next rngHeaderFooter
  Next rngSection
' ************************************************************************
' Seitenzahlenbereiche der "Sections" ermitteln, wenn dort Fehler in Kopf- oder Fußzeilen:
  If ActiveDocument.Sections.Count > 1 Then
    If InStr(strFehlermeldung, "Kopfzeile") > 0 _
    Or InStr(strFehlermeldung, "Fußzeile") > 0 Then
```

```
      For i = 1 To ActiveDocument.Sections.Count
        Set Rng1 = ActiveDocument.Sections(i).Range
        Seitenzahlende = ActiveDocument.Sections(i).Range.Information(wdActiveEndPageNumber)
        Rng1.Collapse Direction:=wdCollapseStart
        Seitenzahlanfang = Rng1.Information(wdActiveEndPageNumber)
        strFehlermeldung = strFehlermeldung & vbCrLf & "Abschnitt ('Section') " & i _
        & " beginnt auf Seite " & Seitenzahlanfang _
        & " und endet auf Seite " & Seitenzahlende & "."
      Next i
      strFehlermeldung = strFehlermeldung & vbCrLf
    End If
  End If
' ******************************************************************
' ursprüngliche Bildschirmansicht wiederherstellen
' und markierte Stelle wieder markieren:
' ******************************************************************
' In Seitenansicht gehen und ggf. das zweite Windows-Fenster schließen:
  If ActiveWindow.View.SplitSpecial = wdPaneNone Then
    ActiveWindow.ActivePane.View.Type = wdPrintView
  Else
    ActiveWindow.Panes(2).Close
    ActiveWindow.View.Type = wdPrintView
  End If
  '
  If ActiveWindow.ActivePane.View.Type = wdNormalView _
  Or ActiveWindow.ActivePane.View.Type = wdOutlineView Then
    ActiveWindow.ActivePane.View.Type = wdPrintView
  End If
  ' In die Hauptkomponente gehen:
  ActiveWindow.ActivePane.View.SeekView = wdSeekMainDocument
  ActiveWindow.ActivePane.View.Type = wdPrintView
  '
  ' Ursprüngliche Seitenansichts-Art wiederherstellen:
  ActiveWindow.View.Type = Ansichtstyp1
  Rng0.Select    ' Funktioniert nur in der Dokument-Hauptkomponente.
  StatusBar = False
  ActiveWindow.View.ShowFieldCodes = False
  Application.ScreenUpdating = True
  Application.DisplayStatusBar = True
  Application.DisplayAlerts = wdAlertsAll
' ********************************************************
' Zeitmessen für die Rechendauer:
      Zeit2 = Timer
      Zeitdauer = Zeit2 - Zeit1
      Zeitdauer = Round(Zeitdauer, 2)
' ********************************************************
' Ausgabe des Ergebnisses in MsgBox und/oder in separate Datei:
' ********************************************************
  StatusBar = "     Auswertung wird vorbereitet"
  If j > 0 Then StrErgebnis = "Bei E-Mail-Adressen wurde nur die Domain geprüft (" & j & "x)." _
        & vbCrLf & vbCrLf
  If L = 0 And IntRosamarkierung = 0 Then
    ' keine Fehler:
    If k = 1 Then
        StrErgebnis = StrErgebnis & "Der geprüfte Hyperlink ist o.k." ' & vbCrLf
    ElseIf k > 1 Then
        StrErgebnis = StrErgebnis & "Die " & k & " geprüften Hyperlinks sind o.k." & vbCrLf
    End If
  ElseIf L = 1 And IntRosamarkierung = 0 Then
    ' einmal rot, kein Rosa:
    StrErgebnis = StrErgebnis & "Ein Hyperlink ist nicht erreichbar, siehe rote Markierung"
  ElseIf L > 1 And IntRosamarkierung = 0 Then
```

```
    ' mehrmals rot, kein Rosa:
    StrErgebnis = StrErgebnis & L & " Hyperlinks sind nicht erreichbar, siehe rote Markierungen."
  ElseIf L = 0 And IntRosamarkierung > 0 Then
    StrErgebnis = StrErgebnis & IntRosamarkierung _
    & " Hyperlink(s) hat/haben eine Weiterleitung oder war/waren nicht prüfbar (rosa" _
    & ChrW(160) & "markiert)."
  Else
    ' rot und rosa vorhanden:
    StrErgebnis = StrErgebnis & L & " Hyperlink(s) ist/sind nicht erreichbar (rot markiert)," _
    & vbCrLf _
    & IntRosamarkierung _
    & " Hyperlink(s) hat/haben eine Weiterleitung oder war/waren nicht prüfbar (rosa" _
    & ChrW(160) & "markiert)."
  End If
  strFehlermeldung = StrErgebnis & vbCrLf & vbCrLf & strFehlermeldung
  DoEvents
' ************************************
  StatusBar = "     Hyperlinkprüfung ist abgeschlossen."
' ************************************
  strFehlermeldung = strFehlermeldung & vbCrLf _
        & "Die Rechenzeit der Hyperlink-Prüfung war " & Zeitdauer & " Sekunden."
  '  Bei zuviel Fehlern passen nicht mehr alle Zeilen in eine Messagbox.
  If L > 6 Then
    ' Zuviele Fehler für eine Messagebox, Ausgabe aller Fehler als separate Datei.
    IntAbfrage = 6
  Else
    Anzeigetext = strFehlermeldung & vbCrLf _
              & "Wollen Sie das Ergebnis in einer separaten Datei?"
    IntAbfrage = MsgBox(Anzeigetext, vbYesNoCancel + vbDefaultButton2, _
            "Prüfunge der Hyperlinks")
     ' Bei "Ja" kommt "6" heraus, bei "Nein" kommt "7" heraus, bei Abbrechen eine "2".
  End If
  If IntAbfrage = 6 Then
    strDateiname = ActiveDocument.Name
    ' Anzeige in separater Datei:
    Documents.Add DocumentType:=wdNewBlankDocument
    ' falls in der Normal.dot schon Name und Datum vorgesehen sind, dieses überschreiben:
    Selection.EndKey Unit:=wdStory, Extend:=True
    Call Namen_und_Datum_ergaenzen
    ' Absatzabstand auf Null stellen:
    With Selection.ParagraphFormat
      .SpaceBefore = 0
      .SpaceAfter = 6
      .Alignment = wdAlignParagraphLeft
    End With
    Selection.Font.Size = 12
    Selection.TypeParagraph
    Selection.Font.Bold = True
    If L > 0 Then
      Selection.TypeText Text:="Fehlerhafte Hyperlinks in """ & strDateiname & """"
    Else
      Selection.TypeText Text:="Hyperlinkprüfung in """ & strDateiname & """"
    End If
    Selection.Font.Bold = False
    Selection.TypeParagraph
    With Selection.ParagraphFormat
      .SpaceBefore = 0
      .SpaceAfter = 0
      .Alignment = wdAlignParagraphLeft
      .LeftIndent = CentimetersToPoints(1.5)
      .FirstLineIndent = CentimetersToPoints(-1.5)
    End With
```

```
    Selection.Font.Size = 11
    Selection.TypeText Text:=strFehlermeldung
  End If
'  ***********************************************************
' Änderungsverfolgung ggf. wieder einschalten:
  ActiveDocument.TrackRevisions = Änderungsverfolgung
End Sub
' ---------------------------------------------------------------------------------------------------------


Private Function Fkt_Hyperlinks_zählen() As Integer
' H. Chris Gast, 27. Januar 2012, neu: 5. + 6. Mai 2018.
'  Vorab prüfen, wieviel Hyperlinks im gesamten Dokument sind:
   '' i = Selection.Range.Hyperlinks.Count
' Damit der Bildschirm bei der Umschaltung auf Feldcode-Ansicht und zurück nicht
' flattert, wird die Bildschirm-Aktualisierung währenddessen unterdrückt:
  Application.ScreenUpdating = False
' ***************************************
Dim rngStory As Range    ' Dokumentkomponente
Dim i As Integer: i = 0
Dim hl As Hyperlink
Dim para As Paragraph
Dim rngHeaderFooter As HeaderFooter
Dim rngSection As Section
Dim shpShape As Shape        ' Textfelder und anderes Shapes in den Kopf- und Fußzeilen.
Dim BL_Textfeld As Boolean   ' Für Suche nach Textfeldern in Kopf- oder Fußzeile
'  ***************************************
   Application.ScreenUpdating = False
   ' With Options
   '   .Pagination = False ' Ergibt Fehlermeldung 8.3.2017.
   ' End With
' ***************************************
' Alle Komponenten des Dokuments durchsuchen.
' Zuerst alle Dokumentkomponenten außer Fuß- und Kopfzeilen durchsuchen:
  For Each rngStory In ActiveDocument.StoryRanges
     If rngStory.StoryType < 6 Then
     '(die Trennlinie über Fußnoten soll nicht erfasst werden)
        ActiveWindow.View.ShowFieldCodes = True
        For Each para In rngStory.Paragraphs
           ' Nur in Absätzen mit dem FeldCode "HYPERLINK" nach Hyperlinks suchen!
           ' Damit wird die fälschliche Erfassung des Inhaltsverzeichnisses vermieden.
           If InStr(para, "HYPERLINK ") > 0 Then
              i = i + para.Range.Hyperlinks.Count
           End If
        Next para
        ActiveWindow.View.ShowFieldCodes = False
     End If
  Next rngStory
' ***************************************
' Dann Kopf und Fußzeilen (in allen Sections) durchsuchen:
  For Each rngSection In ActiveDocument.Sections
     rngSection.Range.Select
     "IntSection = Selection.Sections(1).Index
     ' Kopfzeilen:
     For Each rngHeaderFooter In rngSection.Headers
       If rngHeaderFooter.LinkToPrevious = False Then
          i = i + rngHeaderFooter.Range.Hyperlinks.Count
          ' ***********
           ' Suche nach Textfeldern in Kopf-/Fußzeilen nur ein einziges Mal durchführen!
           ' (sie werden sonst bei jedem rngHeaderFooter gefunden!)
           If BL_Textfeld = False Then
              For Each shpShape In rngHeaderFooter.Shapes
```

```
            If shpShape.Type = msoTextBox Then
              i = i + shpShape.TextFrame.TextRange.Hyperlinks.Count
            End If
          Next
          BL_Textfeld = True
        End If
        ' **********
      End If
    Next rngHeaderFooter
    ' *********
    ' Fußzeilen:
    For Each rngHeaderFooter In rngSection.Footers
      If rngHeaderFooter.LinkToPrevious = False Then
        i = i + rngHeaderFooter.Range.Hyperlinks.Count
      End If
    Next rngHeaderFooter
  Next rngSection
  ' *********************
' In die Hauptkomponente zurückkehren:
  ActiveWindow.ActivePane.View.Type = wdPrintView
  ActiveWindow.ActivePane.View.SeekView = wdSeekMainDocument
' Bildschirm aktualisieren:
  Application.ScreenUpdating = True
  ' *********************
  Fkt_Hyperlinks_zählen = i
End Function
' ---------------------------------------------------------------------------------------------------------------


Private Sub Hyperlink_Prüfung_Range(Rng1 As Range)
' H. Chris Gast, 2011, 2012, 25. August 2015, 20. Januar 2018, 5. + 7. + 22. + 23. Mai 2018.
' Die eigentliche Hyperlink-Prüfung.
' Die Feldfunktion wird als "falsch" gewertet, wenn sie nur mit "www" beginnt.
Dim hl As Hyperlink
Dim para As Paragraph
Dim IntFehlerseitenzahl As Integer
Dim strFehlerseitenzahl As String
Dim strMaildomain As String
Dim IntSection As Integer
Dim strSectionangabe As String
' N ist laufende Nummer.
' L ist die Anzahl der Fehler (rot)
' IntRosamarkierung die Anzahl der Weiterleitung/nicht-erreichbaren Links (rosa).
' ************************************
  ActiveWindow.View.ShowFieldCodes = True
  For Each hl In Rng1.Hyperlinks
    StatusBar = "    Bitte warten, das Makro läuft. Hyperlink " & N + 1 & " von " & k & " wird geprüft."
    DoEvents
    hl.Range.Select
    On Error GoTo Zeile_Fehler
    ' *********************
    ' Wenn der Hyperlink fehlerhaft nur "www." enthält:
    If Left(hl.Address, 4) = "www." Then
      ' Für die Prüfung reparieren (ergänzt 22. Mai 2018):
      hl.Address = Replace(hl.Address, "www.", "http://www.")
      ' (Das nützt leider nichts, wenn es eigentlich "https" heißen müsste)
    End If
    ' **********************
    If Left(hl.Address, 4) = "http" Then
      ' Hyperlink im Internet (hier funktioniert die "Dir"-Funktion nicht):
      N = N + 1
      If Fkt_URL_existiert(hl.Address) = False Then
```

```
        "L = L + 1
        hl.Range.Select
        IntFehlerseitenzahl = hl.Range.Information(wdActiveEndPageNumber)
        ' ****************
        ' bei Kopf- und Fußzeilen funktioniert die Seitenangabe nicht:
        If Selection.Range.StoryType > 2 Then
          If ActiveDocument.Sections.Count > 1 Then
            IntSection = Selection.Range.Sections(1).Index
            strSectionangabe = " in 'Section' " & IntSection & ": "
          Else
            strSectionangabe = " auf Seite 1 ff: "
          End If
          Select Case Rng1.StoryType
            Case 6, 7, 10
              strFehlerseitenzahl = "Kopfzeile" & strSectionangabe
            Case 8, 9, 11
              strFehlerseitenzahl = "Fußzeile" & strSectionangabe
            Case 5
              strFehlerseitenzahl = "Textfeld auf Seite " & IntFehlerseitenzahl & " ff: "
            Case Else
              strFehlerseitenzahl = "unbekannte Dokumentkomponente" & strSectionangabe
          End Select
        Else
          strFehlerseitenzahl = "Seite " & IntFehlerseitenzahl & ": "
        End If
        ' ******
        strFehlermeldung = strFehlermeldung _
        & strFehlerseitenzahl & hl.Address & vbCrLf
        ' ****************
        ' Wenn es eine Weiterleitung gibt, dieses in die Fehlermeldung aufnehmen.
        If strWeiterleitung <> "" Then
          IntRosamarkierung = IntRosamarkierung + 1
          Selection.Range.HighlightColorIndex = wdPink
          Selection.Collapse Direction:=wdCollapseEnd
          Selection.TypeText Text:=" "
          strFehlermeldung = strFehlermeldung _
          & "      (Es besteht eine Weiterleitung nach " & strWeiterleitung & ") " & vbCrLf
        Else
          ' Webseite nicht erreicht:
Zeile_Fehler:
          Selection.Range.HighlightColorIndex = wdRed
          L = L + 1
          ' Normale Fehlermeldung (unverändert):
        End If
        ' ******
      Else
        hl.Range.HighlightColorIndex = wdBrightGreen
      End If
      ' *********
    ElseIf Left(hl.Address, 7) = "mailto:" Then
      ' E-Mail-Adresse (hier wird nur der Domainname geprüft):
      N = N + 1
      j = j + 1
      strMaildomain = Mid(hl.Address, InStr(hl.Address, "@") + 1)
      If Right(LCase(hl.Address), 7) = "@gmx.de" Then
        ' Nichts tun! Dies hatte 2012 stets Fehler ergeben,
        ' obwohl diese Adresse als E-Mail existiert.
      ElseIf Fkt_URL_existiert("http://www." & strMaildomain) = False _
      And Fkt_URL_existiert("https://www." & strMaildomain) = False Then
        ' den seltenen Fall von E-Mails auf einer Seite ohne "www" testen:
        If Fkt_URL_existiert("http://" & strMaildomain) = False Then
          L = L + 1
```

```
        IntFehlerseitenzahl = hl.Range.Information(wdActiveEndPageNumber)
        ' ************
        ' bei Kopf- und Fußzeilen reicht die Seitenangabe nicht aus:
        If Selection.Range.StoryType > 2 Then
          If ActiveDocument.Sections.Count > 1 Then
            IntSection = Selection.Range.Sections(1).Index
            strSectionangabe = " in 'Section' " & IntSection & ": "
          Else
            strSectionangabe = " auf Seite 1 ff: "
          End If
          Select Case Rng1.StoryType
            Case 6, 7, 10
              strFehlerseitenzahl = "Kopfzeile" & strSectionangabe
            Case 8, 9, 11
              strFehlerseitenzahl = "Fußzeile" & strSectionangabe
            Case 5
               strFehlerseitenzahl = "Textfeld auf Seite " & IntFehlerseitenzahl & " ff: "
            Case Else
              strFehlerseitenzahl = "unbekannte Dokumentkomponente" & strSectionangabe
           End Select
        Else
           strFehlerseitenzahl = "Seite " & IntFehlerseitenzahl & ": "
        End If
        ' ************
        strFehlermeldung = strFehlermeldung _
          & strFehlerseitenzahl & hl.Address & vbCrLf
        hl.Range.Select
        Selection.Range.HighlightColorIndex = wdRed
        Selection.Collapse Direction:=wdCollapseEnd
      End If
    Else
      hl.Range.HighlightColorIndex = wdBrightGreen
    End If
  ElseIf Left(hl.Address, 4) = "ftp:" Then
    N = N + 1
    ' Separate Funktion nötig:
    If Fkt_FTP_URL_Existiert(hl.Address) = False Then
      L = L + 1
      hl.Range.Select
      Selection.Range.HighlightColorIndex = wdRed
      '' IntRosamarkierung = IntRosamarkierung + 1
      IntFehlerseitenzahl = hl.Range.Information(wdActiveEndPageNumber)
      Selection.Collapse Direction:=wdCollapseEnd
      ' ******************
      ' bei Kopf- und Fußzeilen reicht die Seitenangabe nicht aus:
       If Selection.Range.StoryType > 2 Then
         If ActiveDocument.Sections.Count > 1 Then
           IntSection = Selection.Range.Sections(1).Index
           strSectionangabe = " in 'Section' " & IntSection & ": "
         Else
           strSectionangabe = " auf Seite 1 ff: "
         End If
         Select Case Rng1.StoryType
          Case 6, 7, 10
            strFehlerseitenzahl = "Kopfzeile" & strSectionangabe
          Case 8, 9, 11
            strFehlerseitenzahl = "Fußzeile" & strSectionangabe
          Case 5
             strFehlerseitenzahl = "Textfeld auf Seite " & IntFehlerseitenzahl & " ff: "
          Case Else
            strFehlerseitenzahl = "unbekannte Dokumentkomponente" & strSectionangabe
         End Select
```

```
        Else
          strFehlerseitenzahl = "Seite " & IntFehlerseitenzahl & ": "
        End If
        ' ******************
        strFehlermeldung = strFehlermeldung _
          & strFehlerseitenzahl & hl.Address & vbCrLf
        hl.Range.Select
        Selection.Range.HighlightColorIndex = wdRed
      Else
        hl.Range.HighlightColorIndex = wdBrightGreen
      End If
      Selection.Collapse Direction:=wdCollapseEnd
    Else
      ' Hyperlinks zu anderen Dateien auf dem Computer bzw. Netzwerk.
      ' Hier muss der "Dir-Befehl" genutzt werden, der aber eine absolute Adresse braucht.
      On Error Resume Next  '  (Bei der folgenden Abfrage entstehen leicht Fehler).
      If Dir(hl.Address) = "" Then
        '  Intranet-Hyperlink ist fehlerhaft oder nicht absolut, deshalb auch folgendes testen:
        If Dir(ActiveDocument.Path & "\" & hl.Address) = "" Then
          L = L + 1
          IntFehlerseitenzahl = hl.Range.Information(wdActiveEndPageNumber)
          ' *************
          ' bei Kopf- und Fußzeilen reicht die Seitenangabe nicht aus:
           If Selection.Range.StoryType > 2 Then
             If ActiveDocument.Sections.Count > 1 Then
               IntSection = Selection.Range.Sections(1).Index
               strSectionangabe = " in 'Section' " & IntSection & ": "
             Else
               strSectionangabe = " auf Seite 1 ff: "
             End If
             Select Case Rng1.StoryType
               Case 6, 7, 10
                 strFehlerseitenzahl = "Kopfzeile" & strSectionangabe
               Case 8, 9, 11
                 strFehlerseitenzahl = "Fußzeile" & strSectionangabe
               Case 5
                  strFehlerseitenzahl = "Textfeld auf Seite " & IntFehlerseitenzahl & " ff: "
               Case Else
                 strFehlerseitenzahl = "unbekannte Dokumentkomponente" & strSectionangabe
             End Select
           Else
             strFehlerseitenzahl = "Seite " & IntFehlerseitenzahl & ": "
           End If
          ' *************
          strFehlermeldung = strFehlermeldung _
            & strFehlerseitenzahl & hl.Address & vbCrLf
          hl.Range.Select
          Selection.Range.HighlightColorIndex = wdRed
          Selection.Collapse Direction:=wdCollapseEnd
        Else
          hl.Range.HighlightColorIndex = wdBrightGreen
        End If
      Else
        hl.Range.HighlightColorIndex = wdBrightGreen
      End If
      On Error GoTo 0   ' Normale Fehlerbehandlung
    End If
    On Error GoTo 0
  Next hl
  ActiveWindow.View.ShowFieldCodes = False
  DoEvents
  ''Debug.Print strFehlermeldung
```

```
End Sub
' ----------------------------------------------------------------------------------------------------------


Private Function Fkt_URL_existiert(sURL As String) As Boolean
' H. Chris Gast, Januar 2012, 16. September 2012.
' Prüft Hyperlinks ins Internet anhand "Status 200".
' Wenn Status = 200, dann prüfen, ob im Quelltext eine Weiterleitung enthalten ist.
' Gibt falsche Ergebnisse, wenn sURL kein richtiger Hyperlink ist, wenn etwa das "http://" fehlt.
'
' Verwendete Quellen (Downloads Januar 2012):
' http://www.herber.de/forum/archiv/1084to1088/t1084301.htm
' http://www.office-loesung.de/ftopic279289_0_0_asc.php
' und andere.
'
' Verbesserungsideen:
' a) Wenn letztes Zeichen in sURL ein "/", dann ohne dieses Zeichen prüfen?
' b) Wenn "http:" nicht erreichbar ist, dasselbe mit "https" prüfen
'    und andere Markierfarbe verwenden?
Dim objXMLHTTP As Object
Dim sBuffer As String: sBuffer = Space(500)  ' (Buffer für die herunterzuladende Datei)
Dim a1, a2, a3 As Integer
strWeiterleitung = ""
  '
  On Error Resume Next
  Set objXMLHTTP = CreateObject("MSXML2.XMLHTTP")
  objXMLHTTP.Open "GET", sURL, False
  objXMLHTTP.Send
  '
  If objXMLHTTP.Status = 200 Then
    Fkt_URL_existiert = True
    ' ***
    ' Auf Weiterleitungen prüfen durch Analyse des Anfangs des Webseiten-Quelltextes:
    sBuffer = objXMLHTTP.responseText
    sBuffer = Left(sBuffer, 500)         ' (Ergibt Anfang der Webseite)
    If InStr(sBuffer, "Error 404") > 0 Then
      Fkt_URL_existiert = False
    ElseIf InStr(sBuffer, "Seite nicht vorhanden") > 0 Then
      Fkt_URL_existiert = False
    ElseIf InStr(LCase(sBuffer), "not found") > 0 Then
      Fkt_URL_existiert = False
    ElseIf InStr(LCase(sBuffer), "not find") > 0 Then
      Fkt_URL_existiert = False
    ElseIf InStr(LCase(sBuffer), "is obsolete") > 0 Then
      Fkt_URL_existiert = False
    ElseIf InStr(LCase(sBuffer), "<meta http-equiv=""refresh") > 0 Then
      ' Weiterleitung ansehen:
      a1 = InStr(LCase(sBuffer), """refresh""")
      If InStr(LCase(Mid(sBuffer, a1)), "url=http://") > 0 _
      Or InStr(LCase(Mid(sBuffer, a1)), "url=https://") > 0 Then
        Fkt_URL_existiert = False
        a2 = a1 + InStr(Mid(sBuffer, a1), "URL=http://") + 3
        a3 = InStr(Mid(sBuffer, a2), """>") - 1
        strWeiterleitung = Mid(sBuffer, a2, a3)
      End If
    End If
    '***
  Else
    ' Status ungleich 200:
    Fkt_URL_existiert = False
  End If
  ' ***
```

```
  Set objXMLHTTP = Nothing
' ++++++++++++++++++++
'"Fehlerbehandlung:
" If Err.Number > 0 Then
"    MsgBox sURL & vbCr & "Fehler " & Err.Number & vbCr & Err.Description, _
"    vbOKOnly, "strFehlermeldung"
" End If
  On Error GoTo 0
End Function
' ---------------------------------------------------------------------------------------------------------


Private Function Fkt_FTP_URL_Existiert(strURL As String) As Boolean
' H. Chris Gast, 23. Mai 2018.
' Bei "ftp:" darf man nicht den Status abfragen (der wäre Null),
' sondern kann die Länge der zurückgegebenen Nachricht abfragen:
Dim objXML_Server As Object
"   If LCase(Left(strURL, 4)) <> "ftp:" Then
"      MsgBox "andere Funktion nehmen!"
"   End If
  Set objXML_Server = CreateObject("MSXML2.XMLHTTP")
'  ****
  objXML_Server.Open "get", strURL, False
  On Error Resume Next  ' war nötig!
  objXML_Server.Send
  Debug.Print "Fehlernummer = " & Err.Number
  "Debug.Print Left(objXML_Server.responseText, 100)
  ' bei "ftp:" => "%PDF-1.4..." , bei "http:" => "<!doctype html public..."
'  ***
  If Len(objXML_Server.responseText) > 20 Then
     Fkt_FTP_URL_Existiert = True
  Else
     Fkt_FTP_URL_Existiert = False
  End If
  Set objXML_Server = Nothing
 End Function
' ---------------------------------------------------------------------------------------------------------


Private Sub MsgFenster(strMeldung As String, strUeberschrift As String)
' MsgBox, die sich nach 1 Sekunde selbst schließt.
' aus: http://www.schreibbuero-richter.de/vba/MsgBox.html
' angepasst 17. Jan. 2012, 5. Mai 2018.
' Funktioniert in Word 2000 unzuverlässig.
Dim IntMeldung As Integer
Dim objShell As Object
Dim lngAnzeigezeit As Long
  lngAnzeigezeit = 1
On Error Resume Next
'  Prüfen, ob Windows Scripting Host installiert ist:
  Set objShell = CreateObject("WScript.Shell")
  If Not IsObject(objShell) Then
     'falls WSH nicht installiert ist, normale MsgBox nehmen:
     IntMeldung = MsgBox(strMeldung, vbInformation, strUeberschrift)
  Else
     ' objShell.PopUp strPrompt, lngAnzeigezeit, strUeberschrift, intSymbol 'z.B. vbInformation
     objShell.Popup strMeldung, lngAnzeigezeit, strUeberschrift, 0
     Set objShell = Nothing
  End If
On Error GoTo 0
End Sub
' ---------------------------------------------------------------------------------------------------------
```

```
Private Sub Namen_und_Datum_ergaenzen()
' H. Chris Gast, 13. + 14. August 2013, 20. Januar 2018, 11. Mai 2018.
' In dieser ausgelagerten Unterprozedur wird der in Word eingestellte Name
' und das Datum an der Stelle des Cursors eingefügt.
' Dabei werden für diese Zeile die Tabulatoren geändert.
Dim Satzspiegelbreite As String
Dim str1 As String
Dim j As Integer
'  ********
'  Das Datum soll rechtsbündig sein, unabhängig von vorhandenen Tabulatoren:
  Satzspiegelbreite = _
    Round(PointsToCentimeters(Selection.Sections(1).PageSetup.TextColumns(1).Width), 1)
  Selection.Expand Unit:=wdParagraph
  Selection.Paragraphs.TabStops.ClearAll
  Selection.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(Satzspiegelbreite), _
     Alignment:=wdAlignTabRight, Leader:=wdTabLeaderSpaces
'  ********
'  Zuerst der Name des jeweiligen Computer-Inhabers:
  '' Environ ("COMPUTERNAME") ' gibt leider nicht den Personennamen.
  '' Environ("username") ' gibt (bei mir) nur ein Nameskürzel.
  ' Name des Dokument-Inhabers abfragen:
  ' Selection.TypeText Text:=ActiveDocument.BuiltInDocumentProperties("Author").Value
  str1 = ActiveDocument.BuiltInDocumentProperties("Author").Value
  ''   ' Bei fremden Dokumenten gibt es dann den Namen des Erstellers
  ''   ' Namen aus der Normal.dot entnehmen:
  ''   str1 = NormalTemplate.BuiltInDocumentProperties("Author")
  ''   ' es handelt sich bei einer neuen Datei stets um eine Datei des Autors!
  ' Es kann sein, dass der Name in der Form "Nachname, Vorname" vorliegt,
  ' deshalb Reihenfolge tauschen:
  j = InStr(str1, ",")
  If j > 0 Then
    str1 = Mid(str1, j + 2) & " " & Left(str1, j - 1)
  End If
  Selection.TypeText Text:=str1
'  ********
'  Leerzeichen und Tabulator:
  Selection.TypeText Text:="   "
  Selection.TypeText Text:=vbTab
'  ********
'  Dann das Datum in der Form "6. März 2013".
  Selection.TypeText Text:=Format(Date, "d. mmmm yyyy")
  Selection.TypeParagraph
'  ********
'  Absatz-Formatierung wieder zurückstellen:
  Selection.ParagraphFormat.Reset
  Selection.TypeParagraph
End Sub
'-----------------------------------------------------------------------------------------------------------
```

5.2 Alle Hyperlinks im Dokument wahlweise aktivieren oder deaktivieren

```
Option Explicit


Sub Hyperlinks_färben_aktivieren_oder_deaktivieren()
'  Dipl.-Ing. Hanna-Chris Gast, Rahmenprozedur 15. + 18. + 21. April 2016, 18. + 22. Mai 2018.
'  Färbt in allen Dokumentkomponenten (StoryRanges) Hyperlinks blau und unterstreicht sie,
'  auch E-Mails werden gefärbt.
'
'  Wenn etwas markiert ist, im Markierungsbereich ohne Rückfrage die Hyperlinks
'  färben und aktivieren.
'
'  Wenn nichts markiert ist, im gesamten Dokument in allen Dokumentkomponenten
'  (in allen StoryRanges) alle Hyperlinks färben
'  und je nach Abfrage Hyperlinks aktivieren oder deaktivieren.
'
'  MsgBox mit Abfrage:
'  Aktiviert bei "ja" die Hyperlinks.
'  Inaktive Hyperlinks auf andere Dokumente im Computer werden nicht aktiviert.
'
'  Bei "Nein" werden alle Hyperlinks in einer zweiten Unterprozedur deaktiviert.
'  Da manchmal bei Hyperlinks  Feldcode  und Feldergebnis voneinander abweichen,
'  ist es sinnvoll, beim Entlinken sowohl ehemaligen Feldcode-Text als auch Feldergebnis anzugeben.
'  Da inaktive Hyperlinks manchmal mühsam zu finden sind, wird hier die Unterprozedur zum
'  Aktivieren von Hyperlinks zuerst aufgerufen.
'
'  Bei "Abbrechen" wird das Makro abgebrochen.
'
'  Rahmenprozedur zur Suche auch in Fußnoten, Endnoten, Kopf- und Fußzeilen.
'  ***************************
Dim Rng0 As Range
   Set Rng0 = Selection.Range
Dim IntStoryType
   IntStoryType = Selection.Range.StoryType
Dim rngStory As Range        ' Dokumentkomponente
Dim isection As Integer
Dim aheaderFooter As HeaderFooter
Dim IntAbfrage As Integer
Dim Bl_Entlinken As Boolean
'  ************************************************
'  Wenn etwas markiert ist, im Bereich der Markierung alle Verweisungen ins Internet
'  als Hyperlink aktivieren:
  If Len(Selection) > 1 Then
     Call Hyperlinks_in_Markierung_aktivieren_und_faerben
     Exit Sub
  End If
' ************************************************
' Wenn nichts markiert ist:
' Abfrage, ob alle Hyperlinks aktiviert oder deaktiviert werden sollen.
  IntAbfrage = MsgBox("Wollen Sie alle Hyperlinks im Dokument aktivieren, dann drücken Sie 'Ja'." _
     & vbCrLf _
     & vbCrLf & "Bei 'Nein' werden alle Hyperlinks deaktiviert (entlinkt), aber gefärbt." _
     & vbCrLf _
     & vbCrLf & "Bei 'Abbrechen' wird das Makro abgebrochen.", _
      vbYesNoCancel, "Hyperlinks aktivieren oder deaktivieren?")
  If IntAbfrage = 6 Then      ' = ja
     Bl_Entlinken = False
  ElseIf IntAbfrage = 7 Then  ' nein
     Bl_Entlinken = True
  Else
```

```
    ' Makro abbrechen = 2
    Exit Sub
  End If
'  ***************************
Dim Zeit1 As Single
Dim Zeit2 As Single
Dim Zeitdauer As Single
'  Zeitmessen für die Rechendauer:
  Zeit1 = Timer
'  ***************************
'  Lästiges "Flackern" des Bildschirms verhindern:
  StatusBar = "Bitte warten - das Makro läuft."
  Application.ScreenUpdating = False
  Options.Pagination = False
'  ***********************************************
'  Hauptkomponente, Fußnoten, Endnoten,
'  und Textfelder durchsuchen:
'  ***********************************************
  For Each rngStory In ActiveDocument.StoryRanges
    ' Hauptkomponente, Fußnoten, Endnoten und Textfelder:
    If rngStory.StoryType > 5 Then Exit For
    rngStory.Select
    ' Die eigentliche Prozedur:
    Call Hyperlinks_in_Markierung_aktivieren_und_faerben
    ' alle Hyperlinks entlinken und wieder färben, wenn gewünscht:
    If Bl_Entlinken = True Then
      Call Hyperlinks_entlinken_und_faerben
    End If
  Next rngStory
'  ***********************************************
'  Kopf- und Fußzeilen durchsuchen:
'  ***********************************************
'  In allen Sections:
  For isection = 1 To ActiveDocument.Sections.Count
    ' ******************
    ' In allen Kopfzeilen des "Section":
    For Each aheaderFooter In ActiveDocument.Sections(isection).Headers
      aheaderFooter.Range.Select
      ' Die eigentliche Prozedur:
      Call Hyperlinks_in_Markierung_aktivieren_und_faerben
      If Bl_Entlinken = True Then
        Call Hyperlinks_entlinken_und_faerben
      End If
    Next aheaderFooter
    ' *****************
    ' in allen Fußzeilen des "Section":
    For Each aheaderFooter In ActiveDocument.Sections(isection).Footers
      aheaderFooter.Range.Select
      ' Die eigentliche Prozedur:
      Call Hyperlinks_in_Markierung_aktivieren_und_faerben
      If Bl_Entlinken = True Then
        Call Hyperlinks_entlinken_und_faerben
      End If
    Next aheaderFooter
    '  ****************
  Next isection
'  ***********************************************
'  In Seitenansicht gehen und ggf. das zweite Windows-Fenster schließen:
  If ActiveWindow.View.SplitSpecial = wdPaneNone Then
    ActiveWindow.ActivePane.View.Type = wdPrintView
  Else
    ActiveWindow.Panes(2).Close
```

```
    ActiveWindow.View.Type = wdPrintView
  End If
  '
  If ActiveWindow.ActivePane.View.Type = wdNormalView _
  Or ActiveWindow.ActivePane.View.Type = wdOutlineView Then
    ActiveWindow.ActivePane.View.Type = wdPrintView
  End If
  ' In die Hauptkomponente gehen:
  ActiveWindow.ActivePane.View.SeekView = wdSeekMainDocument
  ActiveWindow.ActivePane.View.Type = wdPrintView
'  ***************************
'  Inhaltsverzeichnisse zurückstellen in Hauptkomponente:
Dim aTableCont As TableOfContents
  For Each aTableCont In ActiveDocument.TablesOfContents
    aTableCont.Update
  Next aTableCont
  For Each aTableCont In ActiveDocument.TablesOfFigures
    aTableCont.Update
  Next aTableCont
'  ***************************
'  Markierung wiederherstellen:
  ActiveDocument.StoryRanges(IntStoryType).Select
  Rng0.Select
'  ***************************
'  Bildschirm wieder einschalten:
  Application.Visible = True
  Options.Pagination = True
  StatusBar = False
  Application.ScreenUpdating = True
  Application.DisplayStatusBar = True
'  ***************************
'  Zeitmessen für die Rechendauer
      Zeit2 = Timer
      Zeitdauer = Zeit2 - Zeit1
      Zeitdauer = Format(Zeitdauer, "##,###0.000")
'  ***************************
'  Auswertung:
  MsgBox "Die Rechenzeit war " & Zeitdauer & " Sekunden."
End Sub
'-----------------------------------------------------------------------------------------------------------------


Private Sub Hyperlinks_in_Markierung_aktivieren_und_faerben()
'  Makro erstellt von Dipl.-Ing. Hanna-Chris Gast, 2011, 2012, 2014,
'  2015, 11. + 31. August 2015, 8. Oktober 2015, 18. + 19. +22. Mai 2018.
'  Dieses Makro färbt Hyperlinks und E-Mails blau und unterstreicht sie.
'  Besonderheit:
'  Es wird auch nach nicht-aktiven Internet- und E-Mail-Adressen gesucht,
'  und diese werden aktiviert.
'  ********************************
Dim afield As Field
'  Markierung speichern:
Dim Rng1 As Range
  Set Rng1 = Selection.Range
Dim RngI As Range  ' Inhaltsverzeichnis, das nicht durchsucht werden darf.
'  ********************************************************************
'  Inaktive Hyperlinks und inaktive E-Mail-Adressen aktivieren:
'  ********************************************************************
  Rng1.Select
'  ***************
  ' Suche nach "www":
  Selection.Find.ClearFormatting
```

```
With Selection.Find
  .Text = "www.*."
  .Replacement.Text = ""
  .Forward = True
  ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
  .Format = True
  .MatchCase = False
  .MatchWholeWord = False
  .MatchAllWordForms = False
  .MatchSoundsLike = False
  .MatchWildcards = True
  Do While .Execute
    'If Selection.InRange(rng1) = False Then Exit Do
    If Selection.InRange(Rng1) = True Then
      ' Ganzen Ausdruck markieren (bis Leerzeichen oder Satzzeichen, auch Tabulator):
      Selection.MoveStartUntil CSet:=" ,!?;" & ChrW(13) & ChrW(160) _
        & vbTab, Count:=wdBackward
      Selection.MoveEndUntil CSet:=" ,!?;" & ChrW(13) & ChrW(160) & vbTab, Count:=wdForward
      '
      ' Wenn Absatzmarke in der Markierung, ist es kein Hyperlink:
      If InStr(Selection.Text, ChrW(13)) > 0 Then
        ' Absatzmarke in Markierung.
        Selection.Collapse Direction:=wdCollapseEnd
        ' Nur wenn im Hyperlink keine Sternchen stehen, Hyperlink erzeugen:
      ElseIf InStr(Selection.Text, "*") = 0 Then
        On Error Resume Next:
        ActiveDocument.Hyperlinks.Add _
            Anchor:=Selection.Range, _
            Address:="http://" & Selection.Text, _
            SubAddress:="", ScreenTip:="", TextToDisplay:=Selection.Text
        Selection.Collapse Direction:=wdCollapseEnd
        If Err.Number > 0 Then
          ActiveDocument.Undo 1
          Selection.Range.HighlightColorIndex = wdPink
          Selection.Collapse Direction:=wdCollapseEnd
        End If
        On Error GoTo 0
      Else
        ' Enthält Sternchen!
        Selection.Range.HighlightColorIndex = wdRed
        Selection.Collapse Direction:=wdCollapseEnd
      End If
    Else
      Selection.Collapse Direction:=wdCollapseEnd
    End If
  Loop
End With
' ************************************
' Suche nach "http://"
Rng1.Select
Selection.Find.ClearFormatting
With Selection.Find
  ".Text = "http://[!w]*."
   .Text = "http://*."
  .Replacement.Text = ""
  .Forward = True
  ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
  .Format = True
  .MatchCase = False
  .MatchWholeWord = False
  .MatchAllWordForms = False
  .MatchSoundsLike = False
```

```
  .MatchWildcards = True
  Do While .Execute
     'If Selection.InRange(rng1) = False Then Exit Do
     If Selection.InRange(Rng1) = True Then
       ' Ganzen Ausdruck markieren:
       Selection.MoveEndUntil CSet:=" ,!;" & ChrW(13), Count:=wdForward
       '
       ' Wenn Absatzmarke in der Markierung, ist es kein Hyperlink:
       If InStr(Selection.Text, ChrW(13)) > 0 Then
         Selection.Collapse Direction:=wdCollapseEnd
       ' In einem Hyperlink darf kein Sternchen stehen:
       ElseIf InStr(Selection.Text, "*") = 0 Then
         ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
             Address:=Selection.Text, _
             SubAddress:="", ScreenTip:="", TextToDisplay:=Selection.Text
         Selection.Collapse Direction:=wdCollapseEnd
       Else
         Selection.Range.HighlightColorIndex = wdRed
         Selection.Collapse Direction:=wdCollapseEnd
       End If
     Else
       Selection.Collapse Direction:=wdCollapseEnd
     End If
  Loop
End With
' ***********************************
' Suche nach "https://"
Rng1.Select
Selection.HomeKey Unit:=wdStory
Selection.Find.ClearFormatting
With Selection.Find
  .Text = "https://[!w]*."
  .Replacement.Text = ""
  .Forward = True
  ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
  .Format = True
  .MatchCase = False
  .MatchWholeWord = False
  .MatchAllWordForms = False
  .MatchSoundsLike = False
  .MatchWildcards = True
  Do While .Execute
     If Selection.InRange(Rng1) = True Then
       ' Ganzen Ausdruck markieren:
       Selection.MoveEndUntil CSet:=" ,!;" & ChrW(13), Count:=wdForward
       '
       ' Wenn Absatzmarke in der Markierung, ist es kein Hyperlink:
       If InStr(Selection.Text, ChrW(13)) > 0 Then
         Selection.Collapse Direction:=wdCollapseEnd
       ' In einem Hyperlink darf kein Sternchen stehen:
       ElseIf InStr(Selection.Text, "*") = 0 Then
         ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
           Address:=Selection.Text, _
           SubAddress:="", ScreenTip:="", TextToDisplay:=Selection.Text
         Selection.Collapse Direction:=wdCollapseEnd
       Else
         Selection.Range.HighlightColorIndex = wdRed
         Selection.Collapse Direction:=wdCollapseEnd
       End If
     Else
      Selection.Collapse Direction:=wdCollapseEnd
     End If
```

```
    Loop
  End With
  ' ***********************************
  ' Suche nach "ftp://"
  Rng1.Select
  Selection.Find.ClearFormatting
  With Selection.Find
    .Text = "ftp://[!w]*."
    .Replacement.Text = ""
    .Forward = True
    ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
    Do While .Execute
      If Selection.InRange(Rng1) = False Then Exit Do
      ' Ganzen Ausdruck markieren:
      "Selection.MoveStartUntil Cset:=" ,!?;" & ChrW(13), Count:=wdBackward
      Selection.MoveEndUntil CSet:=" ,!;" & ChrW(13), Count:=wdForward
      '
      ' Wenn Absatzmarke in der Markierung, ist es kein Hyperlink:
      If InStr(Selection.Text, ChrW(13)) > 0 Then
        Selection.Collapse Direction:=wdCollapseEnd
      ' In einem Hyperlink darf kein Sternchen stehen:
      ElseIf InStr(Selection.Text, "*") = 0 Then
        ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
          Address:=Selection.Text, _
          SubAddress:="", ScreenTip:="", TextToDisplay:=Selection.Text
        Selection.Collapse Direction:=wdCollapseEnd
      Else
        Selection.Range.HighlightColorIndex = wdRed
        Selection.Collapse Direction:=wdCollapseEnd
      End If
    Loop
  End With
' ***************************************
' Suche nach E-Mails bzw. nach dem Zeichen "@":
  Rng1.Select
  Selection.Find.ClearFormatting
  With Selection.Find
    .Text = "[0-9A-Za-z]\@*.[A-Za-z][A-za-z]"
    .Replacement.Text = ""
    .Forward = True
    ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
    Do While .Execute
      If Selection.InRange(Rng1) = False Then Exit Do
      ' Ganzen Ausdruck markieren:
      'Selection.MoveStartUntil Cset:=" ,!?;" & ChrW(13), Count:=wdBackward
      ' (gab Problem am Anfang der Dokumentkomponente)
      Selection.MoveStartWhile CSet:="ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÜ" _
        & "._-abcdefghijklmnopqrstuvwxyzäöüß", Count:=wdBackward
      'Selection.MoveEndUntil CSet:=" ,!;" & ChrW(13), Count:=wdForward
      Selection.MoveEndWhile CSet:="ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÜ" _
```

```
        & "._-abcdefghijklmnopqrstuvwxyzäöüß", Count:=wdForward
      ' In einem Hyperlink darf kein Sternchen stehen:
      If InStr(Selection.Text, "*") = 0 Then
        ' Für E-Mails Tabstopp bei 2 Zentimeter:
        Selection.ParagraphFormat.TabStops.ClearAll
        Selection.ParagraphFormat.TabStops.Add _
          Position:=CentimetersToPoints(2), _
          Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
        With Selection.Font
          .Color = wdColorBlue
          ' Hier war der Unterstrich nicht vom Leerzeichen zu unterscheiden.
          ' Deshalb jetzt doppelte Unterstreichung:
          .Underline = wdUnderlineDouble
          .UnderlineColor = wdColorGreen "wdColorBlue
        End With
        ' 11. August 2015 + 22. Mai 2018.
        If Selection.Range.Hyperlinks.Count = 0 Then
          ActiveDocument.Hyperlinks.Add _
            Anchor:=Selection.Range, Address:="mailto:" & Selection.Text, _
            SubAddress:="", ScreenTip:="", TextToDisplay:=Selection.Text
        End If
        Selection.Collapse Direction:=wdCollapseEnd
      Else
        ' Sternchen in E-Mails sind unzulässig!
        Selection.Range.HighlightColorIndex = wdRed
        Selection.Collapse Direction:=wdCollapseEnd
      End If
    Loop
  End With
' ****
' Markierung wiederherstellen:
  Rng1.Select
' ************************************
' aktive Hyperlinks (inklusive E-Mail-Adressen) blau färben
' und unterstreichen,
' ausgenommen Inhaltsverzeichnis:
  If ActiveDocument.TablesOfContents.Count > 0 Then
    Set RngI = ActiveDocument.TablesOfContents(1).Range
  Else
    Selection.HomeKey Unit:=wdStory
    Set RngI = Selection.Range
  End If
' ***************************
  For Each afield In Rng1.Fields
    If afield.Type = wdFieldHyperlink _
    And afield.Result.InRange(RngI) = False Then
      With afield.Result
        .Font.Color = wdColorBlue
        .Font.Underline = True
        .HighlightColorIndex = wdNoHighlight
      End With
    End If
  Next afield
  Rng1.Select
End Sub
' ---------------------------------------------------------------------------------------------------------


Private Sub Hyperlinks_entlinken_und_faerben()
' H. Chris Gast, 2011, 2012, 2014, 2015, 21. April 2016, 5. April 2017.
' Alle Hyperlinks werden in der Markierung deaktiviert, aber die Unterstreichung
' wird wieder hergestellt. E-Mails werden doppelt unterstrichen.
```

```
'
' Da manchmal bei Hyperlinks Feldcode  und Feldergebnis voneinander abweichen,
' ist es sinnvoll, beim Entlinken den Feldcode-Text dazuzusetzen:
' Weicht der Link vom angezeigten Text ab, wird beides angezeigt,
' sowohl Feldcode als auch ehemaliges Feldergebnis:
' Muster: "http://www.din."  ((ursprüngliche Anzeige = "DIN-Webseite"))
'
' Extras:
' Leerzeichen in Hyperlink werden grün markiert
' (eigentlich sind ja Leerzeichen in Internetadressen zu vermeiden).
' "%20" ist der UTF8-Code für Leerzeichen, der im Makro decodiert wird.
' ************
' Markierung speichern:
Dim Rng1 As Range
  Set Rng1 = Selection.Range
Dim IntAbfrage As Integer
Dim Feldcode As String
Dim Feldergebnis As String
Dim Ergebnis As String
Dim afield As Field
' ***********
' Prozedur nur anwenden, wenn Hyperlinks vorhanden:
  If Rng1.Hyperlinks.Count = 0 Then Exit Sub
' ************************************************
' Hyperlinks aufheben, aber blaufärben und unterstreichen:
' Bei Abweichung zwischen Feldergebnis und Feldcode wird beides angezeigt.
' Jedoch werden die Wort "Hyperlink" und "Mailto:" entfernt,
' ebenso ggf. der überflüssig gewordene Zusatz "\o" mit der Anzeigeoption.
' ************************************************
' Feldcode-Ansicht wählen, um das Wort "Hyperlink" zu suchen:
  ActiveWindow.View.ShowFieldCodes = True
  ' **************
  ' Inhaltsverzeichnis ausnehmen:
  Dim RngI As Range
  If ActiveDocument.TablesOfContents.Count > 0 Then
    Set RngI = ActiveDocument.TablesOfContents(1).Range
  Else
    Selection.HomeKey Unit:=wdStory
    Set RngI = Selection.Range
  End If
' ********************************************************
  For Each afield In Rng1.Fields
     If InStr(afield.Code, "HYPERLINK") <> 0 _
     And afield.Result.InRange(RngI) = False Then
       ' Wenn Feldcode vom Feld-Ergebnis abweicht, beides anzeigen:
       Feldcode = afield.Code
       Feldcode = Replace(Feldcode, "HYPERLINK ", "")
       Feldcode = Trim(Replace(Feldcode, """", ""))
       Feldcode = Replace(Feldcode, "mailto:%20", "")
       Feldcode = Replace(Feldcode, "mailto:", "")
       Feldcode = Replace(Feldcode, "mailto: ", "")
       If InStr(Feldcode, "\o") > 4 Then _
          Feldcode = Left(Feldcode, InStr(Feldcode, "\o") - 1)
       Feldcode = Trim(Feldcode)
       ' 21.4.2016:
       Feldcode = Replace(Feldcode, "%20", " ")
       ' ********
       Feldergebnis = Trim(afield.Result)
       Ergebnis = "  ((ursprüngliche Anzeige = """ & Feldergebnis & """))"
       ' Verknüpfung löschen und Textstelle wieder blaufärben:
       afield.Result.Select
       afield.Unlink
```

```
      Selection.Font.Reset
      Selection.Text = Feldcode
      If InStr(Selection, "@") = 0 Then
        With Selection
          .Font.Color = wdColorBlue
          .Font.Underline = True
          '' .HighlightColorIndex = wdNoHighlight
        End With
      Else
        With Selection
          .Font.Color = wdColorBlue
          .Font.Underline = wdUnderlineDouble
          .Font.UnderlineColor = wdColorGreen
          '' .HighlightColorIndex = wdNoHighlight
        End With
      End If
      Selection.Collapse Direction:=wdCollapseEnd
      Selection.Font.Reset
      Selection.TypeText Text:="  "
      ' Ursprünglichen Anzeigetext, wenn er abwich, dahintersetzen
      ' (das Wort "Mailto" sollte dabei nicht erscheinen. 9. Juli 2015):
      ' If Feldcode <> Feldergebnis Then
      If Feldcode <> Replace(Feldergebnis, "mailto:", "") Then
        'Selection.InsertAfter Ergebnis
        Selection.Collapse Direction:=wdCollapseEnd
        With Selection
          .Text = Ergebnis
          .Font.ColorIndex = wdGreen
          .Font.Italic = True
          .Font.Name = "Arial"
          .Font.Size = 9
        End With
      End If
    End If
  Next afield
  ' **************
  ActiveWindow.View.ShowFieldCodes = False
' ******************************************
' Unterstrichene Unterstriche markieren:
  Rng1.Select
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
      .Text = "_"
      .Font.Underline = wdUnderlineDouble ' wdUnderlineSingle
      .Replacement.Text = "_"
      .Forward = True
      .Format = False
      ' .Replacement.Font.Underline = wdUnderlineSingle ''wdUnderlineNone
      .Replacement.Font.UnderlineColor = wdColorRed
      .Replacement.Font.Bold = True
      '.Replacement.Highlight = True
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
  ' Bei doppelter Unterstreichung:
  Rng1.Select
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
```

```
  With Selection.Find
      .Text = "_"
      .Font.Underline = wdUnderlineSingle
      .Replacement.Text = "_"
      .Forward = True
      .Format = False
      .Replacement.Font.Underline = wdUnderlineDouble
       ".Replacement.Font.UnderlineColor = wdColorRed
       ".Replacement.Font.Color = wdColorRed
      .Replacement.Font.Bold = True
      '.Replacement.Highlight = True
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
' ****************
' Unterstrichene Leerzeichen markieren:
  Rng1.Select
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
      .Text = " "
      .Font.Underline = wdUnderlineSingle
      .Replacement.Text = " "
      .Forward = True
      .Format = False
      " .Replacement.Font.Underline = wdUnderlineDouble ' wdUnderlineNone
      " .Replacement.Font.UnderlineColor = wdColorRed
      .Replacement.Highlight = True
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
' Doppelt unterstrichene Leerzeichen:
  Rng1.Select
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
      .Text = " "
      .Font.Underline = wdUnderlineDouble
      .Replacement.Text = " "
      .Forward = True
      .Format = False
      " .Replacement.Font.Underline = wdUnderlineDouble ' wdUnderlineNone
      " .Replacement.Font.UnderlineColor = wdColorRed
      .Replacement.Highlight = True
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
' ************************************
  Rng1.Select
End Sub
' ----------------------------------------------------------------------------------------------------
```

6 Gänsefüßchen durch typografische Anführungszeichen ersetzen

Manchmal muss ich typografische Anführungszeichen verwenden, auch wenn ich sie nicht mag und privat nie verwende.

Im folgenden Makro wird die vorhandene Einstellung der Autokorrektur-Optionen gespeichert, dann ggf. für das Makro vorübergehend geändert. Ferner wird sichergestellt, dass keine Feldcodes sichtbar sind (diese würden durch typografische Anführungszeichen unbrauchbar). Außerdem darf in der Markierung kein Makro-Code enthalten sein. Der würde sonst bei typografischen Anführungszeichen unbrauchbar.

Mit dem Befehl "Suchen → Ersetzen" werden dann alle Anführungszeichen durch typografische ("deutsche") Anführungszeichen ersetzt.

Eine sichere Unterscheidung zwischen halben Anführungszeichen (2. Ebene) und Apostrophs per Makro ist mir nicht gelungen. Man denke nur an den Zungenbrecher

'Der Papst hat 's Besteck z'spät b'stellt.'

Deshalb konnte ich 'Halbe einfache Anführungszeichen' nicht zufriedenstellend in typografische umwandeln und habe darauf verzichtet.

Tabelle 6.1: Anführungszeichen in Abhängigkeit von der Sprache

	Typografisch		**Einfach**	
	1. Ebene	**2. Ebene**	**1. Ebene**	**2. Ebene**
Deutsch	„Zitat“	‚Zitat‘	"Zitat"	'Zitat'
Englisch	“quotation”	‘quotation’		
Französisch	« citation »	‹ci tation›		

Ich kann nicht immer Apostrophs von halben Anführungszeichen unterscheiden. Es werden hierbei von Word nur die englischen Regeln angewendet. Im Englischen ist auch das typografische Apostroph mit dem halben typografischen Anführungszeichen rechts identisch. In deutschen Texten besteht hier ein Unterschied.

Ich habe also auf die Typografischen halben Anführungszeichen (2. Ebene) völlig verzichtet und hier stattdessen im Makro einige Befehlszeilen ergänzt, die nur die Apostrophs in typografische Apostrophs umwandeln, auch Apostrophs am Wortanfang. Dafür muss die Option für Typografische Anführungszeichen wieder vorübergehend abgeschaltet werden.

Am Schluss werden die AutoKorrektur-Optionen zurückgesetzt.

In den folgenden beiden Makros ist der Ersatz von Zeichen auf die Markierung beschränkt!

```
Sub a4a_Gänsefüßchen_typografisch()
'  Makro erstellt von Dipl.-Ing. Hanna-Chris Gast am 8. Mai 2011, Stand 11. Februar 2012.
'  Dieses Makro ersetzt alle einfachen Gänsefüßchen durch typografische Anführungszeichen.
'  Problem:  Es werden auch bei englischen Zitaten deutsche Anführungszeichen gesetzt.
'  Halbe Anführungszeichen werden hier  durch typografische Apostrophs ersetzt
'  (weil halbe Anführungszeichen bei mir nie vorkommen im Gegensatz zu Apostrophs).
' Am Schluss wird wieder die Einstellung im Computer zurückgestellt.
Dim Gänsefüßchenart1 As Boolean
Dim Gänsefüßchenart2 As Boolean
'
'  Autokorrektur für die Anführungszeichen einschalten:
  If Options.AutoFormatAsYouTypeReplaceQuotes = False Then
     Gänsefüßchenart1 = False
     Options.AutoFormatAsYouTypeReplaceQuotes = True
  Else
    Gänsefüßchenart1 = True
  End If
  If Options.AutoFormatReplaceQuotes = False Then
     Gänsefüßchenart2 = False
     Options.AutoFormatReplaceQuotes = True
   Else
    Gänsefüßchenart2 = True
  End If
'
'''  Die ganze Datei bearbeiten:
''   Selection.WholeStory
'
' Sicherstellen, dass keine Makro-Programme verändert werden:
If InStr(Selection, "End Sub") > 0 And InStr(Selection, "Sub ") > 0 Then
  MsgBox "Die markierte Stelle enthält vermutlich eine Makro-Prozedur, " & vbCrLf _
      & "wo Gänsefüßchen und Apostrophs nicht verändert werden dürfen!" & vbCrLf & vbCrLf _
      & "Das Makro wird deshalb abgebrochen.", vbCritical, "Achtung!"
  Exit Sub
End If
'
' Sicherstellen, dass keine Feldcodes sichtbar sind
' (dort sind typografische Anführungszeichen verboten):
 If ActiveWindow.View.ShowFieldCodes = True Then
    ActiveWindow.View.ShowFieldCodes = False
  End If
' ********************
' Ersetzung von Gänsefüßchen durch Gänsefüßchen in der Markierung:
   Selection.Find.ClearFormatting
   '' Selection.Find.Style = wdStyleNormal      ' Ersetzung auf "Standard" beschränken
   Selection.Find.Replacement.ClearFormatting
   ''   Selection.Find.LanguageID = wdGerman
   With Selection.Find
      .Text = """"
      .Replacement.Text = """"
      .Forward = True
''         .Wrap = wdFindContinue   ' Würde zum Ersetzen in der ganzen Datei führen.
         .Wrap = wdFindStop
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
   End With
   Selection.Find.Execute Replace:=wdReplaceAll
'
'  ********************************************
```

```
'  Alle  einfachen halben Anführungszeichen (Apostrophs) ersetzen
'  (es werden leider die englischen Zeichen verwendet, aber wenigstens stimmen dann
'   weitgehend die Semikolons:
'
' Ohne folgende Option ist ein typografisches Apostroph an einen Wortbeginn unmöglich:
  Options.AutoFormatAsYouTypeReplaceQuotes = False
  Options.AutoFormatReplaceQuotes = False
'
   Selection.Find.ClearFormatting
  ''  Selection.Find.Style = wdStyleNormal      '  ' Ersetzung auf "Standard" beschränken..
   Selection.Find.Replacement.ClearFormatting
   With Selection.Find
      .Text = ChrW(39)                  ' Das einfache Apostroph '''
           .Replacement.Text = ChrW(8217)    ' Damit Apostrophs auch am Wortanfang richtig werden!
      '     .Replacement.Text = " ' "
                                      ' Damit werden halbe Anführungszeichen nur auf englische Art ersetzt.
      .Forward = True
      ''    .Wrap = wdFindContinue
           .Wrap = wdFindStop
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
   End With
   Selection.Find.Execute Replace:=wdReplaceAll
'
' ****************************************
'  Ursprüngliche Gänsefüßchenart in Word wieder einstellen:
   If Gänsefüßchenart1 = False Then
       Options.AutoFormatAsYouTypeReplaceQuotes = False
    End If
    If Gänsefüßchenart2 = False Then
       Options.AutoFormatReplaceQuotes = False
    End If
End Sub
```

Wer wie ich keine typografischen Anführungszeichen mag, kann das folgende Makro benutzen:

```
Sub a4b_Gänsefüßchen_einfach()
'  Dipl.-Ing. Hanna-Chris Gast,8. Mai 2011, 11. Februar 2012, 17. Januar 2017.
'  Dieses Makro ersetzt alle markierten Gänsefüßchen durch einfache Gänsefüßchen (")
'  sowie alle halben Anführungszeichen und Apostrophs durch einfache Apostrophs (').
Dim Gänsefüßchenart1 As Boolean
Dim Gänsefüßchenart2 As Boolean
'
'  Autokorrektur für die Anführungszeichen abschalten:
   If Options.AutoFormatAsYouTypeReplaceQuotes = True Then
       Gänsefüßchenart1 = True
       Options.AutoFormatAsYouTypeReplaceQuotes = False
    Else
      Gänsefüßchenart1 = False
   End If
   If Options.AutoFormatReplaceQuotes = True Then
       Gänsefüßchenart2 = True
       Options.AutoFormatReplaceQuotes = False
    Else
```

```
    Gänsefüßchenart2 = False
  End If
'
' ****************************
'''  Die ganze Datei bearbeiten:
''   Selection.WholeStory
'
' Anführungszeichen durch einfache Gänsefüßchen "QUOTATION MARK" ersetzen:
   Selection.Find.ClearFormatting
   Selection.Find.Replacement.ClearFormatting
   With Selection.Find
      .Text = """"
      .Replacement.Text = """"
      .Forward = True
       ''  .Wrap = wdFindContinue
          .Wrap = wdFindStop
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
   End With
   Selection.Find.Execute Replace:=wdReplaceAll
'
'  Englische "RIGHT DOUBLE QUOTATION MARK" ersetzen durch einfache Gänsefüßchen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = ChrW(8221)
    .Replacement.Text = """"
    .Forward = True
    ''  .Wrap = wdFindContinue
       .Wrap = wdFindStop
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
'
'  Alle Apostrophs (= einfache halbe Anführungszeichen) ersetzen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "'"
    .Replacement.Text = "'"
    .Forward = True
    ''  .Wrap = wdFindContinue
       .Wrap = wdFindStop
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
'
'  "halbes Anführungszeichen unten" ("SINGLE LOW-9 QUOTATION MARK")
```

```
'   durch einfaches "Apostroph" ersetzen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = ChrW(8218)  ' Das Zeichen Unicode dez. 8218 sieht wie ein Komma aus.
    .Replacement.Text = "'"
    .Forward = True
      '  .Wrap = wdFindContinue
         .Wrap = wdFindStop
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
'
'  ******************************
'  Ursprüngliche Gänsefüßchenart in Word wieder einstellen:
  If Gänsefüßchenart1 = False Then
     Options.AutoFormatAsYouTypeReplaceQuotes = False
   End If
   If Gänsefüßchenart2 = False Then
     Options.AutoFormatReplaceQuotes = False
   End If
End Sub
```

7 Aufzählungen per Makro erzeugen

Die hier entwickelten Makros für Aufzählungen erzeugen keine Formatvorlagen und sind unabhängig von in der Datei vorhandenen Formatvorlagen. Die Makros erzeugen Aufzählungen unabhängig von Aufzählungs-Ebenen. Es wird die vorhandene Einrückung am Beginn der Markierung als Referenz genommen. Ich habe mir dies Makros als Buttons in eine Menüband-Registerkarte (Ribbon-Tab) eingefügt.

```
Sub 5a_Aufzählung_Buchstaben()
' Dipl.-Ing. Hanna-Chris Gast, erstellt 18. Oktober 2011, 13. Januar 2014.
' Die Einrückung richtet sich nach dem Anfang der Markierung.
Dim Einrückung As Single
Dim rng0 As Range
Dim rng1 As Range
' ******************************
' Einrückung bestimmen (dabei letzten Abschnitt in einer Tabelle gezielter ansprechen,
  '  damit nicht das ganze Tabellenfeld markiert wird):
  If Selection.Information(wdWithInTable) Then
      ' Bei letztem Absatz in Tabellen-Zelle Absatzmarke ergänzen.
      Selection.Cells(1).Range.InsertAfter Chr(13)
      Selection.Expand Unit:=wdParagraph  ' macht Problem in letztem Absatz einer Tabellenzelle.
     'Absatzmarke wieder entfernen:
     Selection.Cells(1).Range.Characters.Last.Previous.Delete
  Else
     Selection.Expand Unit:=wdParagraph
  End If
  Set rng0 = Selection.Range
  Set rng1 = Selection.Range
' ******************************
  rng0.Collapse  ' (Die Einrückung wird nur für den ersten Absatz  bestimmt)
  ' Einrückung = PointsToCentimeters(Rng0.ParagraphFormat.LeftIndent)
'    Einrückung = Round(PointsToCentimeters(Rng0.ParagraphFormat.LeftIndent), 2)
  If Selection.Range.ListFormat.ListType = wdListNoNumbering Then
      Einrückung = Round(PointsToCentimeters(rng0.ParagraphFormat.LeftIndent _
              + rng0.ParagraphFormat.FirstLineIndent), 2)
  Else
     Einrückung = Round(PointsToCentimeters(rng0.ParagraphFormat.LeftIndent), 2)
  End If
'
'' Ggf. Hinweis, dass Buchstaben nur für 1. Ebene:
'  If Not Selection.Range.ListFormat.ListType = wdListNoNumbering Then
'     MsgBox "Hier handelt es sich um eine Aufzählung höherer Ebene." & vbCrLf _
'          & " Eigentlich sollten Buchstaben nur für die 1. Ebene verwendet werden."
'  End If
' ******************************
' Störende vorhandene Tabstopps löschen bzw. überschreiben:
  Selection.ParagraphFormat.TabStops.ClearAll  ' löscht nicht die Standard-Tabstopps!
  ActiveDocument.DefaultTabStop = CentimetersToPoints(0)
  Selection.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(0.7 + Einrückung), _
      Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
'
' Die Aufzählung mit Buchstaben:
  With ListGalleries(wdNumberGallery).ListTemplates(5).ListLevels(1)
     .NumberFormat = "%1)"
     .TrailingCharacter = wdTrailingTab
     .NumberStyle = wdListNumberStyleLowercaseLetter
     .Alignment = wdListLevelAlignLeft
     '     .NumberPosition = CentimetersToPoints(0)
     .NumberPosition = CentimetersToPoints(0 + Einrückung)
```

```
    .TextPosition = CentimetersToPoints(0.71 + Einrückung)
    .TabPosition = CentimetersToPoints(0.71 + Einrückung)
    .ResetOnHigher = 0
    .StartAt = 1
    With .Font
      '' .Size = 12
      .Bold = wdUndefined
      .Italic = False
      .Name = rng0.Font.Name  ' .Name = "Arial"
    End With
  End With
  ListGalleries(wdNumberGallery).ListTemplates(5).Name = ""
  Selection.Range.ListFormat.ApplyListTemplate _
      ListTemplate:=ListGalleries(wdNumberGallery).ListTemplates(5), _
      ContinuePreviousList:=False, _
      DefaultListBehavior:=wdWord8ListBehavior, _
      ApplyTo:=WdListApplyTo.wdListApplyToSelection
'
''' Zeilenabstand einstellen:
''  With Selection.ParagraphFormat
''      .SpaceBefore = 0
''      .SpaceAfter = 9  ' 3
''      .SpaceAfterAuto = False
''  End With
'  ***********************************
   ''' Cursor ans Ende der Markierung setzen:
   '' Selection.Collapse Direction:=wdCollapseEnd
   '' Selection.MoveLeft Unit:=wdCharacter, Count:=1
  rng1.Select
End Sub


Sub a5b_Aufzählung_Ziffern()
' Dipl.-Ing. Hanna-Chris Gast, 18.10.2011, Stand 13. Januar 2014.
' Die Einrückung richtet sich nach dem Anfang der Markierung.
Dim Einrückung As Single
Dim rng0 As Range
'  ******************************
'  Einrückung bestimmen (dabei letzten Abschnitt in einer Tabelle gezielter ansprechen,
   '  damit nicht das ganze Tabellenfeld markiert wird):
   If Selection.Information(wdWithInTable) Then
      ' Bei letztem Absatz in Tabellen-Zelle Absatzmarke ergänzen.
      Selection.Cells(1).Range.InsertAfter Chr(13)
      Selection.Expand Unit:=wdParagraph  ' macht Problem in letztem Absatz einer Tabellenzelle.
     'Absatzmarke wieder entfernen:
     Selection.Cells(1).Range.Characters.Last.Previous.Delete
   Else
      Selection.Expand Unit:=wdParagraph
   End If
   Set rng0 = Selection.Range
'  ******************************
   rng0.Collapse  ' (Die Einrückung wird nur für den ersten Absatz  bestimmt)
   If Selection.Range.ListFormat.ListType = wdListNoNumbering Then
      Einrückung = Round(PointsToCentimeters(rng0.ParagraphFormat.LeftIndent _
              + rng0.ParagraphFormat.FirstLineIndent), 2)
   Else
      Einrückung = Round(PointsToCentimeters(rng0.ParagraphFormat.LeftIndent), 2)
   End If
'  ****************************
'  Störende vorhandene Tabstopps löschen bzw. überschreiben:
   Selection.ParagraphFormat.TabStops.ClearAll  ' löscht nicht die Standard-Tabstopps!
```

```
  ActiveDocument.DefaultTabStop = CentimetersToPoints(0)
  Selection.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(0.7 + Einrückung), _
    Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
'  ****************************
'  Die Aufzählung mit Ziffern:
  With ListGalleries(wdOutlineNumberGallery).ListTemplates(1).ListLevels(1)
    .NumberFormat = "%1)"
    .TrailingCharacter = wdTrailingTab
    .NumberStyle = wdListNumberStyleArabic
    .Alignment = wdListLevelAlignLeft
    .NumberPosition = CentimetersToPoints(0 + Einrückung)
    .TextPosition = CentimetersToPoints(0.7 + Einrückung)
    .TabPosition = CentimetersToPoints(0.7 + Einrückung)
    .ResetOnHigher = 0
    .StartAt = 1
    With .Font
   "  .Size = 12
      .Bold = False
      .Italic = False
      .Name = rng0.Font.Name  ' .Name = "Arial"
    End With
  End With
  Selection.Range.ListFormat.ApplyListTemplate _
    ListTemplate:=ListGalleries(wdOutlineNumberGallery).ListTemplates(1), _
    ContinuePreviousList:=False, _
    DefaultListBehavior:=wdWord8ListBehavior, _
    ApplyTo:=WdListApplyTo.wdListApplyToSelection
'
''' Zeilenabstand einstellen:
"   With Selection.ParagraphFormat
"       .SpaceBefore = 0
"       .SpaceAfter = 9 ' 3
"       .SpaceAfterAuto = False
"  End With
  ' Cursor ans Ende der Markierung setzen:
  "  Selection.Collapse Direction:=wdCollapseEnd
  "  Selection.MoveLeft Unit:=wdCharacter, Count:=1
 rng0.Select
End Sub


Sub a5c_Aufzählung_Striche()
' Dipl.-Ing. Hanna-Chris Gast, 18.10.2011, Stand 13. Januar 2014.
' Die Einrückung richtet sich nach dem Anfang der Markierung.
Dim Einrückung As Single
Dim rng0 As Range
' ******************************
' Einrückung bestimmen (dabei letzten Abschnitt in einer Tabelle gezielter ansprechen,
  '  damit nicht das ganze Tabellenfeld markiert wird):
 If Selection.Information(wdWithInTable) Then
     ' Bei letztem Absatz in Tabellen-Zelle Absatzmarke ergänzen.
     Selection.Cells(1).Range.InsertAfter Chr(13)
     Selection.Expand Unit:=wdParagraph  ' macht Problem in letztem Absatz einer Tabellenzelle.
    'Absatzmarke wieder entfernen:
    Selection.Cells(1).Range.Characters.Last.Previous.Delete
 Else
    Selection.Expand Unit:=wdParagraph
 End If
 Set rng0 = Selection.Range
' ******************************
 rng0.Collapse  ' (Die Einrückung wird nur für den ersten Absatz  bestimmt)
```

```
 ' Einrückung = PointsToCentimeters(Rng0.ParagraphFormat.LeftIndent)
 ' Einrückung = Round(PointsToCentimeters(Rng0.ParagraphFormat.LeftIndent), 2)
 If Selection.Range.ListFormat.ListType = wdListNoNumbering Then
    Einrückung = Round(PointsToCentimeters(rng0.ParagraphFormat.LeftIndent _
            + rng0.ParagraphFormat.FirstLineIndent), 2)
 Else
   Einrückung = Round(PointsToCentimeters(rng0.ParagraphFormat.LeftIndent), 2)
 End If
' *************************
' Störende vorhandene Tabstopps löschen bzw. überschreiben:
 Selection.ParagraphFormat.TabStops.ClearAll  ' löscht nicht die Standard-Tabstopps!
 ActiveDocument.DefaultTabStop = CentimetersToPoints(0)
 Selection.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(0.7 + Einrückung), _
    Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
'
' Die Aufzählung mit Strichen
 With ListGalleries(wdBulletGallery).ListTemplates(6).ListLevels(1)
     .NumberFormat = ChrW(8212)
     .TrailingCharacter = wdTrailingTab
     .NumberStyle = wdListNumberStyleBullet
     .Alignment = wdListLevelAlignLeft
     .NumberPosition = CentimetersToPoints(0 + Einrückung)
     .TextPosition = CentimetersToPoints(0.7 + Einrückung)
     .TabPosition = CentimetersToPoints(0.7 + Einrückung)
     .ResetOnHigher = 0
     .StartAt = 1
     With .Font
   "    .Size = 12
        .Bold = False
        .Italic = False
        .Name = "Arial"
     End With
     .LinkedStyle = ""
   End With
'
 Selection.Range.ListFormat.ApplyListTemplate _
   ListTemplate:=ListGalleries(wdBulletGallery).ListTemplates(6), _
   ContinuePreviousList:=False, _
   ApplyTo:=WdListApplyTo.wdListApplyToSelection, _
   DefaultListBehavior:=wdWord8ListBehavior
'
 rng0.Select
End Sub


Sub a5q_Aufzählungen_in_Text_umwandeln()
'  Makro erstellt von Dipl.-Ing. Hanna-Chris Gast, 8. September 2010.
  "   Selection.WholeStory
' Achtung: Dieses Makro wandelt auch Abschnittsnummern in feste Zahlen um.
  Selection.Range.ListFormat.ConvertNumbersToText
End Sub
```

Mit diesem Makro sollte man keinesfalls das ganze Dokument erfassen. Dieses Makro zerstört nämlich innerhalb der Markierung auch die automatische Nummerierung der Abschnittsüberschriften.

```
Sub a5z_Aufzählung_entfernen()
'  gefunden: 8. September 2010.
'  Entfernt in der Markierung alle Nummerierung und Einrückung.
  Selection.Range.ListFormat.RemoveNumbers
End Sub
```

8 Tabellenmakros

Eine elegante Lösung für das komfortable Erstellen von neuen Tabellen findet sich unter: www.schreibbuero-richter.de/word/tabelle.html

8.1 Rahmenlinien bei zerfransten Tabellen reparieren

Ändern sich bei mehrseitige Tabellen die Seitenumbrüche, lassen sich in einigen Fällen die vertikalen Tabellen-Linien nur noch dadurch reparieren (zum Fluchten bringen), dass man eine neue Tabelle erstellt und Zelle für Zelle den Inhalt der Tabelle über die Zwischenablage in die neue Tabelle überträgt.

11	12	13 14	
21	22	23	24
31	32	33	34
41	42	43	44
51	52	53	54

In dem folgenden Makro wird die Tabelle neu erstellt, indem **Zelle für Zelle** der Zelleninhalt in die neue Tabelle übertragen wird. Da die neue Tabelle erst formatiert und dann gefüllt wird, bleiben Markierungen und Text-Farben auch in Word 2000 erhalten.

Es werden die Kopfzeilen der alten Tabelle gezählt, und der neuen Tabelle die gleiche Anzahl an Kopfzeilen zugewiesen.

Bei verbundenen Zellen oder ungleicher Zellenanzahl je Reihe werden in der neuen Tabelle leere Zellen erzeugt. Hier muss von Hand entsprechend wieder Zellen verbunden werden. Deswegen bleibt die alte Tabelle erhalten. Sie kann dann, wenn die neue Tabelle o. k. ist, von Hand gelöscht werden.

Anleitung: Cursor in zu reparierende Tabelle stellen, Makro starten.

Nach etwa 3 bis 4 Sekunden pro Seite erscheint dann in einer neuen Datei eine reparierte Version der gesamten Tabelle.

	11	12	13 14	
	21	22	23	24
	31	32	33	34
41	42	43	44	
51	52	53	54	

Im Beispiel hier müssen die Inhalte der Zeilen 1 bis 3 nach links geschoben und dann die letzte Spalte gelöscht werden.

Verbundene Zellen (hier 13/14) müssen neu verbunden werden, ansonsten werden alle Text-Formatierungen in den Zellen wiederhergestellt. Danach kann die neue Tabelle anstelle der bisherigen eingefügt werden.

```
Sub a6_Tabelle_neu_erstellen_Schattierungserhalt()
' Dipl.-Ing. Hanna-Chris Gast, neu Mai 2014, 6. März 2017, 23. + 24. Mai 2017.
' Dieses Makro erzeugt eine neue Tabelle und fügt den Inhalt der alten Tabelle zellenweise dahinein.
' Word 2000 bis Word 2010, Windows XP und Windows 7.
' Text-Farben und Formatierungen bleiben erhalten, jetzt auch Hintergrund-Schattierungen.
' Die Anzahl der Kopfzeilen wird wieder hergestellt, sofern sie kleiner ist als die Anzahl der Reihen.
' Formate und Absatzausrichtung in Zelle bleibt erhalten.
```

```
' Es erfolgt eine Anpassung an den Satzspiegel am Ort der Tabelle, auch bei Querformat.
'
' Dieses Makro war in Word 2000 unter Windows XP schneller als in Word 2010 unter Windows 7!
' Die Zwischendatei bleibt während des Rechenvorgangs unsichtbar.
' Problem in Word 2000: Es entsteht bei der Testtabelle eine ungewünschte Silbentrennung.
'
' Hintergrundschattierung der Zelle soll möglichst erhalten bleiben.
' ******************************************************************************
  Dim Dok1 As Word.Document  ' die zu bearbeitende Datei
  Dim Dok2 As Word.Document  ' die vorübergehende Zwischen-Datei
  Dim a As Integer  ' Anzahl der Reihen
  Dim B As Integer  ' Anzahl der Spalten
  Dim C As Integer  ' Anzahl der Zellen
  Dim z As Cell     ' Tabellenfeld
  Dim i As Integer  ' jeweilige Reihe
  Dim j As Integer  ' jeweilige Spalte
  Dim N As Integer  ' Anzahl der Tabellenzeilen (Reihen)
  Dim K As Integer  ' Anzahl der Kopfzeilen
  Dim L As Integer  ' Anzahl der Zellen
  Dim Tabelle1 As Table  ' vorhandene Tabelle
  Dim Tabelle21 As Table ' temporäre Tabelle
  Dim Tabelle22 As Table ' verbesserte Tabelle
  Dim Warnhinweis1 As String: Warnhinweis1 = ""  ' Text, wenn Kopfzeilenzahl nicht bestimmbar
  Dim Warnhinweis2 As String: Warnhinweis2 = ""  ' Text, wenn verbundene Zellen vorhanden
  Dim Warnhinweis3 As String: Warnhinweis3 = ""  ' Text, wenn Schattierung nicht bestimmbar
  Dim IntAusrichtung As Integer
  Dim Ansichtstyp1 As Integer
  Dim Satzspiegelbreite_Points As Integer
  Dim B_Querformat As Boolean
' 6. März 2017
  Dim lnShade As Long
' ***********************************************
' Zeitmessen für die Rechendauer:
  Dim Zeit1
  Dim Zeit2
  Dim Zeitdauer
  Zeit1 = Timer
' ***********************************************
  If Selection.Information(wdWithInTable) = False Then
     MsgBox "Der Cursor ist nicht in einer Tabelle!"
     Exit Sub
  End If
' ******************************
Dim rng0 As Range  ' Markierung bei Start des Makros
'  Anfangsmakierung speichern:
   Set rng0 = Selection.Range
'  Hinweis in der Statuszeile:
   Application.StatusBar = "        Bitte warten - das Makro läuft."
'  ***************************
' bisherige Tabelle kopieren und sich Datei merken:
  Selection.Tables(1).Select
  Set Tabelle1 = Selection.Tables(1)
  Selection.Copy
  Set Dok1 = ActiveDocument
' ****************************
' Satzspiegelbreite merken:
  '  Satzspiegelweite bestimmen (Stand: 25. Jan. 2014):
'  Bei normales Papierformat kommt der Bundsteg ("Gutter") hinzu:
   If Selection.PageSetup.Orientation = wdOrientLandscape = False Then
      With Selection.Range.Sections(1).PageSetup
         Satzspiegelbreite_Points = (.PageWidth - (.LeftMargin + .RightMargin + .Gutter))
      End With
```

```
    B_Querformat = False
  Else
    ' Bei Querformat kommt kein Bundsteg hinzu:
    With Selection.Range.Sections(1).PageSetup
      Satzspiegelbreite_Points = (.PageWidth - (.LeftMargin + .RightMargin))
    End With
    B_Querformat = True
  End If
' ****************************************
' In einem (Zwischen-)Dokument die Tabelle zwischenspeichern:
  Set Dok2 = Documents.Add  ' Temporäre Datei
  If B_Querformat = True Then Dok2.PageSetup.Orientation = wdOrientLandscape
  Err.Clear
  ' falls wie bei mir in der Normal-dot schon Name und Datum vorgesehen ist:
  Selection.WholeStory
  Selection.Paste
  Selection.MoveUp
  Set Tabelle21 = Selection.Tables(1)
' ***********************************************************************
' ***********************************************************************
' Bildschirmaktualisierung während des Programms zur Beschleunigung abstellen.
  ' Zuerst die vorhandene Bildschirmansicht speichern:
    Ansichtstyp1 = (ActiveWindow.View.Type)
  ' Bildschirm zur Beschleunigung auf "Normalansicht" stellen, wenn dies nicht schon der Fall ist:
    If ActiveWindow.View.Type = wdNormalView = False Then
      ActiveWindow.View.Type = wdNormalView
    End If
  ' Bildschirmaktualisierung abstellen.
  StatusBar = "Bitte warten - das Makro läuft."        ' Geht nicht immer.
  Application.DisplayStatusBar = False   ' funktioniert nicht in Word 2010.
  Application.ScreenUpdating = False
  Options.Pagination = False
' Scrollbars ausblenden:   ' (brachte nichts)
  With ActiveWindow
      .DisplayHorizontalScrollBar = False
      .DisplayVerticalScrollBar = False
  End With
' ************************
' Noch mehr beschleunigen durch Unsichtbar-Machen von Word:
 ' On Error GoTo Zeile_Word_sichtbar
 ' Word.Application.Visible = False
  '   ActiveDocument.Visible = False  ' geht so nicht!
  Dok2.Windows(1).Visible = False
' ***********************************************************************
' ***********************************************************************
' Anzahl der Spaltenanzahl und der Reihen (Tabellenzeilen):
  Selection.Tables(1).Select
  a = Selection.Tables(1).Rows.Count
  B = Selection.Tables(1).Columns.Count
  C = Selection.Cells.Count
  '' MsgBox "Es gibt " & a & " Reihen und " & b & " Spalten."
' ***********************************************************************
' Kopfzeilen zählen:
' ****************************
  Selection.Tables(1).Select
  K = 0
  N = a ' = Selection.Tables(1).Rows.Count
  ''  MsgBox "Die Tabelle hat " & n & " Zeilen."
' In die erste Zeile gehen:
  '' Selection.Tables(1).Rows(1).Select  ' Gibt Fehler bei vertikal verbundenen Zellen.
  Selection.Tables(1).Select
  Selection.Collapse
```

```
' Kopfzeilen zählen:
  For i = 1 To N
    If Selection.Information(wdWithInTable) = False Then Exit For ' (bei verbundenen Reihen).
    If Selection.Rows.HeadingFormat = True Then
      ' jeweilige Kopfzeile markieren und Text zentrieren:
      '' Selection.Rows(1).Select
        '' Gibt Fehler, wenn irgendwo in der Tabelle vertikal verbundene Zellen sind.
      Selection.EndKey Unit:=wdRow
      Selection.HomeKey Unit:=wdRow, Extend:=True
      Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
      '
      If Selection.Information(wdEndOfRangeRowNumber) <> i Then
        i = 1  ' ergibt k = 0
        Warnhinweis1 = "Achtung: Bitte Kopfzeilen von Hand  zuweisen." & vbCrLf & vbCrLf
        Exit For
      End If
      Selection.MoveDown Unit:=wdLine, Count:=1
    Else
      Exit For
    End If
  Next i
  K = i - 1   ' Anzahl der Kopfzeilen.
  If K = N Then K = 0  ' es dürfen nicht alle Zeilen als Kopfzeile markiert sein.
' ************************************************************
' Neue Tabelle erstellen unterhalb der kopierten und formatieren(ab Word 2000):
  '  zuerst mit Cursor zwei Zeilen unterhalb der kopierten Tabelle gehen:
  Tabelle21.Select
  Selection.Collapse Direction:=wdCollapseEnd
  Selection.TypeParagraph
  Selection.TypeParagraph
  '
  ActiveDocument.Tables.Add Range:=Selection.Range, _
      NumRows:=a, NumColumns:=B, _
      DefaultTableBehavior:=wdWord9TableBehavior, _
      AutoFitBehavior:=wdAutoFitWindow  ' wdAutoFitContent 'wdAutoFitFixed
  ' Neue Tabelle formatieren (ab Word2000):
  ' (Kopfzeile wird erst ganz am Ende zugeordnet)
  Set Tabelle22 = Selection.Tables(1)
  Tabelle22.AutoFormat Format:=wdTableFormatGrid5, _
      ApplyBorders:=True, ApplyHeadingRows:=False, _
      ApplyLastRow:=False, ApplyFirstColumn:=False, _
      ApplyLastColumn:=False, AutoFit:=False, _
      ApplyShading:=False, ApplyColor:=False
  ' Damit keine fehlerhafte Silbentrennung auftritt, zuerst:
  Tabelle22.AutoFitBehavior (wdAutoFitContent)
' ********************
' Mindesthöhe einer Tabellenzeile (Reihe) festlegen:
  Tabelle22.Rows.HeightRule = wdRowHeightAtLeast
  Tabelle22.Rows.Height = CentimetersToPoints(0.5)
' Seitenumbrüche innerhalb einer Tabellenzeile (Reihe) verhindern!
  With Tabelle22
     .Rows.AllowBreakAcrossPages = False
  End With
' ********************************************************************
' Original-Tabelle (bzw. die Kopie in der Dok2) Zelle für Zelle auslesen
' und in neue Tabelle kopieren:
  Tabelle21.Select
  For Each z In Selection.Cells
    z.Select
    i = z.RowIndex
    j = z.ColumnIndex
    lnShade = Selection.Cells(1).Shading.BackgroundPatternColor
```

```
    ' Fehler abfangen bei Bestimmung der Schattierung (23. Mai 2017):
    If lnShade = -1 Or lnShade = 9999999 Then
      ' unbestimmbare Schattierfarbe (wohl wegen verbundener Zellen)
      lnShade = wdColorDarkRed 'wdColorAutomatic
      Warnhinweis3 = "Unbestimmbare Farbschattierung in Reihe " & i & " Spalte " & j & "!" _
      & vbCrLf & "Die Zelle wurde zwecks erleichterter Prüfung dunkelrot schattiert." _
      & vbCrLf & vbCrLf
    End If
    ' Tabellenfeldmarke mitkopieren (jetzt gibt es dadurch keine doppelte Tabellenfeldmarke mehr,
    ' und es wird dadurch die Formatierung übertragen.
    ' ********************
    ' In die Zwischenablage kopieren, wenn die Zelle irgendetwas enthält
    ' (wenn die Markierung leer ist, würde der Copy-Befehl zum Absturz führen):
    If Selection.Type > 1 Then   ' Auch ein alleinstehendes Bild muss erfasst werden!
       Selection.Copy
       Selection.CopyFormat
       '''''''''''''
       ''   ' Ausrichtung des letzten Absatzes der Zelle erfassen:
       ''   Selection.Collapse Direction:=wdCollapseEnd
       ''   lntAusrichtung = Selection.ParagraphFormat.Alignment
       '''''''''''''
       ' in die neue Tabelle kopieren:
       ''   Tabelle22.Cell(i, j).Range.ParagraphFormat.Alignment = lntAusrichtung
       Tabelle22.Cell(i, j).Range.Paste
       Tabelle22.Cell(i, j).Range.Shading.BackgroundPatternColor = lnShade
       '''''''''''''
    End If
  Next z
' ********************************************************************
' Kopfzeilen am Schluss der Tabelle neu zuweisen, zentrieren und abgrenzen
' ********************************************************************
' In die erste Tabellen-Zeile gehen:
  '' Selection.Tables(1).Rows(1).Select  ' Gibt Fehler bei vertikal verbundenen Zellen.
  Tabelle22.Select
  Selection.Collapse
' Kopfzeilen neu zuweisen und zentrieren:
  If K > 0 Then
    For i = 1 To K
        Selection.EndKey Unit:=wdRow
        Selection.HomeKey Unit:=wdRow, Extend:=True
        Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
        Selection.Rows.HeadingFormat = True
        Selection.ParagraphFormat.KeepWithNext = True
        If i > 1 Then
          Selection.Borders(wdBorderTop).LineWidth = wdLineWidth050pt
        End If
        Selection.MoveDown Unit:=wdLine, Count:=1
    Next i
  ' Kopfzeilen durch eine dicke Linie abgrenzen:
    ''Selection.EndKey Unit:=wdRow
    ''Selection.HomeKey Unit:=wdRow, Extend:=True
    Selection.Tables(1).Range.Rows(K + 1).Range.Select
    With Selection.Borders(wdBorderTop)
        .LineStyle = wdLineStyleSingle
        .LineWidth = wdLineWidth150pt
        .Color = wdColorBlack
    End With
  End If
' *********************
' Verbundene Zellen angeben:
  If C <> a * B Then
    Warnhinweis2 = "Achtung: Es gab verbundene oder geteilte Zellen, " _
```

```
        & "die von Hand wieder neu verbunden werden müssen." & vbCrLf & vbCrLf
  End If
' ****************************************
' kopierte Tabelle löschen:
  Tabelle21.Delete
  ' Markierung auf Einfügemarke reduzieren:
  Selection.Collapse
' ****************************************
' Neu erzeugte Tabelle etwas nach-formatieren:
  '  Gesamte Tabelle mittig auf die Seite:
  Tabelle22.Rows.Alignment = wdAlignRowCenter   ' (Tabelle zentrieren)
  With Tabelle22.Range.ParagraphFormat
    .SpaceBefore = 3
    .SpaceAfter = 3
    .SpaceAfterAuto = False
  End With
  Tabelle22.Range.Cells.VerticalAlignment = wdCellAlignVerticalCenter
' ***
' Tabelle an Satzspiegel des Dok1 anpassen:
  ' Damit keine fehlerhafte Silbentrennung auftritt, zuerst:
  Tabelle22.AutoFitBehavior (wdAutoFitContent)
 ' dann Tabelle an Satzspiegel der ursprünglichen Datei anpassen:
  Tabelle22.PreferredWidthType = wdPreferredWidthPoints
  Tabelle22.PreferredWidth = Satzspiegelbreite_Points
 ' Hier ein Wartebefehl einfügen für Word 2010,
 ' 0,1 Sekunden warten:
Dim tt As Single
   tt = Timer
   Do While Timer < tt + 0.3  'Sekunden
      DoEvents
   Loop
' Feste Zellenbreiten, die sich nicht von alleine ändern:
  Tabelle22.AutoFitBehavior (wdAutoFitFixed)
' ******************************************************************
' ursprüngliche Bildschirmansicht wiederherstellen usw.
' ******************************************************************
  Application.DisplayStatusBar = True
  ActiveWindow.View.Type = Ansichtstyp1
  Options.Pagination = True
  Application.ScreenUpdating = True
  Application.DisplayAlerts = wdAlertsAll
  '
  Application.ScreenUpdating = True
  With ActiveWindow
      .DisplayHorizontalScrollBar = True
      .DisplayVerticalScrollBar = True
  End With
' ****************
' Dok2 wieder sichtbar machen:
  Dok2.Windows(1).Visible = True
' *******************************************************************
' Anfangsmarkierung wieder herstellen:
  '' rng0.Select
  '' Markierung soll in der temporären Datei bleiben.
' Hinweis in der Statuszeile beenden:
  Application.StatusBar = "      "
' ********************
' Im Suchen/Ersetzen-Menü alles zurückstellen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
' ********************
' Zeitmessen für die Rechendauer:
```

```
        Zeit2 = Timer
        Zeitdauer = Zeit2 - Zeit1
        Zeitdauer = Round(Zeitdauer, 2)
'  Warteschleife, damit neue Tabelle fertig ist, bevor der Warnhinweis kommt:
        Do While Timer < Zeit2 + 0.4  'Sekunden
          DoEvents
       Loop
'  ******************************************************************
'  Zeitmessung ggf. mit Warnhinweis, wenn die Kopfzeilen nicht zählbar
'  oder verbundene Zellen vorhanden sind:
   MsgBox "Bitte prüfen Sie die neu erzeugte verbesserte Tabelle in der temporären Datei." _
      & vbCrLf & vbCrLf _
      & Warnhinweis1 & Warnhinweis2 & Warnhinweis3 _
      & "Die Rechenzeit war " & Zeitdauer & " Sekunden."
Exit Sub
'  ******************************************************************
'  Fehlermeldung bei Abbruch:
Zeile_Word_sichtbar:
    Word.Application.Visible = True
    Dok2.Windows(1).Visible = True
   ''If Err.Number > 0 Then
      MsgBox "Fehler: " & vbCrLf & "Fehlernummer: " & Err.Number _
           & vbCrLf & "Fehlerbeschreibung: " & Err.Description
   ''End If
   Err.Clear
End Sub
' ---------------------------------------------------------------------------------------------------
```

8.2 Tabellen drehen (transponieren)

Mit dem folgenden Makro wird eine Tabelle gedreht. Ich arbeite hier mit einer Zwischendatei. Von dort wird Zelle für Zelle der Zelleninhalt in eine neue gedrehte (transponierte) Tabelle übertragen. Für Testzwecke bleibt hier aber die alte Tabelle noch erhalten. Ferner habe ich hier für Testzwecke ein Zeitmessung eingebaut. Die Absatz-Formatvorlagen werden hier nicht mitkopiert (Tabellen-Steuerzeichen bleibt vom Kopieren ausgenommen). Ggf. sind die entsprechenden Programmzeilen zu löschen.

```
Sub Tabelle_transponieren()
' Erstellt von Dipl.-Ing. Hanna-Chris Gast, 11. März 2012, Stand 12. März 2012.
' Dieses Makro erzeugt eine neue Tabelle mit vertauschten Spalten und Reihen.
' Text-Farben bleiben erhalten.
  Dim doc1 As Word.Document  ' die zu bearbeitende Datei
  Dim doc2 As Word.Document  ' die vorübergehende Zwischen-Datei
  Dim a As Integer  ' Anzahl der Reihen
  Dim b As Integer  ' Anzahl der Spalten
  Dim z As Cell
  Dim i As Integer  ' jeweilige Reihe
  Dim j As Integer  ' jeweilige Spalte
' ***********************************************
'  Zeitmessen für die Rechendauer:
  Dim Zeit1
  Dim Zeit2
  Dim Zeitdauer
   Zeit1 = Timer
'  ***********************************************
  If Selection.Information(wdWithInTable) = False Then
     MsgBox "Der Cursor ist nicht in einer Tabelle!"
     Exit Sub
```

```
  End If
' *************************
' bisherige Tabelle kopieren und sich Datei merken:
  Selection.Tables(1).Select
  Selection.Copy
  Set doc1 = ActiveDocument
' ****************************
'  Spaltenanzahl und Reihenanzahl:
   Selection.Tables(1).Select
   a = Selection.Tables(1).Rows.Count
   b = Selection.Tables(1).Columns.Count
   '' MsgBox "Es gibt " & a & " Reihen und " & b & " Spalten."
'
'  Bildschirmanzeige unterdrücken und Hinweis in der Statuszeile:
   Application.StatusBar = "         Bitte warten - das Makro läuft."
   Application.ScreenUpdating = False
'  ****************************
'  In einem (Zwischen-)Dokument die Tabelle zwischenspeichern und  zellenweise zurückkopieren:
   Documents.Add DocumentType:=wdNewBlankDocument
   ' falls wie bei mir in der Normal-dot schon Name und Datum vorgesehen ist:
   Selection.WholeStory
   Selection.Paste
   Set doc2 = ActiveDocument
'  ****************************************
'  zurück in erste Datei:
   doc1.Activate
   '''  Neue Tabelle erstellen anstatt der bisherigen:
   '''   Selection.Tables(1).Delete
'  Neue Tabelle erstellen unterhalb der bisherigen:
   Selection.Collapse Direction:=wdCollapseEnd
   Selection.TypeParagraph
   Selection.TypeParagraph
   ActiveDocument.Tables.Add Range:=Selection.Range, NumRows:=b, NumColumns:=a, _
       DefaultTableBehavior:=wdWord9TableBehavior, AutoFitBehavior:=wdAutoFitFixed
   Selection.Tables(1).AutoFormat Format:=wdTableFormatGrid5, ApplyBorders:=True, _
      ApplyShading:=True, ApplyFont:=True, ApplyColor:=True, _
     ApplyHeadingRows:=False, ApplyLastRow:=False, ApplyFirstColumn:=False, _
     ApplyLastColumn:=False, AutoFit:=True
'  ******************************************************
'  Wieder in die Zwischendatei wechseln und dort zellenweise auslesen:
   doc2.Activate
   Selection.Tables(1).Select
   For Each z In Selection.Cells
     z.Select
      i = z.RowIndex
      j = z.ColumnIndex
     If InStr(Selection, ChrW(7)) > 0 Then
       Selection.SetRange Start:=Selection.Start, End:=Selection.Start + InStr(Selection, ChrW(7)) - 2
     End If
     ' MsgBox Selection
     If Selection.Type = 2 Then
         ' Wenn das Feld leer ist, würde der Copy-Befehl zum Absturz führen.
         Selection.Copy
         doc1.Activate
         Selection.Tables(1).Cell(j, i).Range.Paste
        doc2.Activate
     End If
   Next z
'  Zwischendatei ohne Speichern schließen:
   ActiveDocument.Close SaveChanges:=False
'  ************************
'  zurück in erste Datei (wenn keine anderen Word-Dateien offen sind, unnötig):
```

```
  doc1.Activate
' ************************
' Neu erzeugte Tabelle etwas formatieren:
  Selection.Tables(1).Rows.Alignment = wdAlignRowCenter
  With Selection.Tables(1).Range.ParagraphFormat
    .SpaceBefore = 3
    .SpaceAfter = 3
    .SpaceAfterAuto = False
  End With
  Selection.Tables(1).Range.Cells.VerticalAlignment = wdCellAlignVerticalCenter
' Tabellenbreite festlegen, sofern gewünscht:
    Selection.Tables(1).AutoFitBehavior (wdAutoFitContent)
  '  Selection.Tables(1).AutoFitBehavior (wdAutoFitFixed)
  '  Selection.Tables(1).PreferredWidth = 99
' *********************
  Application.StatusBar = "      "
' Bildschirmanzeige wieder anstellen:
  Application.ScreenUpdating = True
' Im Suchen/Ersetzen-Menü alles zurückstellen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
' **********************
 ' Zeitmessen für die Rechendauer:
      Zeit2 = Timer
      Zeitdauer = Zeit2 - Zeit1
      Zeitdauer = Round(Zeitdauer, 2)
 MsgBox "Die Rechenzeit war " & Zeitdauer & " Sekunden."
End Sub
```

Testtabelle vorher

Reihe 1 Spalte a	1b	1c	1d
Reihe 2 Spalte a	2b	2c	2d
Reihe3 Spalte a	3b	3c	3d

Testtabelle nachher

Reihe 1 Spalte a	Reihe 2 Spalte a	Reihe3 Spalte a
1b	2b	3b
1c	2c	3c
1d	2d	3d

8.3 Tabellenfortsetzung mit Nummerierung (i von n)

Das hier erstellte Makro ergänzt oben auf den Folgeseiten von mehrseitigen Tabellen die Tabellen-Nummer und eine Angabe, um die wievielte Seite der Tabelle es sich handelt.

Um die Tabelle nicht zu zerhacken, entwickelte ich hier eine Lösung mit **Textfeldern außerhalb des Satzspiegels**, welche die Tabelle selbst nicht verändern. Makro a96a für alle Tabellen des Dokuments bzw. Einzel-Prozedur a96c für eine einzelne Tabelle).

Der letzte oder vorletzte Absatz vor der Tabelle (die Tabellennummer mit Titel) muss mit dem Wort "Tabelle" oder "Table" oder "Tableau" beginnen. Für Anwendungen in anderer Sprache ist das Makro entsprechend abzuwandeln. Hinter dem Absatz mit dem Titel wird der Zusatz "(*1 von nn*)" ergänzt. Dieser Zusatz befindet sich in einem separaten Absatz (Formatvorlage "Standard") mit unsichtbarer Absatzmarke, damit er nicht auch im Inhaltsverzeichnis auftaucht.

Damit kein Seitenumbruch zwischen Titel und Tabelle gerät, wird jeweils ein Seitenumbruch oberhalb des Textfeldes "(# von nn)" ergänzt (Seitenumbruch oberhalb des Ankerpunkts).

Nach Einfügen oder Entfernen von Text vor einer Tabelle oder von Tabellenzeilen (Reihen) in einer Tabelle müssen die hier erzeugten Textfelder mit der Fortsetzungs-Nummerierung über den Folgeseiten wieder entfernt werden. Ebenso die Eigenschaft "Seitenumbruch oberhalb". Dafür dient das zweite Makro (a96b). Dieses wird auch von der Rahmenprozedur a96a vor der Neu-Nummerierung aufgerufen, so dass mit einem Makro im gesamten Dokument die Tabellenfortsetzungs-Nummerierung auf den neuesten Stand gebracht werden kann.

Ein Makro, welches die Tabellen seitenweise zerstückelt, aber dafür ohne Textfelder auskommt, kann ich auf Anfrage jederzeit liefern. Auch bei diesem Makro können alle Zerstückelungen mit dem a96b-Makro wieder rückgängig gemacht werden.

Offene Probleme:
Angaben zwischen Titel und Tabelle wie "Maße in Millimeter" müssen vor dem Start des Makros in eine Tabellenkopfzeile mit unsichtbaren Rahmenlinien gepackt werden, wenn sie wiederholt werden sollen.

Bei mehreren Absätzen zwischen Titel und Tabellen (was sehr selten ist) wird abgebrochen.

Es kann in seltenen Fällen ein Teil der Kopfzeile verdeckt werden, wenn die Seitenzahlen oben mittig sind und der Abstand unter der Seitenzahl zu gering ist.

Bei vertikal verbundenen Zellen beim Seitenumbruch kann es Probleme geben.

Das fertige Makro-Modul für Tabellenfortsetzungen

```
Option Explicit


Sub a96a_alle_Tabellen_fortgesetzt_Textfelder()
' Erstellt von Dipl.-Ing. Hanna-Chris Gast, 15. Februar 2012, 4. Juli 2014, 21. Nov. 2014.
' Diese Rahmenprozedur ruft für jede Tabelle das Makro für die Tabellenfortsetzung auf.
' Geeignet für Word 2000 bis Word 2010.
'
' Die Tabellen werden nicht zerstückelt, es werden nur Textfelder im oberen Seitenrand ergänzt.
' *****************
Dim tbl As Word.Table
Dim Rng0 As Range
' *****************
  If ActiveDocument.Tables.Count = 0 Then
    MsgBox "Das Dokument enthält keine Tabellen."
    Exit Sub
  End If
' *************
  Set Rng0 = Selection.Range
' *************
' Ggf. vorhandene Tabellenseiten-Nummerierung löschen.
  Call a96b_Tabellenfortsetzungs_Nummerierung_loeschen
' ************
  For Each tbl In ActiveDocument.Tables
    tbl.Select
```

```
    Call a96c_Einzel_Tabelle_fortgesetzt_Textfelder
  Next
'  Markierung wiederherstellen:
  Rng0.Select
End Sub
' ----------------------------------------------------------------------------------------------------------


Sub a96c_Einzel_Tabelle_fortgesetzt_Textfelder()
' Hanna-Chris Gast, 2012, 2014, 2015, Stand 5. November 2015.
' Geeignet für Word 2000 bis Word 2010, Windows XP und Windows7.
' http://www.siebener-kurier.de/chris-Aufsaetze
'
' Dieses Makro ergänzt hinter dem Tabellentitel der markierten Tabelle  den Zusatz "(1 von nn)"
' und auf den Folgeseiten der markierten Tabelle oberhalb des Satzspiegels
' die Tabellennummer mit relativer Seitenzahl in Textfeldern "Tabelle # (i von nn)".
' Diese Makroversion funktioniert unabhängig von den voreingestellten Rändern.
' Die Tabelle mit Titel wird wegen dem Titelzusatz auf der 1. Seite um eine Tabellenzeile länger.
'
' Dieses Makro berücksichtigt Tabellen mit 2 Titel-Absätzen,
' etwa mit "Maße in Millimeter" unter dem eigentlichen Titel.
' Das Makro funktioniert nicht, wenn der Titel erst der drittletzte Absatz ist.
'
' Werden in der Tabelle nachträglich Tabellenzeilen (Reihen) ergänzt oder gelöscht, wird
' eine neue Seiten-Aufteilung nötig. Hierfür dient die Prozedur "a96b" zum Löschen aller
' Fortsetzungsnummern und die Neuerstellung.  Der Tabellennummern-Zusatz der ersten
' Seite "(1 von nn)" wird deshalb mit einer Textmarke versehen, damit man ihn nachher
' wieder besser löschen kann. Bei den Folgeseiten werden einfach die betreffenden Textfelder
' (Shapes) gelöscht.
'
' Tabellen mit unsichtbarem Rahmen werden ignoriert.
' Es wird die Möglichkeit gegeben, Tabellen ohne brauchbaren Titel per "ok" zu überspringen.
'
' Wenn sich durch das Einfügen von "1 von nn" unter dem Tabellen-Titel die Seitenzahl ändert,
' wird dies jetzt berücksichtigt.
'
' Damit der Titelzusatz nicht im Inhaltsverzeichnis auftauchen kann, wird er auf der ersten
' Tabellenseite in einen separaten Absatz unter dem Titel gepackt.
'
' Die Absatzmarke zwischen Tabellentitel und dem Zusatz "(1 von nn)" wird hier unsichtbar
' gemacht. Sie wird aber erst unsichtbar, wenn man die Steuerzeichen ausblendet:
'   " ActiveWindow.ActivePane.View.ShowAll = False"  bzw. per Tastatur "Strg + Shift + *"
' Hilfsweise wird sie (nur auf dem Bildschirm) rot eingefärbt.
'
' Bei verbundenen Zellen am Seitenumbruch wird Zellenweise weitergegangen, bis sich
' der Cursor nicht mehr unten auf einer Seite befindet.
' Trotzdem versagt dieses Makro noch bei einigen vertikal verbundenen Tabellenzellen!
' ******************************************************
Dim i As Integer            ' laufende Nummer in Schleifen
Dim j As Integer            ' absolute Tabellennummer
Dim k As Integer            ' Anzahl der Kopfzeilen
Dim LNN As Integer          ' Anzahl der Zeilen
Dim nn As Integer           ' Gesamt-Seitenzahl der Tabelle
Dim IntAbfrage As Integer
"Dim Seitenzahl(2) As Integer  ' Anfangs- und Endseite der Tabelle
Dim Tabellennummer As String  ' die Tabellennummer mit dem Wort "Tabelle".
Dim Abstand_oben As Integer   ' Textblock-Abstand vom oberen Papierrand
Dim rng0 As Range           ' Ranges für Markierzwecke
Dim rng2 As Range           ' Bereich: "(1 von nn)"
Dim Rng3 As Range           ' Tabellentitel mit Nummer (max. 20 Zeichen)
Dim RngTabelleAnfang As Range
Dim RngTabelleEnde As Range
```

```
Dim RngKopf As Range          ' Bereich der Kopfzeilen der Tabelle
Dim aTable As Word.Table
'  ****************************************
'  Abfrage, ob Cursor in Tabelle:
  If Selection.Information(wdWithInTable) = False Then
    MsgBox "Es ist keine Tabelle markiert!"
    Exit Sub
  End If
'  ****************************************
  Set aTable = Selection.Tables(1)
'  Tabellenseiten zählen:
  aTable.Select
  "Seitenzahl(2) = Selection.Information(wdActiveEndPageNumber)
  "Selection.Collapse
  "Seitenzahl(1) = Selection.Information(wdActiveEndPageNumber)
  "nn = Seitenzahl(2) - Seitenzahl(1) + 1
  Set RngTabelleAnfang = Selection.Range
  Set RngTabelleEnde = Selection.Range
  RngTabelleEnde.Collapse Direction:=wdCollapseEnd
  RngTabelleAnfang.Collapse Direction:=wdCollapseStart
  nn = RngTabelleEnde.Information(wdActiveEndPageNumber) _
      - RngTabelleAnfang.Information(wdActiveEndPageNumber) + 1
  If nn < 2 Then
    ' MsgBox "Die Tabelle geht nicht über mehrere Seiten."
    Exit Sub
  End If
'  ***************************************
'  Tabellen mit unsichtbaren Linien ignorieren:
  If aTable.Borders.OutsideLineStyle = wdLineStyleNone Then
    ' MsgBox "Dies ist keine normale Tabelle."
    Exit Sub
  End If
'  ****************************************
'  Absolute Word-interne Tabellennummer bestimmen für die Namen der Textmarken:
  For j = 1 To ActiveDocument.Tables.Count
    "If aTable.Range.InRange(ActiveDocument.Tables(j).Range) Then
    If Selection.InRange(ActiveDocument.Tables(j).Range) Then
      ' j ist die absolute Tabellennummer der ausgewählten Tabelle.
      Exit For
    End If
  Next j
  Debug.Print j
'  ***********************************************************************
'  Die eigentliche Prozedur
'  ***********************************************************************
'  Zeilenumbruch innerhalb einer Tabellenzeile verhindern:
  Selection.Rows.AllowBreakAcrossPages = False
  ' funktioniert leider bei vertikal verbundenen Zellen unbefriedigend!
'  ***************************
'  Kopfzeilen zählen und zusammenhalten:
  aTable.Range.Select
  k = 0
  LNN = Selection.Tables(1).Rows.Count
  " MsgBox "Die Tabelle hat " & n & " Zeilen."
'  ***************************
'  In die erste Zeile gehen:
  " Selection.Tables(1).Rows(1).Select  ' Gibt Fehler bei vertikal verbundenen Zellen.
  "Selection.Tables(1).Select
  Selection.Collapse
  Set RngTabelleAnfang = Selection.Range
'  Kopfzeilen zählen:
  For i = 1 To LNN
```

```
      If Selection.Information(wdWithInTable) = False Then
        MsgBox "Es dürfen nicht alle Reihen als Kopfzeile markiert sein!", vbCritical
        Exit Sub
      ElseIf Selection.Rows.HeadingFormat = True Then
        ' jeweilige Kopfzeile markieren:
        '' Selection.Rows(1).Select
        ' ergab Fehler, wenn dort in der Tabelle vertikal verbundene Zellen sind.
        Selection.EndKey Unit:=wdRow
        Selection.HomeKey Unit:=wdRow, Extend:=True
        ' zur nächsten Reihe:
        Selection.MoveDown Unit:=wdLine, Count:=1
      Else
        ' Dies ist keine Kopfzeile mehr. Das Kopfzeilenende ist dann:
        k = Selection.Information(wdEndOfRangeRowNumber) - 1
        Set RngKopf = Selection.Range  ' = Kopfzeilenende + eine Zeile
        RngKopf.SetRange Start:=RngTabelleAnfang.Start, End:=RngKopf.End - 3
        RngKopf.Select
        Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
        With Selection.ParagraphFormat
          .WidowControl = True
          .KeepWithNext = True
          .KeepTogether = True
        End With
        '''''''''''''
        Exit For
      End If
    Next i
    If i > LNN Then
      MsgBox "Es dürfen nicht alle Reihen als Kopfzeile markiert sein!", vbCritical
      Exit Sub
    End If
'  ********************************************************************
'  Tatsächliche Tabellennummer anhand des Tabellentitels feststellen
'  *************
'  Tabellentitel von der Tabellennummer möglichst sauber abtrennen:
    Selection.Collapse
    Selection.MoveUp Unit:=wdParagraph, Count:=1
    Selection.Expand Unit:=wdParagraph  ' (ganzer Absatz vor der Tabelle)
    ' Wenn 1. Zeichen der Markierung ein Seitenumbruch und ein anderes Steuerzeichen,
    ' dann dieses aus der Markierung herausnehmen:
    If AscW(Left(Selection, 1)) < 32 Then Selection.MoveStart Unit:=wdCharacter, Count:=1
'
'  Abfrage, ob am Anfang der Markierung das Wort "Tabelle" steht:
    If InStr(LCase(Selection), "tabelle") <> 1 _
    And InStr(LCase(Selection), "table") <> 1 _
    And InStr(LCase(Selection), "tableau") <> 1 Then
      '  (Absatz beginnt nicht mit "Tabelle" bzw. "Table" oder "Tableau")
      '  Vielleicht steht hier "Maße in Millimeter" oder Ähnliches?
      If InStr(Selection.Range.Paragraphs(1).Range.Text, " in ") > 0 _
      And Selection.Range.Paragraphs(1).Range.Words.Count < 5 Then
        ' "Maße in Millimeter oder ähnliches":
        ActiveWindow.ScrollIntoView Selection.Range
        IntAbfrage = _
            MsgBox("Die Zeile: " & vbCrLf & "   " & Selection.Range.Paragraphs(1).Range.Text _
            & "sollte vielleicht auf den Folgeseiten wiederholt werden?" & vbCrLf _
            & vbCrLf & "Wollen Sie trotzdem weitermachen (OK)? " _
            & vbCrLf & "Oder wollen Sie das Makro abbrechen? " _
            & "Sie könnten diesen Absatz dann vielleicht (von Hand) in eine unsichtbare " _
            & "Tabellen-Kopfzeile packen (Abbrechen).", vbOKCancel)
        If IntAbfrage <> 1 Then
          ' Makro beenden.
          End
```

```
      End If
    Else
      ' Kein "Maße im Millimeter" oder Ähnliches. Absatz ignorieren.
    End If
    '  ***
    '  Den zweitletzten Absatz über der Tabelle durchsuchen:
    Selection.MoveUp Unit:=wdParagraph, Count:=2
    Selection.Expand Unit:=wdParagraph  ' (2. Absatz vor der Tabelle)
    '  Wenn 1. Zeichen der Markierung ein Seitenumbruch und ein anderes Steuerzeichen,
    '  dann dieses aus der Markierung herausnehmen:
    If AscW(Left(Selection, 1)) < 32 Then Selection.MoveStart Unit:=wdCharacter, Count:=1
    ' Wenn auch dieser Absatz nicht mit "Tabelle" bzw. "Table" oder "Tableau" beginnt,
    ' dann die Prozedur für die Tabelle oder das gesamte Makro abbrechen:
    If InStr(LCase(Selection), "tabelle") <> 1 _
    And InStr(LCase(Selection), "table") <> 1 _
    And InStr(LCase(Selection), "tableau") <> 1 Then
      ' ***
      IntAbfrage = MsgBox("Die Tabellennummer ist hier unklar. " & vbCrLf _
          & "Es muss hier das Wort ""Tabelle"" bzw. ""Table"" vorkommen." & vbCrLf _
          & vbCrLf & "Wollen Sie bei der nächsten Tabelle weitermachen (OK)? " _
          & vbCrLf & "(Abbrechen = Makro abbrechen).", vbOKCancel)
      If IntAbfrage = 1 Then
        ' Unterprozedur verlassen zur nächsten Tabelle:
        Exit Sub
      Else
        ' Makro gänzlich abbrechen:
        End
      End If
      ' ***
    End If
  End If
'  ***********************************************************
'  Es ist jetzt der Absatz mit der Tabellennummer markiert.
'  ***********************************************************
'  Abfrage, ob automatische Tabellen-Nummerierung vorliegt:
  ' Dazu Titellänge ggf. auf 20 Zeichen begrenzen (um ggf. Felder im Titel auszulassen):
  If Len(Selection) > 20 Then
    Selection.Collapse
    Selection.MoveRight Unit:=wdCharacter, Count:=20, Extend:=wdExtend
  End If
  If Selection.Fields.Count > 0 Then
    ' bei automatischer Tabellennummerierung das letzte Feld mit-erfassen:
    Selection.Fields(Selection.Fields.Count).Select
    Selection.MoveRight Unit:=wdCharacter, Count:=1
    Selection.MoveStart Unit:=wdParagraph, Count:=-1
    If AscW(Left(Selection, 1)) < 32 Then Selection.MoveStart Unit:=wdCharacter, Count:=1
    Tabellennummer = Selection.Text
  Else
    ' bei Tabellennummerierung von Hand:
    Set rng0 = Selection.Range
    Dim x1
    Dim x2
    ' Abgrenzen zwischen Nummer und Titel vermutlich Leerzeichen oder Doppelpunkt:
    x1 = InStr(8, Selection, " ")
    If x1 = 0 Or x1 > 15 Then x1 = 16  ' (Begrenzung einführen)
    x2 = InStr(8, Selection, ":")
    If x2 = 0 Or x2 > 18 Then x2 = x1   ' (Begrenzung, falls irgendwo im Titel ein Doppelpunkt ist)
    If Len(Selection) > x2 Then
      rng0.SetRange Start:=Selection.Range.Start, End:=Selection.Range.Start + x2 - 1
      ' Ergäbe zu kleine Werte, wenn Felder enthalten sind!
      rng0.Select
    End If
```

```
    If Left(Selection, 1) = ChrW(13) Then Selection = Mid(Selection, 2)
    If Right(Selection, 1) = ChrW(13) Then Selection.MoveEnd Unit:=wdCharacter, Count:=-1
    '' Selection.Copy
    Tabellennummer = Selection.Text
  End If
  Set Rng3 = Selection.Range   ' (Das Wort "Tabelle" mit der Tabellennummer)
' ******************************************************************************
' An die Tabellennummer (ggf. hinter Titel) auf der ersten Tabellenseite
' einen Textzusatz "(1 von nn)" mit nummerierter Textmarke ergänzen:
' ******************************************************************************
  Selection.MoveDown Unit:=wdParagraph, Count:=1
  Selection.MoveLeft Unit:=wdCharacter
  Set rng0 = Selection.Range      ' künftiger Textmarken-Anfangspunkt
  ' Damit dieser Zusatz nicht ins Inhaltsverzeichnis kommen kann,
  ' hierfür einen neuen Absatz bilden:
  Selection.ParagraphFormat.SpaceAfter = 0
  Selection.TypeParagraph
  ' (Der neue Absatz hat die Formatvorlage "Standard")
  Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
  Selection.ParagraphFormat.SpaceBefore = 0  ' (eigentlich überflüssig)
  Selection.ParagraphFormat.SpaceAfter = 6   ' (Abstand unterhalb)
  ' Text des Zusatzes kursiv und mager:
  Selection.Font.Italic = True
  Selection.Font.Bold = False
  ' ******************************************************
  ' nn kann sich geändert haben durch den Zusatz einer Zeile unter dem Titel:
  nn = RngTabelleEnde.Information(wdActiveEndPageNumber) _
      - RngTabelleAnfang.Information(wdActiveEndPageNumber) + 1
  ' Wenn die Tabelle jetzt genau am Seitenumbruch endet, 1 Seite abziehen:
  ''  If RngTabelleEnde.Information(wdFirstCharacterLineNumber) = 1 Then
  If RngTabelleEnde.Information(wdVerticalPositionRelativeToTextBoundary) = 0 Then
    nn = nn - 1
  End If
  ' Bei nur noch einer Seite abbrechen:
  If nn < 2 Then
    ' Tabelle hat nur noch eine Seite. Die Seitenumbrüche haben sich zu stark verschoben!
    ''MsgBox "Diese Tabelle geht jetzt nicht mehr über mehrere Seiten. Bitte prüfen!", _
    ''vbCritical
    ' angefangene Zeile mit Absatzmarke löschen:
    Selection.MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdExtend
    Selection.Delete Unit:=wdCharacter, Count:=1
    '  Damit kein Seitenumbruch zwischen Titel und Tabelle gerät oder in die Tabelle:
    Selection.SetRange Start:=Rng3.Start, End:=aTable.Range.End
    With Selection.ParagraphFormat
      .WidowControl = True
      .KeepWithNext = True
      .KeepTogether = True
    End With
    '
    IntAbfrage = _
    MsgBox("Diese Tabelle geht jetzt nicht mehr über mehrere Seiten." & vbCrLf _
    & vbCrLf & "Wollen Sie trotzdem mit der nächsten Tabelle weitermachen (OK)? " _
    & vbCrLf & "Oder wollen Sie das Makro gänzlich abbrechen? (Abbrechen).", vbOKCancel)
    If IntAbfrage <> 1 Then
      End
    Else
      Exit Sub
    End If
  End If
  ' ******************************************************
  Selection.Font.Hidden = False  ' unsichtbar erst am Ende der Prozedur!
  Selection.TypeText Text:=" (1" & ChrW(160) & "von" & ChrW(160) & nn & ")"
```

```
  ' Textmarke ergänzen (damit man diesen Zusatz später wieder leichter löschen kann):
  Set rng2 = Selection.Range
  rng2.SetRange Start:=rng0.Start + 1, End:=rng2.End + 1
  With ActiveDocument.Bookmarks
    .Add Range:=rng2, Name:="Tabellennummer_absolut" & j & "_" & (100 * Timer)
    .DefaultSorting = wdSortByName
    .ShowHidden = False
   End With
' ************************************
' Damit kein Seitenumbruch zwischen Titel und Tabelle gerät, 3 Absätze verbinden:
  Rng3.Select
  Selection.MoveDown Unit:=wdLine, Count:=3, Extend:=wdExtend
  With Selection.ParagraphFormat
    .WidowControl = True
    .KeepWithNext = True
    .KeepTogether = True
  End With
  Selection.Collapse
' ********************************************************************************
' Auf den Folgeseiten über den Tabellen-Fortsetzungen je ein Textfeld mit
' Tabellennummer und relativer Seitenzahl einfügen "Tabelle X (i von nn)":
' ********************************************************************************
' Das Textfeld soll oberhalb des oberen Randabstands liegen:
  Abstand_oben = Selection.PageSetup.TopMargin   ' 71 bei normalen Rändern.
  Abstand_oben = Abstand_oben - 20   ' Textfeld soll oberhalb vom Rand sein.
' ************************************
' Abweichungen für einige spezielle Dokumente:
  If ActiveDocument.AttachedTemplate <> NormalTemplate Then
  ' Bei ISO-Kanal ist das Textfeld weiter unten anzuordnen:
    If Selection.Range.PageSetup.Orientation = wdOrientPortrait Then
      If Abstand_oben < 22.5 Then
        Abstand_oben = 83
      ElseIf Abstand_oben < 42 Then
        ' Bei DKE mit "hochkant" ebenfalls Abweichung nötig (Randabstand = 60):
        Abstand_oben = 42
      End If
    Else
     If Abstand_oben < 22.5 Then
       Abstand_oben = 63
     End If
    End If
  End If
' ************************************
  For i = 2 To nn
    ' Gehe zur nächsten Seite:
    Selection.GoTo What:=wdGoToPage, Which:=wdGoToNext, Count:=1, Name:=""
    ' Kontrolle wegen Seitenübergreifender Zellen:
    Selection.Cells(1).Select
   '' Do While Selection.Information(wdVerticalPositionRelativeToTextBoundary) > 0
    ' Ärger bei verbundenen Zellen:
    Do While Selection.Information(wdVerticalPositionRelativeToPage) > 400
      'Selection.MoveDown Unit:=wdRow
      ' Selection.MoveDown Unit:=wdParagraph, Count:=3
      Selection.MoveRight Unit:=wdCell
    Loop
    ' Abstand_oben = Selection.Information(wdVerticalPositionRelativeToPage)
    '' ist nicht brauchbar, da die Kopfzeilen mit unbekannter Höhe dazwischenliegen.
    ' ********************************
    ' Verschiebung der künftigen Ankerzeile nach oben verhindern:
    Selection.ParagraphFormat.PageBreakBefore = True
    ' ********************************
    ' Textfeld erzeugen mit angepasstem Abstand von oben:
```

```
    If Selection.Range.PageSetup.Orientation = wdOrientLandscape Then
      ' Querformat:
      ActiveDocument.Shapes.AddTextbox(msoTextOrientationHorizontal, _
                  327#, Abstand_oben, 150.75, 18).Select
    Else
      ' Hochkant-Format:
      ActiveDocument.Shapes.AddTextbox(msoTextOrientationHorizontal, _
                  223#, Abstand_oben, 150.75, 18).Select
    End If
    ' (ein breiteres Textfeld würde ggf. zuviel von den Kopfzeilen verdecken)
    Selection.ShapeRange.Line.Visible = msoFalse   ' Rahmen unsichtbar
    ' Neues Textfeld erzeugen:
    ActiveDocument.Shapes(ActiveDocument.Shapes.Count).Select
    Selection.ShapeRange.TextFrame.TextRange.Select
    ' ****
    'In das Textfeld Text einfügen:
    Selection.Style = "Standard"
    Selection.Font.Bold = True
    Selection.Font.Name = "Arial"
    Selection.Font.Size = 10    '  (oder Textfeld höher machen)
    Selection.TypeText Text:=Tabellennummer
    ''   Selection.PasteSpecial DataType:=wdPasteText
    Selection.Font.Bold = False
    Selection.Font.Italic = True
    Selection.TypeText Text:=" (" & i & " von " & nn & ")"
    Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
    ' ****
    ' Zurück in die Tabelle gehen:
    ''  Selection.MoveDown Unit:=wdScreen, Count:=1  ' (zu weit unten)
     Selection.ShapeRange.Anchor.Select
     Selection.Collapse
     ' Hier an dieser Stelle funktionierten die Befehle mit "Absätze verbinden"
     ' nicht, offenbar wegen dem Anker innerhalb der Markierung.
     '' Application.Run MacroName:="Normal.Allgemeines.Absätze_verbinden"
  Next i
'  ********************************************************************
'  Die Absatzmarke hinter dem Titel unsichtbar machen (14. März 2015)
'  und als Hinweis für die Ansicht mit Steuerzeichen auf den Bildschirm rot-färben:
  rng2.Select
  Selection.Collapse
  Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
  Selection.Font.Hidden = True
  Selection.Font.ColorIndex = wdRed
  Selection.MoveRight Unit:=wdCharacter, Count:=1
'  ********************************************************************
'  Cursor zurück an den Tabellenkopf (Tabellennummer) stellen:
  Rng3.Select
End Sub
' ------------------------------------------------------------------------------------------------------------------



Sub a96b_Tabellenfortsetzungs_Nummerierung_loeschen()
' Hanna-Chris Gast, 2012, 2014, 16. März 2015, 10. April 2015, 17. Juli 2015.
'
' Dieses Makro löscht alle von den beiden Makros für Tabellenfortsetzung erzeugten
' Tabellentitel-Ergänzungen, die eine spezielle Textmarke tragen.
' Ferner werden alle "Seitenumbrüche oberhalb" aus den sichtbaren Tabellen entfernt.
Dim i As Integer
Dim Rng0 As Range  ' Cursorstelle am Anfang
  Set Rng0 = Selection.Range
```

```
' ********************************************************************
' Titelfortsetzungen löschen, die aus Makro T96p stammen
' (Tabellenfortsetzung mit Zerstückelung der Tabelle):
' ********************************************************************
' Titel-Zusätze anhand der Textmarken löschen (rückwärts zählen):
' (die Tabellenfortsetzungen wurden vom Makro T96p vorher mit Textmarken versehen):
  If ActiveDocument.Bookmarks.Count > 0 Then
    For i = ActiveDocument.Bookmarks.Count To 1 Step -1
      If ActiveDocument.Bookmarks(i).Name Like "Tabellenfortsetzung_*" Then
        ' Absatzmarke ggf. wieder sichtbar machen:
        ActiveDocument.Bookmarks(i).Range.Select
        Selection.MoveStart Unit:=wdCharacter, Count:=-1
        Selection.Font.Hidden = False
        Selection.Font.ColorIndex = wdAuto
        Selection.ParagraphFormat.Reset
        ' ***
        ActiveDocument.Bookmarks(i).Select
        ActiveDocument.Bookmarks(i).Range.Delete
        If Selection.Style = ActiveDocument.Styles(wdStyleNormal) _
        And Asc(Selection) = 13 Then Selection.Delete
      End If
    Next i
  End If
  ' Tabellenfortsetzung mit Tabellenkopf-Wiederholung:
  If ActiveDocument.Bookmarks.Count > 0 Then
    For i = ActiveDocument.Bookmarks.Count To 1 Step -1
      If ActiveDocument.Bookmarks(i).Name Like "Tabellenfortsetzung_*" Then
        ActiveDocument.Bookmarks(i).Range.Delete
      End If
    Next i
  End If
' ***************************
' "Seitenumbruch oberhalb" in allen sichtbaren Tabellen löschen:
  For i = 1 To ActiveDocument.Tables.Count
    If ActiveDocument.Tables(i).Borders.OutsideLineStyle <> wdLineStyleNone Then
      ActiveDocument.Tables(i).Range.ParagraphFormat.PageBreakBefore = False
    End If
  Next i
' ***************************
' Cursor-Position wiederherstellen:
  Rng0.Select
' ********************************************************************
' ********************************************************************
' Tabellenfortsetzungen löschen, die aus Makro a96c stammen
' (Tabellenfortsetzung mit Textfeldern):
' ********************************************************************
' Dieses Makro löscht alle Textfelder, die als Text "T*#... von #" enthalten,
' und alle Tabellentitel-Ergänzungen,
' die eine spezielle Textmarke "Tabellennummer_absolut#*" tragen.
' Ferner werden die "Seitenumbrüche oberhalb" aus den (sichtbaren) Tabellen entfernt.
' *****
' Textfelder (Shapes) anhand des Textinhalts löschen:
  For i = ActiveDocument.Shapes.Count To 1 Step -1
    If ActiveDocument.Shapes(i).Type = msoTextBox Then  ' 17
      Set Rng0 = ActiveDocument.Shapes(i).TextFrame.TextRange
      If Rng0.Text Like "Ta*# von #*" _
      Or Rng0.Text Like "Ta*# of #*" _
      Or Rng0.Text Like "Ta*# sur #*" Then
        ActiveDocument.Shapes(i).Delete
      End If
    End If
  Next i
```

```
'  ******
  Set Rng0 = Selection.Range
' Titel-Zusätze anhand der Textmarken löschen:
' (die Titelszusätze wurden vom Makro a96c vorher mit Textmarken versehen):
  If ActiveDocument.Bookmarks.Count > 0 Then
    For i = ActiveDocument.Bookmarks.Count To 1 Step -1  '(rückwärts zählen)
      If ActiveDocument.Bookmarks(i).Name Like "Tabellennummer_absolut#*" Then
        ' Absatzmarke ggf. wieder sichtbar machen (14. März 2015):
        ActiveDocument.Bookmarks(i).Range.Select
        Selection.MoveStart Unit:=wdCharacter, Count:=-1
        Selection.Font.Hidden = False
        Selection.Font.ColorIndex = wdAuto
        Selection.ParagraphFormat.Reset
        ' ***
        ActiveDocument.Bookmarks(i).Select
        ActiveDocument.Bookmarks(i).Range.Delete
        If Selection.Style = ActiveDocument.Styles(wdStyleNormal) _
        And Asc(Selection) = 13 Then Selection.Delete
      End If
    Next i
  End If
'  "Seitenumbruch oberhalb" in allen Tabellen löschen:
    For i = 1 To ActiveDocument.Tables.Count
      If ActiveDocument.Tables(i).Borders.OutsideLineStyle <> wdLineStyleNone Then
        ActiveDocument.Tables(i).Range.ParagraphFormat.PageBreakBefore = False
      End If
    Next i
'  ********************************************************************
'  Cursor-Position wiederherstellen:
  Rng0.Select
End Sub
' ---------------------------------------------------------------------------------------------------
```

9 Ausmisten von lästigen Formatvorlagen ("Styles")

Das folgende Makro zum Ausmisten von Formatvorlagen gleicht alle Formatvorlagen ("Styles" der zugehörigen Dokumentvorlage ("Dot-Datei", englisch "Template") an und löscht alle Formatvorlagen, die in der Dokumentvorlage nicht vorhanden sind. Dieses Makro habe ich für Word 2003 entwickelt, funktioniert aber von Word 2000 bis Word 2010. Wenn man in Word 2003 die automatische Erzeugung von Formatvorlagen nicht deaktiviert, entstehen immer mehr Formatvorlagen, und wenn man fremde Texte unachtsam in eine Word-Datei einfügt, fängt man sich leider oft fremde Formatvorlagen ein.

Es empfiehlt sich, die Absatzformatierung aller Überschriften anschließend zu resetten. Dagegen die Formatvorlage "Standard" sollte keinesfalls resettet werden.

```
Sub a7a_Formatvorlagen_an_Dokumentvorlage_angleichen()
' Dipl.-Ing. Hanna-Chris Gast, 15. Juni 2011, 7. Juli 2011, Stand 16. April 2012.
' Soll alle Formatvorlagen ("Styles") löschen, die nicht in der Dokumentvorlage (der ".dot-Datei") sind,
' und alle übrigen Formatvorlagen an die Dot-Datei angleichen.
' Funktioniert in Word 2000 bis Word 2010.
  Dim doc1 As Word.Document  ' die zu bearbeitende Datei
  Dim doc2 As Word.Document  ' die zugehörige dot-Datei
    Set doc1 = ActiveDocument
  Dim DotStylesListe As String:    DotStylesListe = ""
  Dim astyle As Style
' ********************
' Zuerst alle Formatvorlagen aus dem "Attached Template" (zugehörige Dokumentvorlage)
' in einem String (Zeichenfolge) auflisten.
  Documents.Open FileName:=ActiveDocument.AttachedTemplate.Path _
        & Application.PathSeparator & ActiveDocument.AttachedTemplate, ReadOnly:=True
  Set doc2 = ActiveDocument
  '
  For Each astyle In ActiveDocument.Styles
    If astyle.InUse = True Then
      DotStylesListe = DotStylesListe & astyle.NameLocal & "   "
    End If
  Next
  doc2.Close SaveChanges:=False
  doc1.Activate
'''  MsgBox DotStylesListe
Dim Abfrage
Dim MsgBoxAnzeige
  MsgBoxAnzeige = "Nur die folgenden Formatvorlagen aus der Templatedatei ''' _
        & ActiveDocument.AttachedTemplate & "' sind zulässig. " _
        & vbCrLf & "Wollen Sie diese übernehmen?" & vbCrLf & vbCrLf & DotStylesListe
  Abfrage = MsgBox(MsgBoxAnzeige, vbYesNo + vbDefaultButton2, "Styles ausmisten")
  ' Bei "Ja" kommt "6" heraus, bei "Nein" kommt "7" heraus.
  If Abfrage <> 6 Then Exit Sub
' *****************
' Im aktuellen Dokument alle verfügbaren Styles (Formatvorlagen) löschen,
' die nicht in der Liste "DotStylesListe" sind, und die übrigen aktualisieren:
  On Error Resume Next
  For Each astyle In ActiveDocument.Styles
    If astyle.InUse = True Then
      If InStr(DotStylesListe, astyle.NameLocal) = 0 Then
        astyle.Delete
      Else
        Application.OrganizerCopy _
          Source:=ActiveDocument.AttachedTemplate.FullName, _
          Destination:=ActiveDocument.FullName, _
```

```
        Name:=astyle, _
        Object:=wdOrganizerObjectStyles
      End If
    End If
  Next astyle
  On Error GoTo 0
   ' Eventuell anschließend alle Querverweise aktualisieren:
   '' Call a2_alle_Felder_aktualisieren_mit_Fehlermeldung_gründlich
   '  Anschließend alle Überschriften resetten
   Call a7c_Überschriften_Format_resetten
End Sub
```

```
Sub a7c_alle_Überschriften_Format_resetten()
' H. Chris Gast, 30. März 2012, Stand 15. April 2012.
' löscht Nummerierungen der Überschriften, aber nicht Farb-Markierungen.
' Funktioniert auch für die Anhangsüberschriften "Anhang A", "a.2", "a.3" usw.
Dim para As Paragraph
   For Each para In ActiveDocument.Paragraphs
    'MsgBox astyle
     If Left(para.Style, 11) = "Überschrift" _
     Or LCase(Left(para.Style, 1)) = "a" Then
  ''     MsgBox para
       para.Range.ParagraphFormat.Reset  ' löscht ggf. falsche Nummerierung
     '  para.Range.Font.Reset                  '  würde Exponenten und Indizes zerstören.
     End If
   Next para
End Sub
```

10 Seitenzahlen einfügen

Problematisch ist das Einfügen von Seitenzahlen, wenn es viele "Abschnittswechsel" ("Sections") gibt. Leicht bleibt auch einmal eine Seite ohne Seitenzahl, nur weil man auf der Seite davor ein Stück zweispaltigen Text hat.

Das folgende Makro für "Seitenzahlen oben mittig" löscht ggf. die alten Seitenzahlen unten, löscht die alten Kopfzeilen und erstellt neue Seitenzahlen oben mit Halbgeviert-Strichen links und rechts (wie auf der Schreibmaschine), außer auf Seite 1. Die Kopfzeile der ersten Seite und alle Fußzeilen bleiben erhalten (ausgenommen Felder mit Seitenzahlen).

Will man die Striche davor und dahinter in Word 2010 mit der eingebauten Funktion machen, bekommt man diese Striche leider auch im Inhaltsverzeichnis. Und das kriegt die nicht so leicht wieder weg, auch wenn man alle Seitenzahlen von Hand überarbeitet. Dieses Problem wird aber **mit dem folgenden Makro behoben.**

```
Sub a8a_Seitenzahl_oben_mittig_mit_Strichen_davor_und_dahinter()
'  Dipl.-Ing. Hanna-Chris Gast, 2012, 2013, 2014, 7. Juli 2015, 14. Januar 2016.
'  Dieses Makro erzeugt Seitenzahlen oben in der Mitte mit Strichen links und rechts,
'  außer auf der ersten Seite.
'  Es löscht dabei im ganzen Dokument alle vorhandenen Kopfzeilen.
'  Das Makro löscht alle alten Seitenzahlen in den Fußzeilen, lässt aber dort alle sonstigen Texte.
'  Das Makro funktioniert auch, wenn mehrere Sections (Abschnitte) vorhanden sind.
'  Auf der ersten Seite wird die Kopfzeile wiederhergestellt (ggf. ohne Seitenzahl).
  Dim i As Integer
  Dim j As Integer
  Dim aField As Field
  Dim k As Integer
'  **************************
'  Lästiges "Flackern" des Bildschirms verhindern:
  Application.ScreenUpdating = False
'  **************************
'  Auf der ersten Seite soll keine Seitenzahl sein, diese dort löschen:
  '' ActiveDocument.Sections(1).PageSetup.DifferentFirstPageHeaderFooter = True
'  Kopfzeile der 1. Seite soll erhalten bleiben, nur ggf. Seitenzahl dort löschen:
  For Each aField In ActiveDocument.Sections(1).Headers(1).Range.Fields
    If aField.Type = 33 Then
       ' 33 = wdFieldPage
       aField.Delete
    End If
  Next aField
  ' in die Kopfzeile der Seite 1 wechseln
  Selection.HomeKey Unit:=wdStory
  ActiveWindow.View.SeekView = wdSeekCurrentPageHeader
  Selection.WholeStory
  Selection.Copy
   'wieder in das "Hauptdokument" wechseln
  ActiveWindow.ActivePane.View.SeekView = wdSeekMainDocument
'  ********
'  Alle Sections (Abschnitte) "j" durchsuchen:
  For j = 1 To ActiveDocument.Sections.Count
    ' In jedem Section gibt es ggf. drei verschieden Kopf- und Fußzeilen ("i"):
    For i = 1 To ActiveDocument.Sections(j).Headers.Count
      ' Alte Seitenzahlen unten löschen:
      For Each aField In ActiveDocument.Sections(j).Footers(i).Range.Fields
        If aField.Type = 33 Then
           ' 33 = wdFieldPage
           aField.Delete
        End If
```

```
        Next aField
        ' Seitenzahlen oben mittig einfügen:
        ActiveDocument.Sections(j).Headers(i).Range.Select
        Selection.Delete
        Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
        "  Selection.TypeText Text:="-  -"
        Selection.TypeText Text:=ChrW(8211) & "  " & ChrW(8211)
        Selection.MoveLeft Unit:=wdCharacter, Count:=2
        Selection.Fields.Add Range:=Selection.Range, Type:=wdFieldEmpty, _
        PreserveFormatting:=False
        Selection.TypeText Text:="Page"
      Next i
      ' *****
      ' Seitenzahlenformat (arabische Zahlen)
      With ActiveDocument.Sections(j).Footers(wdHeaderFooterPrimary).PageNumbers
        .NumberStyle = wdPageNumberStyleArabic
         " .NumberStyle = wdPageNumberStyleLowercaseRoman
         " .NumberStyle = wdPageNumberStyleNumberInDash  ' unbedingt vermeiden!!!
        .HeadingLevelForChapter = 0
        .IncludeChapterNumber = False
         "  .ChapterPageSeparator = wdSeparatorHyphen
        .RestartNumberingAtSection = False
        .StartingNumber = 0
      End With
    Next j
'  *******
'  Zurück zur Hauptkomponente:
    If ActiveWindow.View.SplitSpecial = wdPaneNone Then
      ActiveWindow.ActivePane.View.Type = wdPrintView
    Else
       ActiveWindow.Panes(2).Close
       ActiveWindow.View.Type = wdPrintView
    End If
    ActiveWindow.ActivePane.View.SeekView = wdSeekMainDocument
'  *******
'  Auf der ersten Seite ohne Seitenzahl:
    ActiveDocument.Sections(1).PageSetup.DifferentFirstPageHeaderFooter = True
    ' Kopfzeile auf Seite 1 löschen:
    " ActiveDocument.Sections(1).Headers(2).Range.Text = "  "
    """"""
'  Kopfzeile der 1. Seite wieder herstellen:
    Selection.HomeKey Unit:=wdStory
    ActiveWindow.View.SeekView = wdSeekCurrentPageHeader
    Selection.WholeStory
    Selection.Paste
    Selection.WholeStory
    If Right(Selection.Text, 2) = ChrW(13) & ChrW(13) Then
      Selection.Collapse Direction:=wdCollapseEnd
      Selection.TypeBackspace
    End If
    ' wieder in das "Hauptdokument" wechseln
    ActiveWindow.ActivePane.View.SeekView = wdSeekMainDocument
'  *******
'  Feldcode-Ansicht abstellen:
    "   ActiveWindow.View.ShowFieldCodes = False
    "   ActiveWindow.View.ShowFieldCodes = False
    ActiveWindow.View.ShowFieldCodes = Not ActiveWindow.View.ShowFieldCodes
    ActiveWindow.View.ShowFieldCodes = Not ActiveWindow.View.ShowFieldCodes
'  ***************************
    Application.ScreenUpdating = True
End Sub
'-----------------------------------------------------------------------------------------------------------
```

Das folgende Makro löscht ggf. die alten Seitenzahlen oben, löscht die alten Fußzeilen und erstellt neue Seitenzahlen unten außen, außer auf Seite 1. Auf Seite 1 soll die alte Fußzeile erhalten bleiben.

```
Sub a8b_Seitenzahlen_unten_außen()
'  H. Chris Gast, September 2012, April 2013, Dezember 2014, 7. + 9. + 22. Juli 2015.
'  Dieses Makro fügt in allen Sections (Abschnitten) Seitenzahlen unten außen ein,
'  außer auf der ersten Seite. Es löscht dabei die bisherigen Fußzeilen außer auf der 1. Seite.
'  Löscht ggf. alte Seitenzahlen unten, und Seitenzahlen oben ab der Seite 2.
'
'  Durch Verzicht auf die direkte Word-Funktion vermeide ich Textrahmen.
'  Für Dokumente, die auf der Std.dot usw. basieren, wird der Fußzeilenabstand
'  wie bei Normen eingestellt.
'
'  Die Fußzeile auf Seite 1 soll erhalten bleiben (außer einer Seitenzahl).
'  Alte Seitenzahlen in Textboxen werden auch gelöscht.
'  ******************
Dim j As Integer  ' Section-Nummer
Dim i As Integer  ' 1 = rechte Seite, 3 = linke Seite, 2 = erste Seite
Dim aField As Field
Dim aheader As Range   '' As HeaderFooter
'
'  Linke und rechte Seiten unterschiedlich:
  '''On Error Resume Next
  ''   ActiveDocument.PageSetup.OddAndEvenPagesHeaderFooter = True
  '''On Error GoTo 0
''''''''''
Dim Bstd As Boolean
  If InStr(LCase(ActiveDocument.BuiltInDocumentProperties(wdPropertyTemplate)), "std") > 0 Then
    Bstd = True
  Else
    Bstd = False
 End If
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' 22. Juli 2015:
' Suche nach Textfeldern in Kopf- und in Fußzeilen mit Seitenzahlen, dafür reicht eine Abfrage:
Dim rngSection As Section
Dim rngHeaderFooter As HeaderFooter
Dim shpShape As Shape       ' Textfelder und anderes Shapes in den Kopf- und Fußzeilen.
Dim BL_Textfeld As Boolean   ' Für Suche nach Textfeldern in Kopf- oder Fußzeile
 For Each rngSection In ActiveDocument.Sections
   For Each rngHeaderFooter In rngSection.Headers
      ' (sie werden sonst bei jedem rngHeaderFooter gefunden!)
      If BL_Textfeld = False Then
        For Each shpShape In rngHeaderFooter.Shapes
          If shpShape.Type = msoTextBox Then
            ''Call Querverweisprüfen_In_Range(shpShape.TextFrame.TextRange)
            'shpShape.TextFrame.TextRange.Select
            ' Alte Seitenzahlen in Textfeldern löschen:
            ''For Each aField In Selection.Range.Fields
            For Each aField In shpShape.TextFrame.TextRange.Fields
               aField.Select
               ''MsgBox aField.Result
               ''aField.Select
               If aField.Type = 33 Then
                 ' 33 = wdFieldPage
                 aField.Delete
               End If
            Next aField
```

```
            BL_Textfeld = True
          End If
       Next shpShape
     End If
   Next rngHeaderFooter
Next rngSection
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

'  Auf der ersten Seite soll keine Seitenzahl sein, diese dort löschen:
  '' ActiveDocument.Sections(1).PageSetup.DifferentFirstPageHeaderFooter = True
'  Fußzeile der 1. Seite soll erhalten bleiben, nur ggf. Seitenzahl dort löschen:
  For Each aField In ActiveDocument.Sections(1).Footers(1).Range.Fields
    If aField.Type = 33 Then
       ' 33 = wdFieldPage
       aField.Delete
    End If
  Next aField
  ' in die Fußzeile der Seite 1 wechseln
  If ActiveWindow.View.SplitSpecial = wdPaneNone Then
    ActiveWindow.ActivePane.View.Type = wdPrintView
  Else
     ActiveWindow.Panes(2).Close
     ActiveWindow.View.Type = wdPrintView
  End If
  On Error Resume Next:
  ActiveWindow.ActivePane.View.SeekView = wdSeekMainDocument
  On Error GoTo 0
  Selection.HomeKey Unit:=wdStory
  ActiveWindow.View.SeekView = wdSeekCurrentPageFooter
  Selection.WholeStory
  Selection.Copy
   'wieder in das "Hauptdokument" wechseln
  ActiveWindow.ActivePane.View.SeekView = wdSeekMainDocument
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

'  Alle Sections (Abschnitte) "j" durchsuchen:
''On Error Resume Next
  For j = 1 To ActiveDocument.Sections.Count
    ' Gerade und ungerade Seiten (Linke und rechte Seiten) unterschiedlich:
    ActiveDocument.Sections(j).PageSetup.OddAndEvenPagesHeaderFooter = True
    If j > 1 Then
      ''On Error Resume Next
      ActiveDocument.Sections(j).PageSetup.DifferentFirstPageHeaderFooter = False
      ''On Error GoTo 0
    End If
    ' In jedem Section gibt es ggf. drei verschieden Kopf- und Fußzeilen ("i"):
    For i = 1 To ActiveDocument.Sections(j).Footers.Count
      If i = 2 Then   ' 2 ???
        ' auslassen (jeweils erste Seite eines Sections)
      Else
        ' Alte Seitenzahlen oben löschen:
        For Each aField In ActiveDocument.Sections(j).Headers(i).Range.Fields
          aField.Select
          ''MsgBox "!"
          If aField.Type = 33 Then
             ' 33 = wdFieldPage
             aField.Delete
          End If
        Next aField
        ' Ggf. auch die alten Striche von Seitennummerierung in Kopfzeilen beseitigen:
        Set aheader = ActiveDocument.Sections(j).Headers(i).Range
          ''  aHeader.Find.ClearFormatting
          ''  aHeader.Range.Find.Replacement.ClearFormatting
        With aheader.Find
```

```
        .Text = ChrW(8211) & "  " & ChrW(8211)
        .Replacement.Text = ""
        .Forward = True
        .Wrap = wdFindContinue
        .Format = True
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = False
        .MatchSoundsLike = False
        .MatchAllWordForms = False
      End With
      aheader.Find.Execute Replace:=wdReplaceAll
      ' bzw. mit einfachen Strichen:
      With aheader.Find
          .Text = "-  -"
          .Replacement.Text = ""
          .Forward = True
          .Wrap = wdFindContinue
          .Format = True
          .MatchCase = False
          .MatchWholeWord = False
          .MatchWildcards = False
          .MatchSoundsLike = False
          .MatchAllWordForms = False
      End With
      aheader.Find.Execute Replace:=wdReplaceAll
      ' *************************
      ' Seitenzahlen unten außen einfügen:
      ActiveDocument.Sections(j).Footers(i).Range.Select
      ' vorhandenen Text löschen:
      Selection.Delete
      If i < 3 Then
        Selection.ParagraphFormat.Alignment = wdAlignParagraphRight
      Else
        Selection.ParagraphFormat.Alignment = wdAlignParagraphLeft
      End If
      '''''''''''''''''''''
      ''ActiveDocument.Sections(j).Footers(1).PageNumbers.Add _
      ''PageNumberAlignment:=wdAlignPageNumberOutside, FirstPage:=True
      '''  Das ergibt Textboxen!
      '''''''''''''''''''''
      Selection.Fields.Add Range:=Selection.Range, Type:=wdFieldEmpty, _
      PreserveFormatting:=False
      Selection.TypeText Text:="Page"
      '''''''''''''''''''''
      ' Für Dateien,die auf der Std.dot basieren:
      If Bstd = True Then
        With Selection.ParagraphFormat
          .SpaceBefore = 14.15
          .SpaceAfter = 14.15
        End With
      End If
      '''''''''''''''''''''
    End If
  Next i
  ' ***************
  ' Seitenzahlenformat (arabische Zahlen)
  With ActiveDocument.Sections(j).Footers(wdHeaderFooterPrimary).PageNumbers
    .NumberStyle = wdPageNumberStyleArabic
     '' .NumberStyle = wdPageNumberStyleLowercaseRoman
     '' .NumberStyle = wdPageNumberStyleNumberInDash  ' unbedingt vermeiden!!!
    .HeadingLevelForChapter = 0
```

```
        .IncludeChapterNumber = False
         '' .ChapterPageSeparator = wdSeparatorHyphen
        .RestartNumberingAtSection = False
        .StartingNumber = 0
      End With
   Next j
' *******
' Zurück zur Hauptkomponente:
   If ActiveWindow.View.SplitSpecial = wdPaneNone Then
      ActiveWindow.ActivePane.View.Type = wdPrintView
   Else
       ActiveWindow.Panes(2).Close
       ActiveWindow.View.Type = wdPrintView
   End If
   On Error Resume Next
   ActiveWindow.ActivePane.View.SeekView = wdSeekMainDocument
   On Error GoTo 0
''''''''*********************************************************
' Fußzeile auf Seite 1 wiederherstellen:
   Selection.HomeKey Unit:=wdStory
   ActiveDocument.Sections(1).PageSetup.DifferentFirstPageHeaderFooter = True
   ActiveWindow.View.SeekView = wdSeekCurrentPageFooter
   Selection.WholeStory
   Selection.Paste
   Selection.WholeStory
   If Right(Selection.Text, 2) = ChrW(13) & ChrW(13) Then
      Selection.Collapse Direction:=wdCollapseEnd
      Selection.TypeBackspace
   End If
   ' wieder in das "Hauptdokument" wechseln
   ActiveWindow.ActivePane.View.SeekView = wdSeekMainDocument
' ******''''''''''*******************************************
' Feldcode-Ansicht abstellen:
   ''   ActiveWindow.View.ShowFieldCodes = False
   ''   ActiveWindow.View.ShowFieldCodes = False
   ActiveWindow.View.ShowFieldCodes = Not ActiveWindow.View.ShowFieldCodes
   ActiveWindow.View.ShowFieldCodes = Not ActiveWindow.View.ShowFieldCodes
End Sub
'---------------------------------------------------------------------------------------------------------
```

11 Inhalt einfügen

Nach Anleitung aus einer Computer-Zeitschrift.
Gewähltes Tastenkürzel: Strg + i wie in Word 97.

```
Sub Inhalte_einfügen()
   Selection.PasteSpecial Link:=False, DataType:=wdPasteText, _
          Placement:=wdInLine, DisplayAsIcon:=False
End Sub
```

Wendet man das Makro an, ohne vorher nach dem Einschalten des Computers irgendetwas in die Zwischenablage kopiert zu haben, gibt es die Fehlermeldung 4605. Kopiert man vorher unsichtbaren Text ("ausgeblendet") in die Zwischenablage, gibt es den Fehler 4198. Hat man ein Bild in der Zwischenablage, gibt es den Fehler 5342. Dies könnte man, wenn man unbedingt will, durch eine Fehlerabfrage wie folgt umgehen (ich selbst brauche es nicht):

```
Sub Inhalt_einfügen()
'  Makro erweitert um Fehlerbehandlung am 7.Okt. 2012
  On Error GoTo Fehlermeldung
  Selection.PasteSpecial DataType:=wdPasteText
Fehlermeldung:
  If Err <> 0 Then
     MsgBox ("Die Zwischenablage enthielt keinen brauchbaren Text.")
  End If
End Sub
```

****

Beim Befehl für "Inhalt-Einfügen" wird der Feste Bindestrich wie bei dem Wort "E-Modul" fälschlich durch ein Leerzeichen ersetzt, was **nicht akzeptable ist**.[6]

Ferner wird bei der Funktion "Inhalt-Einfügen" in Word 2010 der Silbentrennstrich (englisch: "Soft Hyphen") durch falsche Zeichen ersetzt, teils durch einen normalen Bindestrich und teils durch das "Nicht-Zeichen" (englisch "not sign", Code 0172), das auf dem Bildschirm zwar genauso aussieht wie der bedingte Silbentrennstrich, aber im Druckbild leider sichtbar bleibt (¬). Dies passiert auch, wenn die Anzeige der Steuerzeichen abgeschaltet ist.[7]

Dies wird mit dem folgenden Makro behoben.

Nicht behobene Nachteile:
Nummerierungen starten neu mit "a)" bzw. "1)".
Es kann nicht gleichzeitig Striche korrigiert werden und der Quelltext von OMML-Formeln in Word 2010 ausgegeben werden.

Eine erweiterte Version, bei der ein Makro für den Ersatz von Symbol- und Wingdingszeichen aufgerufen wird, und bei der Indizes und Exponenten erhalten bleiben, siehe [5].

```
Sub Inhalt_einfügen_korrigiert()
'  Hanna-Chris Gast, 2010, 2014, Mai bis 25. August 2015, 14. + 23. September 2015.
'  Inhalt einfügen, aber Feste Bindestriche als Bindestriche retten
```

[6] Word verwendet für den **Festen Bindestrich** (NON-BREAKING HYPHEN) das Steuerzeichen mit dem dezimalen Code "30". Der Code des eigentlichen Unicode-Zeichens für ist hexadezimal: 2011 und dezimal: 8209. Dieses ist aber nicht mit den Standard-Schriftarten von Word darstellbar.

[7] Word verwendet für den "**Bedingten Bindestrich**" (SOFT HYPHEN) das Steuerzeichen mit dem dezimalen Code "31". Der Code des eigentlichen Unicode-Zeichens hat den dezimalen Wert 173, dieser funktioniert aber in Word nicht.

```
'  und Bedingte Trennstriche löschen.
'  Funktioniert von Word 2000 bis Word 2010, Windows XP und Windows 7.
'  Zugeordnetes Tastenkürzel: Strg + i.
'
'  Neue Version mit ggf. unsichtbarer Zwischendatei, in der die Striche bearbeitet werden.
'  Kein Flackern des Bildschirms mehr!
'
'  Problem und Lösung:
'  In Word 2000 geht der Bedingte Silbentrennstrich beim "Inhalt einfügen" verloren,
'  in Word 2010 wird (Stand Mai 2015) der Silbentrennstrich durch falsche Zeichen
'  ersetzt, teils durch normalen Bindestrich und teils durch das "Nicht-Zeichen (NOT SIGN)",
'  Code 0172, das zwar auf dem Bildschirm genauso aussieht, aber im Druckbild völlig
'  unakzeptabel ist. Das Makro löscht deshalb den Bedingten Silbentrennstrich ersatzlos.
'
'  Die Zwischendatei ist unsichtbar, damit das Makro schneller läuft.
'  Bei auftretenden Fehlern wird diese durch die Fehlerbehandlung geschlossen,
'  da unsichtbare offene Dateien sonst lästig sind.
'
'  Zeichen einiger Sonderschriftarten, die nicht Unicode-kompatibel sind, werden dunkelrosa gefärbt.
'
'  Offenes Manko:
'  Wenn Striche korrigiert werden, dann beginnen Nummerierungen in der Kopie mit a) bzw. 1) usw.
'  statt mit der ursprünglichen Nummer.
'
'  Kompromisslösung für OMML-Formeln:
'  Um den Quelltext von OMML-Formeln anzeigen zu können, wird bei Vorhandensein solcher Formel
'  auf die Korrektur von Strichen und auf den Hinweis auf Sonderzeichen verzichtet.
'  ********************
Dim Dok1 As Word.Document
Dim Dok2 As Word.Document
Dim BlUmbruch As Boolean: BlUmbruch = False
Dim B_Strich As Boolean: B_Strich = False
Dim aField As Field
Dim k As Integer: k = 0
'  ********************
  WordBasic.DisableAutoMacros 1   ' Disables auto macros
'  ********************
'  Lästiges "Flackern" des Bildschirms verhindern:
  StatusBar = "Bitte warten - das Makro läuft."
  Application.ScreenUpdating = False
  Options.Pagination = False
'  ******************************************************************************
'  Zwischen-Dokument erstellen:
  Set Dok1 = ActiveDocument
  Set Dok2 = Documents.Add(DocumentType:=wdNewBlankDocument, Visible:=False)
  Dok2.Activate
  ' Falls in der Normal-dot ein Briefkopf vorhanden ist, diesen löschen:
  Selection.WholeStory
  Selection.Delete
'  ****************
'  In neuem Zwischendokument Text aus Zwischenablage einfügen,
'  dabei prüfen, ob die Zwischenablage nicht leer ist und auch Text enthält:
On Error GoTo Zeile_Zwischenablage_leer
  Selection.Paste
On Error GoTo Zeile_Fehler
'  ****************
'  Prüfen, ob etwas eingefügt wurde:
  Selection.WholeStory
  Selection.MoveEnd Unit:=wdCharacter, Count:=-1
  If Len(Selection) = 1 And 10 <= Asc(Selection) And Asc(Selection) <= 14 Then
    BlUmbruch = True
  End If
```

```
' ************************************
' Enhält der Text eine ehemalige OMML-Formel?
  If Val(Application.Version) >= 14 _
  And Selection.Fields.Count > 0 Then
    For Each aField In Selection.Fields
      If aField.Code Like "*QUOTE " & ChrW(1) & "*" Then
        ' Feldtyp ist 35 und enthält Grafik (18.9.2015).
        "MsgBox "Rückkonvertiertes Word-2010-Bild!"
        k = k + 1
      End If
    Next aField
    "MsgBox "Es gibt in der Markierung " & k & " in Grafik konvertierte Word-2010-Bild(er)."
  End If
' ************************************
' Feste Bindestriche durch normales Bindestrich-Minus ersetzen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
      .Text = "^~"
      .Replacement.Text = "-"
      .Forward = True
      .Wrap = wdFindContinue
      .Format = False
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
  If Selection.Find.Found = True Then B_Strich = True
' ************************************************
' Bedingte Silben-Trennstriche (Steuerzeichen) alle beseitigen
' (Word 2010 ersetzt sie sonst falsch durch das "Nicht-Zeichen 0172):
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = ChrW(31)
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
  If Selection.Find.Found = True Then B_Strich = True
' ************************************************
' Sonderschrift "BSsymb9" codieren:
   Selection.Find.ClearFormatting
   Selection.Find.Replacement.ClearFormatting
   With Selection.Find
      .Font.Name = "BSsymb9"
      .Text = ""
      .Replacement.Font.Name = "Arial"
      .Replacement.Text = "<!BSsymb9-Zeichen!>"
      .Forward = True
      .Wrap = wdFindContinue
      .Format = True
```

```
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = False
        .MatchSoundsLike = False
        .MatchAllWordForms = False
    End With
    Selection.Find.Execute Replace:=wdReplaceAll
    If Selection.Find.Found = True Then B_Strich = True
'  ******************************************************************
    If B_Strich = False _
    Or k > 0 Then
      ' Zwischendatei schließen:
      Dok2.Close (wdDoNotSaveChanges)
    '  Im ursprünglichen Ziel-Dokument bearbeiteten Text einfügen:
      Dok1.Activate
      Selection.PasteSpecial DataType:=wdPasteText
      GoTo Zeile_Schließen
    End If
'  ******************************************************************
'  Alles in der Zwischendatei markieren und kopieren,
'  dann als "Inhalt" in derselben Zwischensdatei einfügen,
'  sofern brauchbarer Text vorhanden ist:
    Selection.WholeStory
    Selection.MoveEnd Unit:=wdCharacter, Count:=-1
    ''''''''
    If Selection.Type > 6 Then
      Application.Visible = True
      WordBasic.DisableAutoMacros 0
      Dok2.Close (wdDoNotSaveChanges)
      Dok1.Activate
      MsgBox "Es ist nur ein Bild markiert, kein einfügbarer Text!"
      Set Dok1 = Nothing
      Set Dok2 = Nothing
      Exit Sub
    End If
    ''''''''
    If Selection.Type = 1 Then
      ' Es ist nichts Sichtbares mehr vorhanden.
      Application.Visible = True
      WordBasic.DisableAutoMacros 0
      Dok2.Close (wdDoNotSaveChanges)
      Dok1.Activate
      MsgBox "Es ist nichts Sichtbares zum Einfügen vorhanden."
      Set Dok1 = Nothing
      Set Dok2 = Nothing
      Exit Sub
    ElseIf BlUmbruch = False Then
      ' Markierung ausschneiden:
      Selection.Cut
      '  "Inhalt" einfügen in der Zwischendatei:
      Selection.PasteSpecial Link:=False, DataType:=wdPasteText, _
        Placement:=wdInLine, DisplayAsIcon:=False
    Else
      Selection.TypeParagraph
    End If
'  ******************************************************************
'  Nicht-ersetzte Zeichen der Windows-Sonderschriftarten
'  in der "Private Use Area", (PUA), dunkelrosa färben:
    Options.DefaultHighlightColorIndex = wdPink
    ActiveDocument.Range.Select
    Selection.Find.ClearFormatting
    Selection.Find.Replacement.ClearFormatting
```

```
  With Selection.Find
    .Replacement.Highlight = True
    .Replacement.Font.ColorIndex = wdAuto
    .Text = "[" & ChrW(57344) & "-" & ChrW(63743) & "]"
    ' Arial und Times New Roman enthalten in der Private Use Area
    ' einige Sonderzeichen, die hier sonst irreführen würden:
    .Replacement.Font.Name = "Courier New"
    .Replacement.Text = ""
    .Forward = True
    ".Wrap = wdFindContinue  ' darf hier nicht sein!!!
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = True
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
  Options.DefaultHighlightColorIndex = wdYellow
' ************************************************
' BSsymb9-Warnung mit dunkelrosa Markierung versehen(1.8.2015):
  Options.DefaultHighlightColorIndex = wdPink
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "<!BSsymb9-Zeichen!>"
    .Replacement.Highlight = True
    .Replacement.Text = ""
    ".Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
  Options.DefaultHighlightColorIndex = wdYellow
' ************************************************
' Alles in der Zwischendatei markieren und in die Zieldatei kopieren
' (außer Absatzmarke am Schluss):
  Selection.WholeStory
  Selection.MoveEnd Unit:=wdCharacter, Count:=-1
  Selection.Copy
' Zwischendatei schließen:
  Dok2.Close (wdDoNotSaveChanges)
' Im ursprünglichen Ziel-Dokument bearbeiteten Text einfügen:
  Dok1.Activate
  Selection.Paste
' ************************************************
Zeile_Schließen:
' ************************************************
' Automakros wieder zulassen:
  WordBasic.DisableAutoMacros 0
' Im Suchen/Ersetzen-Menü alles zurückstellen
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
      .Text = ""
      .Replacement.Text = ""
      .Forward = True
      .Wrap = wdFindContinue
```

```
        .Format = False
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = False
        .MatchSoundsLike = False
        .MatchAllWordForms = False
    End With
'  Bildschirm wieder einschalten:
    Options.Pagination = True
    StatusBar = False
    Application.ScreenUpdating = True
    Application.DisplayStatusBar = True
'  ***********************************************
'  Angabe, wenn auf nötige Korrekturen verzichtet wurde
'  (Warteschleife ist hier nötig):
    Dim tt As Single
    tt = Timer
     Do While Timer < tt + 0.1  'Sekunden
        DoEvents
    Loop
    ' ***
    If B_Strich = True _
    And k > 0 Then
      MsgBox "Es wurde auf eine Korrektur von Strichen bzw. Sonderschriftarten verzichtet, " _
          & "um den Quelltext der OMML-Formel(n) richtig anzuzeigen."
    End If
' Makro ist hier zuende:
Exit Sub
' ********************************************************************
' Fehlerroutinen:
Zeile_Zwischenablage_leer:
    Application.Visible = True
    WordBasic.DisableAutoMacros 0
    Dok2.Close (wdDoNotSaveChanges)
    Dok1.Activate
    MsgBox "Die Zwischenablage (Clipboard) ist leer!", vbExclamation
    '   Set Dok1 = Nothing
    '   Set Dok2 = Nothing
Exit Sub
'  ************************************
Zeile_Fehler:
    Application.Visible = True
    WordBasic.DisableAutoMacros 0
    Dok2.Close (wdDoNotSaveChanges)
    Dok1.Activate
    MsgBox "Fehler in Sub Fehlerbehandlung" & vbCrLf & "Fehlernummer: " & Err.Number & _
        vbCrLf & "Fehlerbeschreibung: " & Err.Description, vbCritical
    '  Für die Fehlersuche hier testweise die Fehlerstelle wieder aufsuchen:
    '   Resume
End Sub
' ---------------------------------------------------------------------------------------------------------
```

12 Absätze verbinden

Das folgende Makro verbindet die markierten Absätze. In einer Tabelle verhindert es den Umbruch der betreffenden Tabellenzeile. Ich habe mir als Tastenkürzel "AltGr + V" festgelegt.

```
Sub Absätze_verbinden()
' , Stand 21. Dezember 2011.
' Zeilen und Absätze verbinden, auch Tabellenzeilen.
'
  With Selection.ParagraphFormat
      .WidowControl = True
      .KeepWithNext = True
      .KeepTogether = True
  End With
'
'  Abfrage, ob Markierung in einer Tabelle. Dann betreffende Tabellenzeile(n) nicht umbrechen.
  If Selection.Information(wdWithInTable) = True Then
    Selection.SelectRow
    Selection.Rows.AllowBreakAcrossPages = False   ' nur die markierte(n) Tabellenzeile
    ' in der ganze Tabelle Umbruch in Tabellenzeilen verhindern:
    "    Selection.Tables(1).Rows.AllowBreakAcrossPages = False
    "    ' Erzwungener Seitenumbruch gibt öfter Ärger als Nutzen:
    "    Selection.ParagraphFormat.PageBreakBefore = False
  End If
End Sub
```

13 lästige Rahmen entfernen

```
Sub Rahmen_entfernen()
'  Dieses Makro beseitigt Rahmen um Zeichen und Absätze.
  With Selection.Font
    With .Borders(1)
        .LineStyle = wdLineStyleNone 'wdLineStyleSingle
        ' .LineWidth = wdUndefined
        .Color = wdColorAutomatic
    End With
    .Borders.Shadow = False
  End With
  With Options
    .DefaultBorderLineStyle = wdLineStyleNone
  '  .DefaultBorderLineWidth = wdLineWidth050pt
    .DefaultBorderColor = wdColorAutomatic
  End With
'  **********
'  Der Befehl "Strg + Q" =  "Selection.ParagraphFormat.Reset"
'  würde hier sonst alle übrigen Absatzformatierungen beseitigen, deshalb hier aufwändiger:
On Error Resume Next
  Selection.Borders(wdBorderTop).LineStyle = wdLineStyleNone
  Selection.Borders(wdBorderLeft).LineStyle = wdLineStyleNone
  Selection.Borders(wdBorderBottom).LineStyle = wdLineStyleNone
  Selection.Borders(wdBorderRight).LineStyle = wdLineStyleNone
  Selection.Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone
  Selection.Borders(wdBorderDiagonalDown).LineStyle = wdLineStyleNone
  Selection.Borders(wdBorderDiagonalUp).LineStyle = wdLineStyleNone
'  ***********'
'  Tabellen beseitigen:
Selection.Rows.ConvertToText Separator:=wdSeparateByParagraphs, _
    NestedTables:=True
End Sub
```

14 Tausender-Gliederung von langen Zahlen

Bei großen Zahlen (außer Jahreszahlen, Telefonnummern und ähnlichem) wird zur besseren Lesbarkeit alle drei Stellen ("Tausender") ein Leerzeichen eingefügt, das zur Vermeidung von Zeilenumbrüchen oder von zu großen Abständen ein **Geschütztes Leerzeichen** sein muss.

— In Deutschland gilt dafür die Norm **DIN 5008**,
"*Schreib- und Gestaltungsregeln für die Textverarbeitung*"

— In Österreich die Norm **A 1080**, "*Richtlinien für die Textgestaltung*"

— In der Schweiz gilt die Norm **SN 010130**, "*Geschäftsbrief*"

"Schmale geschützte Leerzeichen" lassen die Zahlen besser als Einheit erkennen, und es sieht besser aus, siehe folgende Tabelle.

Hex. Code	Dezimaler Code	Deutsche Beschreibung	Englischer Name	Anwendungs-Beispiel
U+00A0	0160	Festes Leerzeichen	NO-BREAK SPACE	**10 000 000,000 000**
U+202F	8239	Schmales geschütztes Leerzeichen "Achtelgeviert"	NARROW NO-BREAK SPACE	**10 000 000,000 000**

14.1 Zeichenabstand an der Cursorstelle erweitern

Für Computer ist es unter Umständen besser, statt des (festen) Leerzeichens einen **erweiterten Zeichen-Abstand** zu wählen, vor allem, wenn beim Empfänger einer Datei passende Schriftarten für das Schmale geschützte Leerzeichen fehlen.

Vorgehensweise ohne Makro:
Jeweils die letzte Ziffer von drei Ziffern markieren, dann bei Word oben auf dem Bildschirm anklicken:
a) Bis Word 2003:
"Format→Z eichen→Z eichenabstand→L aufweite-erweitert→ 2 Punkt"
b) In Word 2010:
"Start→S chriftart (Pfeil nach unten rechts)→R eiter:" Erweitert"→Abstand"Erweitert"→ "2Pt"
Das ergibt: # ### ###, ###

Braucht man dieses häufiger, ist die Verwendung eines Makros bequemer. Mit dem folgenden Makro erhält man passend eine **Lücke für die Tausender-Trennung** in einer Zahl.

Ich habe mir für das folgende Makro den Tastaturbefehl (Tastenkürzel)
AltGr + Leerzeichen.
festgelegt. Das Makro hatte ich ursprünglich mit dem Makro-Rekorder aufgezeichnet (siehe 1.1).

```
Sub erweiterter_Zeichenabstand()
    ' Makro aufgezeichnet am 2006-07-28 von Hanna-Chris GAST
    ' empfohlenes Tastenkürzel: AltGr + Leertaste
    ' Das Makro erzeugt einen erweiterten Zeichenabstand an der Cursorstelle.
      Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
      With Selection.Font
        .Spacing = 2
        .Scaling = 100
        .Position = 0
        .Kerning = 0
        .Animation = wdAnimationNone
      Selection.MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdExtend
      End With
    End Sub
```

Das "halbe Leerzeichen" entsteht dann an der Stelle des Cursors.

14.2 Tausender-Gliederung von langen Zahlen im gesamten Dokument

Das folgende Makro fügt bei allen großen Zahlen Schmale Feste Leerzeichen an den Tausender-Stellen ein. Hierbei halte ich mich an die **Schweizer Norm**, die bei vierstelligen Zahlen diese Gliederung weitgehend weglässt. So braucht mein Makro nicht zu unterscheiden zwischen Jahreszahlen und Maßangaben. Es wird das ganze Dokument nach Zahlen durchsucht, ausgenommen Fußnoten, Kopf- und Fußzeilen sowie Textfeldern [8]

Die abweichenden Regeln für Postleitzahlen und Telefonnummern werden hier leider nicht berücksichtigt. Die vom Makro erzeugten Farbmarkierungen (farbige Hervorhebungen) dienen hierbei der nachträglichen Fehlersuche. So können ggf. Postleitzahlen und Telefonnummern oder Norm-Nummern leichter von Hand zurückkorrigiert werden.

Anmerkung: Eine Makro-Version, welche die abweichenden Regeln bei Norm-Nummern berücksichtigt, kann ich auf Anfrage anbieten. Den Verzicht auf Lücken bei Telefonnummern mit mehr als vier Ziffern empfinde ich als Zumutung! Von daher bin ich nicht bereit, bei diesen eine Ausnahme zu programmieren.

```
Sub Tausender_Lücken_einfügen_Schmale_Feste_Leerzeichen()
' H. Chris Gast, 1. bis  14. März 2012, 11. April 2014, korrigiert 2. Mai 2014.
' Zahlen unterteilen alle drei Stellen durch Schmale Feste Leerzeichen, wenn mehr als 4 Stellen.
' Also nicht bei vierstelligen Zahlen.
' Die Farbmarkierungen dienten der Fehlersuche bei der Entwicklung des Makros.
' ******************************
' Lücke einfügen, 3 Stellen rechts vom Komma:
Zeile_1:
  Selection.Find.ClearFormatting
  With Selection.Find
    .Text = ",^#^#^#^#"
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindContinue
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute
  If Selection.Find.Found Then
    Selection.SetRange Start:=Selection.Start + 4, End:=Selection.Start + 4
    Selection.TypeText Text:=ChrW(8239)
    Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
    Selection.Range.HighlightColorIndex = wdYellow
    Selection.MoveRight Unit:=wdCharacter, Count:=1
    GoTo Zeile_1
  End If
'  ************************
'  weiter rechts vom Komma:
Zeile_2:
```

[8] Der Sonderfall von Punkten zur Tausendertrennung bei Geldbeträgen wird vom Makro nicht berücksichtigt.

```
  Selection.Find.ClearFormatting
  With Selection.Find
    .Text = "^#" & ChrW(8239) & "^#^#^#^#"
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindContinue
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute
  If Selection.Find.Found Then
     Selection.SetRange Start:=Selection.Start + 5, End:=Selection.Start + 5
     Selection.TypeText Text:=ChrW(8239)
     Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
     Selection.Range.HighlightColorIndex = wdYellow
     Selection.MoveRight Unit:=wdCharacter, Count:=1
     GoTo Zeile_2
  End If
' ******************************
' Links vom Komma bzw. bei Zahlen ohne Komma:
Zeile_3:
  Selection.Find.ClearFormatting
  With Selection.Find
    .Text = "[0-9][0-9][0-9][0-9][0-9][!0-9]"
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindContinue
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
  End With
  Selection.Find.Execute
  If Selection.Find.Found Then
     Selection.SetRange Start:=Selection.Start + 2, End:=Selection.Start + 2
     Selection.TypeText Text:=ChrW(8239)
     Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
     Selection.Range.HighlightColorIndex = wdYellow
     Selection.MoveRight Unit:=wdCharacter, Count:=1
     GoTo Zeile_3
  End If
' ************************************
'  Sonderfall: Tabellenzellenmarke:
Zeile4:
  Selection.Find.ClearFormatting
  With Selection.Find
    .Highlight = False
    .Text = "^#^#^#^#^#" & ChrW(7)
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindContinue
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
```

```
    .MatchWildcards = False
  End With
  Selection.Find.Execute
  If Selection.Find.Found Then
     '' MsgBox ""
     Selection.SetRange Start:=Selection.Start + 2, End:=Selection.Start + 2
     Selection.TypeText Text:=ChrW(8239)
     Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
     Selection.Range.HighlightColorIndex = wdPink ' wdYellow
     Selection.MoveRight Unit:=wdCharacter, Count:=1
     GoTo Zeile_4
  End If
' ***********************************
'  Weiter links vom Komma:
Zeile_5:
  Selection.Find.ClearFormatting
  With Selection.Find
    .Text = "^#^#^#^#" & ChrW(8239) & "^#"
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute
  If Selection.Find.Found Then
     Selection.SetRange Start:=Selection.Start + 1, End:=Selection.Start + 1
     Selection.TypeText Text:=ChrW(8239)
     Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
     Selection.Range.HighlightColorIndex = wdYellow
     Selection.MoveRight Unit:=wdCharacter, Count:=1
     GoTo Zeile_5
  End If
'  ***********************************
'  (hier eventuell das Schmale Leerzeichen durch "Erweiterten Abstand" ersetzen).
End Sub
```

Das "Schmale Feste Leerzeichen" ChrW(8239) ist leider nicht in allen Schriftarten vorhanden. Man kann man stattdessen auch

— nach Schweizer Art das Apostroph (Hochkomma) wählen: **ChrW(39)**.
— oder das normale **Feste Leerzeichen ChrW(160)**.

Man kann auch die Schmalen Festen Leerzeichen wie ich durch einen **erweiterten Abstand** ersetzen, dann bleibt der Quelltext unversehrt:

Vor dem "**End Sub**" ist dann einzufügen:

```
'  Schmales Festes Leerzeichen überall durch Erweiterten Abstand ersetzen:
  Selection.HomeKey Unit:=wdStory
Zeile_Schleifenbeginn:
'  ****************************************
  Selection.Find.ClearFormatting
  '  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = ChrW(8239)
    .Replacement.Text = ""
```

```
    .Forward = True
    .Wrap = wdFindStop
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute ' Replace:=wdReplaceAll
  '
  If Selection.Find.Found Then
    ' Erweiterten Abstand erzeugen:
    Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
    Selection.Delete Unit:=wdCharacter, Count:=1
   '' Call Halbschritt
        Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
      With Selection.Font
        .Spacing = 2
      End With
      Selection.MoveRight Unit:=wdCharacter, Count:=1
      Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
      Selection.Range.HighlightColorIndex = wdDarkYellow
    Selection.Collapse Direction:=wdCollapseEnd
    GoTo Zeile_Schleifenbeginn
  End If
```

****

15 Makros für Farben

Für mich selbst erwies sich das Unterscheiden von grünen, roten oder blauen Textfarben im Schwarzweißdruck eigentlich nie als Problem, da ich grüne Kommentare stets *kursiv* setze, blau bei mir fast nur in Hyperlinks mit Unterstreichung vorkommt, und ich sowieso lieber mit Farbmarkierungen arbeite und rot in fertigen Texten nie verwende. Problematisch waren für mich aber die farbigen Schattierungen von Tabellenzellen, weil die Druckerei sie in meinem Buch über die griechische Schrift für den Computer in andere Grauwerte umsetzte als mein eigener Drucker. Ich musste deshalb nach dem gescheiterten ersten Probedruck ein neues Druckmanuskript mit unterschiedlichen grauen Tabellenzellen-Schattierungen einreichen.

Hinweis: Für mich persönlich ist die Word-Farbe "Türkis" (wdTurquoise) eher ein Hellblau oder Himmelblau! Als "Türkis" empfinde ich eher bläuliche Grüntöne: Muster für Türkis. Und so ein echtes Grün.

15.1 Farbkreis und Farbtabellen

Das folgende Bild (Farbkreis aus Wikipedia [9], Kreis mit Zahlenwerten von mir ergänzt [10]) zeigt den Farbkreis mit den Winkeln nach dem "HLS-System". Die Winkel auf dem Farbkreis entstehen durch Transformation der RGB-Werte in Polarkoordinaten.

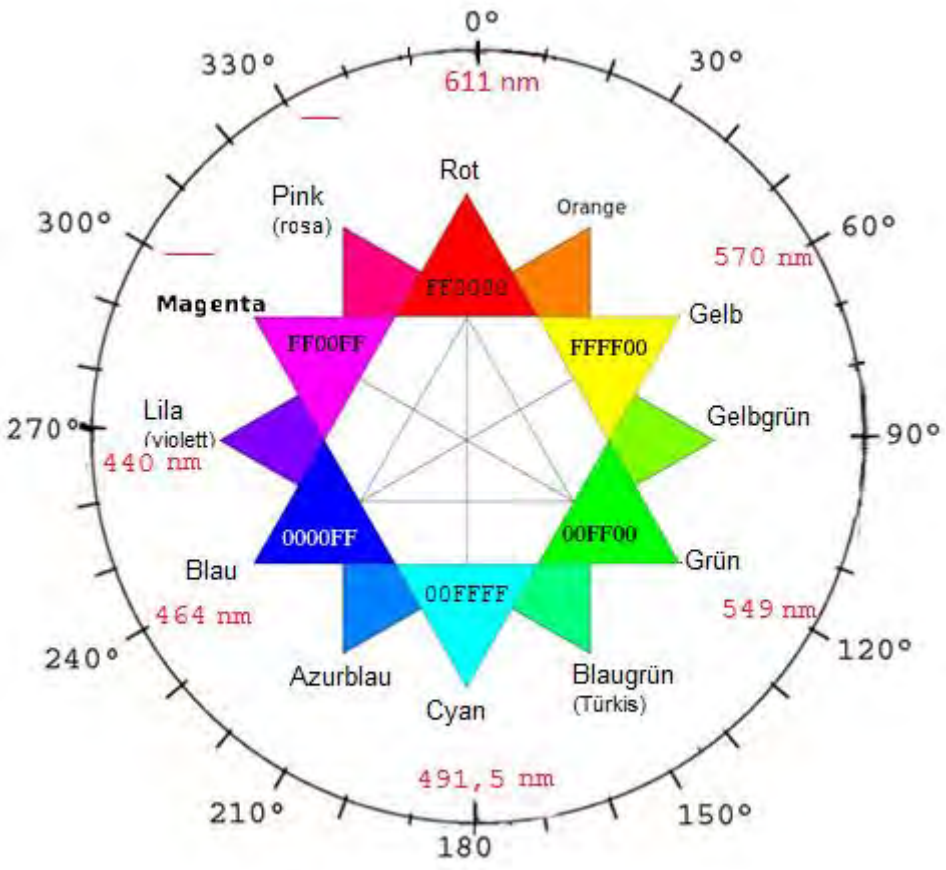


Bild 15.1: Farbkreis mit Winkelangabe

[9] "RGB color wheel", in https://en.wikipedia.org/wiki/Color_wheel, (Download März 2017).

[10] Wellenlängen mit sRGB nach J. Behrends, http://www.magnetkern.de/spektrum.html (März 2017), dagegen "Lighting Analysts", USA, http://docs.agi32.com/AGi32/Content/defining_luminaires/SRE_Concepts.htm (5.April 2017), mit ähnlichen Zahlenwerten bei rot, grün, cyan und blau, aber abweichend bei Gelb und Violett. Siehe auch Wikipedia https://en.wikipedia.org/wiki/Spectral_color (5. April 2017).

Die Anteile von RGB-Wert ("Rot, Grün, Blau" bzw. englisch "Red, Green, Blue") werden meist als Zahlenwerte von Null bis 255 angegeben (8 Bit). Bei den

Die folgenden Tabellen zeigen die benannten Farben in Word.
Die erste Tabelle zeigt die Index-Farben der "Selection.xxx.ColorIndex", die zweite Tabelle die RGB-Farben für "Selection.xxx.Color". Die deutsche Bezeichnung fügte ich weitgehend nach Fundstellen im Internet hinzu, habe sie aber gemäß meinem eigenen Farbempfinden zum Teil korrigiert.

Tabelle 15.1: Die 16 "ColorIndex"-Farben für Schriftfarben und für Farb-Markierungen

ColorIndex	Als Textfarbe	Als Hintergrund	Rot	Grün	Blau	Farbcode "Color"	Farbkreis-Winkel (Hue)
-1	wdByAuthor [a]						
0	wdAuto = schwarzer Text	wdAuto = weißer Hintergrund				-16777216	–
1	wdBlack	Schwarz	0	0	0	0	–
15	wdGray50	Grau-50	128	128	128	8 421 504	–
16	wdGray25	Grau-25	192	192	192	12 632 256	–
8	wdWhite	Weiß	255	255	255	16 777 215 *- 603 914 241*	–
13	wdDarkRed	Dunkelrot	128	0	0	128	0,0°
6	wdRed	Rot	255	0	0	255	0,0°
14	wdDarkYellow	Dunkelgelb	128	128	0	32 896	60,0°
7	wdYellow	Gelb	255	255	0	65 535	60,0°
4	wdBrightGreen	Hellgrün	0	255	0	65 280	120,0°
11	wdGreen	Grün	0	128	0	32768	120,0°
3	wdTurquoise	Türkis (cyan/hellblau)	0	255	255	16 776 960	180,0°
10	wdTeal	Blaugrün	0	128	128	8 421 376	180,0°
2	wdBlue	Blau	0	0	255	16 711 680	240,0°
9	wdDarkBlue	Dunkelblau	0	0	128	8 388 608	240,0°
12	wdViolet	Violett *(Magenta)*	128	0	128	8 388 736	300,0°
5	wdPink	~~Rosa~~ (Magenta)	255	0	255	16711935	300,0°

[a] WdbyAuthor: *Zeigt bei Änderungsverfolgung den jeweiligen Autor durch eine Farbe an (nicht getestet)*

Für Tabellenschattierungen gibt es in Word einige weitere ColorIndices, wenn auch nicht so viele wie in Excel. Sie sind in dem Makro für Farbanalyse und Kontrast berücksichtigt.

MsgBox Selection.Cells.Shading.BackgroundPatternColorIndex

Es gibt 60 Farbbezeichnungen in Word (siehe folgende Tabelle). Ferner gibt es tausende Farbcodes ohne Name.

Bei "wdColorAutomatic" gibt es einen negativen Code und bei der Farbe "Weiß" sowohl einen positiven als auch einen negativen Code.

Bei einigen dunklen Hintergrundsfarben wird die Schriftfarbe statt in der gewählten Schriftfarbe in Weiß dargestellt, etwa bei den Hintergrundsfarben "wdColorBlack" und "wdColorBlue" und "wdColorGray80", nicht jedoch bei den weniger dunklen Schattierungen wie wdColorGray75.

```
MsgBox "Der Text der markierten Stelle hat die Farbe: " & Selection.Font.Color
MsgBox "Die Schattier-Farbe hat hier den Code: " &
        Selection.Font.Shading.BackgroundPatternColor
```

Tabelle 15.2: Die 60 Word-Farben mit RGB-Wert und Farkreis-Winkel (1 von 2)

Schattierungs-Farbe	Textfarbe	Deutsch	Code ("long")	RGB - Rot	RGB-Grün	RGB - Blau	RGB-Hex	Farb-kreis-winkel (Hue)
wdColor Automatic *(hier weiß)*	wdColor Automatic *(hier schwarz)*	Automatisch Standard;	- 16777216	–	–	–	–	–
wdColorBlack	wdColorBlack	Schwarz	0	0	0	0	# 00 00 00	–
wdColorGray05	wdColorGray05	Grau 05	15987699	243	243	243	# F3 F3 F3	–
wdColorGray10	wdColorGray10	Grau 10	15132390	230	230	230	# E6 E6 E6	–
wdColorGray125	wdColorGray125	Grau 12,5	14737632	224	224	224	# E0 E0 E0	–
wdColorGray15	wdColorGray15	Grau 15	14277081	217	217	217	# D9 D9 D9	–
wdColorGray20	wdColorGray20	Grau 20	13421772	204	204	204	#CCCCCC	–
wdColorGray25	wdColorGray25	Grau 25	12632256	192	192	192	# C0 C0 C0	–
wdColorGray30	wdColorGray30	Grau 30	11776947	179	179	179	# B3 B3 B3	–
wdColorGray35	wdColorGray35	Grau 35	10921638	166	166	166	# A6 A6 A6	–
wdColorGray375	wdColorGray375	Grau 37,5	10526880	160	160	160	# A0 A0 A0	–
wdColorGray40	wdColorGray40	Grau 40	10066329	153	153	153	# 99 99 99	–
wdColorGray45	wdColorGray45	Grau 45	9211020	140	140	140	# 8C 8C 8C	–
wdColorGray50	wdColorGray50	Grau 50	8421504	128	128	128	# 80 80 80	–
wdColorGray55	wdColorGray55	Grau 55	7566195	115	115	115	# 73 73 73	–
wdColorGray60	wdColorGray60	Grau 60	6710886	102	102	102	# 66 66 66	–
wdColorGray625	wdColorGray625	Grau 62,5	6316128	96	96	96	# 60 60 60	–
wdColorGray65	wdColorGray65	Grau 65	5855577	89	89	89	# 59 59 59	–
wdColorGray70	wdColorGray70	Grau 70	5000268	76	76	76	# 4C 4C 4C	–
wdColorGray75	wdColorGray75	Grau 75	4210752	64	64	64	# 40 40 40	–
wdColorGray80	wdColorGray80	Grau 80	3355443	51	51	51	# 33 33 33	–
wdColorGray85	wdColorGray85	Grau 85	2500134	38	38	38	# 26 26 26	–
wdColorGray875	wdColorGray875	Grau 87,5	2105376	32	32	32	# 20 20 20	–
wdColorGray90	wdColorGray90	Grau 90	1644825	25	25	25	# 19 19 19	–
wdColorGray95	wdColorGray95	Grau 95	789516	12	12	12	# 0C 0C 0C	–
wdColorWhite	wdColorWhite	Weiß	16777215 *- 603914241*	255	255	255	# FF FF FF	–
wdColorDarkRed	wdColorDarkRed	Dunkelrot	128	128	0	0	# 80 00 00	0,0°
wdColorRed	wdColorRed	Rot	255	255	0	0	# FF 00 00	0,0°

Tabelle 16.2 (2 von 2)

Schattierungs-Farbe	Textfarbe	Deutsch	Code ("long")	RGB - Rot	RGB-Grün	RGB - Blau	RGB-Hex	Farb-kreis-winkel (Hue)
wdColorBrown	wdColorBrown	Braun	13209	153	51	0	# 99 33 00	20,0°
wdColorOrange	wdColorOrange	Orange	26367	255	102	0	# FF 66 00	24,0°
wdColorTan	wdColorTan	Hellbraun	10079487	255	204	153	# FF CC 99	30,0°
wdColorLightOrange	wdColorLightOrange	Hellorange	39423	255	153	0	# FF 99 00	36,0°
wdColorGold	wdColorGold	Gold	52479	255	204	0	# FF CC 00	48,0°
wdColorDarkYellow	wdColorDarkYellow	Dunkelgelb	32896	128	128	0	# 80 80 00	60,0°
wdColorLightYellow	wdColorLightYellow	Hellgelb	10092543	255	255	153	# FF FF 99	60,0°
wdColorOliveGreen	wdColorOliveGreen	Olivgrün	13107	51	51	0	# 33 33 00	60,0°
wdColorYellow	wdColorYellow	Gelb	65535	255	255	0	# FF FF 00	60,0°
wdColorLime	wdColorLime	Gelbgrün	52377	153	204	0	# 99 CC 00	75,0°
wdColorLightGreen	**wdColorLightGreen**	Hellgrün	13434828	204	255	204	#CCFFCC	120,0°
wdColorBrightGreen	wdColorBrightGreen	Hellgrün	65280	0	255	0	# 00 FF 00	120,0°
wdColorDarkGreen	wdColorDarkGreen	Dunkelgrün	13056	0	51	0	# 00 33 00	120,0°
wdColorGreen	wdColorGreen	Grün Dunkelgrün!	32768	0	128	0	# 00 80 00	120,0°
wdColorSeaGreen	wdColorSeaGreen	Seegrün	6723891	51	153	102	# 33 99 66	150,0°
wdColorAqua	wdColorAqua	Aquamarin	13421619	51	204	204	# 33 CC CC	180,0°
wdColorLightTurquoise	wdColorLightTurquoise	Helles Cyan	16777164	204	255	255	# CC FF FF	180,0°
wdColorTeal	wdColorTeal	Blaugrün	8421376	0	128	128	# 00 80 80	180,0°
wdColorTurquoise	wdColorTurquoise	Cyan	16776960	0	255	255	# 00 FF FF	180,0°
wdColorSkyBlue	wdColorSkyBlue	Himmelblau	16763904	0	204	255	# 00 CC FF	192,0°
wdColorDarkTeal	wdColorDarkTeal	Dunkles Blaugrün	6697728	0	51	102	# 00 33 66	210,0°
wdColorPaleBlue	wdColorPaleBlue	Blassblau	16764057	153	204	255	# 99 CC FF	210,0°
wdColorLightBlue	wdColorLightBlue	Hellblau	16737843	51	102	255	# 33 66 FF	225,0°
wdColorBlue	wdColorBlue	Blau	16711680	0	0	255	# 00 00 FF	240,0°
wdColorBlueGray	wdColorBlueGray	Blaugrau	10053222	102	102	153	# 66 66 99	240,0°
wdColorDarkBlue	wdColorDarkBlue	Dunkelblau	8388608	0	0	128	# 00 00 80	240,0°
wdColorIndigo	wdColorIndigo	Indigoblau	10040115	51	51	153	# 33 33 99	240,0°
wdColorLavender	wdColorLavender	Flieder	16751052	204	153	255	# CC 99 FF	270,0°
wdColorPink	wdColorPink	Rosa (Magenta)	16711935	255	0	255	# FF 00 FF	300,0°
wdColorViolet	wdColorViolet	Violett	8388736	128	0	128	# 80 00 80	300,0°
wdColorPlum	wdColorPlum	Pflaume	6697881	153	51	102	# 99 33 66	330,0°
wdColorRose	wdColorRose	Rose	13408767	255	153	204	# FF 99 CC	330,0°

15.2 Farbanalyse des markierten Zeichens, mit Ausgabe von RGB, CMYK, HSL und HSV sowie Grauwerten (auch bei "Theme colours") mit Prüfung des Kontrastes

Das folgende, aufwendige Makro zur Farbanalyse bei einem markierten Zeichen ist vielleicht für **Farbenblinde** nützlich.

Es werden, soweit vorhanden, die Word-Farbnamen zweisprachig, ansonsten werden die Farbcodes und die Farbe des betreffenden Farbkreis-Bereiches angegeben.

Es wird die Textfarbe, die farbige Hervorhebung bzw. Schattierung des Zeichens, die Schattierung des Absatzes und ggf. die Tabellenzellen-Schattierung angegeben. Ferner werden die Grautöne von Zeichen und Hintergrund berechnet, und es wird angegeben, ob der Kontrast ausreichend ist.

Die Farbe wird als RGB-Wert, als HSL-, HSV-Koordinaten, und als CMYK-Werte ausgegeben.
Üblich sind Zahlenwerte von Null bis 255 (8 Bit). Nur der Farbton ("Hue") wird dabei als Winkel in Grad von 0 ° (rot) bis 360° (wieder rot) ausgegeben (auch in Photoshop) [11].

15.2.1 Gelöste Probleme bei der Farbanalyse wegen "Theme colours"

Mein altes Makro für die Farbanalyse lieferte in Word 2010 plötzlich bei neuen Word-Dateien unsinnige RGB-Farbwerte. Eindeutig bläuliche Buchstaben hatten angeblich keinen Blauanteil usw.

Ursache: [12]
Seit Word 2007 hat Microsoft ein anderes Farbcodier-System. Wenn die Farben nur Codes der neuen Windows-Tabelle enthalten ("Theme colours"), etwa in docx-Dateien (oder ungespeichert in doc-Dateien), gibt es keine absolute Farbe, die sich per Makro in RGB umrechnen lässt. Mit "Schriftartfarbe: Hintergrund 2" oder "Akzent 2" lässt sich meines Erachtens wenig anfangen, siehe das folgende Bild.

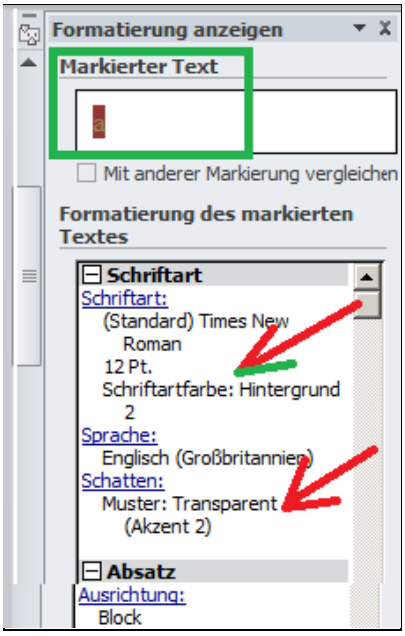


Bild 15.2: Screenshots bei "Theme colours"

[11] Bei Microsoft-Word und in Paint fand ich allerdings eine Ausgabe des Farbtons ("Hue") als 8-Bit-Wert von "0 bis 255".
[12] Siehe Toni Jollans: http://www.wordarticles.com/Articles/Colours/2007.php, Abruf März 2017.

Abhilfe: Beim Button für die Textfarbe oder beim Button für die Schattierung auf Weitere Farben… drücken, dann Benutzerdefiniert. Dann kann man die Farbkomponenten wahlweise als "RGB" (siehe Bild) oder als "HSL" auslesen.

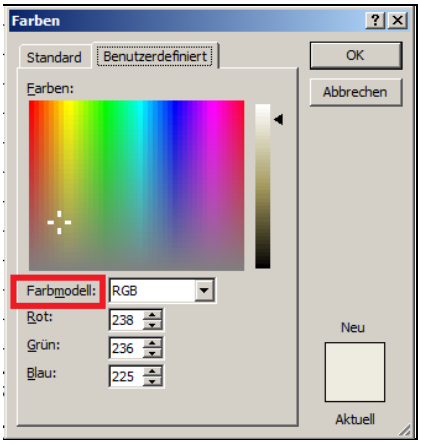


Bild 15.3: Screenshots bei "Theme colours"

Mein Versuch, die RGB-Werte auch bei "Theme Colours" direkt per VBA-Makro zu bestimmen, misslang mir zuerst. "Selection.Font.Color" usw. ergibt eine hohe negative Zahl (im Hexadezimalsystem 8 Ziffern statt 6 Ziffern, es fehlen die Ziffern für die Farbe Blau). Siehe die beiden folgenden Beispiele:

wdColorAutomatic:
Hex(-16 777 216) = FF000000 ' (wdColorAutomatic)
Andere Hex-Schreibweise: 0xFF000000

wdColorWhite:
Hex(-603 914 241) = DC00FFFF ' (wdColorWhite)
Hex(16 777 215) = FFFFFF ' auch (wdColorWhite)

hellblaue Farbe *(aus einer Internet-Seite)*
hex(-587 137 025) = DD00FFFF
hex(13 411 903) = CCA63F

Mögliche Lösungen:

a) Makro abbrechen (**unbefriedigende** "Null-Lösung");

```
If Selection.Font.Color < 0 Then
  ' "Theme colours" in docx-Dateien (oder ungespeichert in sonstigen), keine absolute Farbe!
  MsgBox "negativer Wert!", vbCritical
  Exit Sub
End If
```

b) Nur den "ColorIndex" auswerten (nur Grundfarbe), daraus ergeben sich aber **keine brauchbaren** Grauwerte;

```
MsgBox Selection.Font.ColorIndex
```

c) Die Datei als "Word 97-2003" speichern (.doc), dann schließen und letztere wieder öffnen;

```
' Druckdatei im alten Word97-2003-Format erzwingen:
ActiveDocument.SaveAs2 FileName:=ActiveDocument.Path _
& "\" & ActiveDocument.Name & ".doc", _
FileFormat:=wdFormatDocument, LockComments:=False, _
Password:="", AddToRecentFiles:=True, WritePassword:="", _
ReadOnlyRecommended:=False, EmbedTrueTypeFonts:=False, _
SaveNativePictureFormat:=False, SaveFormsData:=False, _
SaveAsAOCELetter:=False, CompatibilityMode:=0
```

d) Markiertes Zeichen (bzw. Absatz oder Tabellenzelle) in eine temporäre Datei (Zwischendatei ist kompatibel mit "Word 97-2003") kopieren, dort analysieren, dann Zwischendatei wieder schließen;

```
If Selection.Font.Color < 0 Then
  ' Umweg über temporäre Datei, um aus Theme-Color eine RGB-Farbe zu machen:
 Selection.Copy
   Set Dok1 = ActiveDocument
   Set Dok2 = Documents.Add(DocumentType:=wdNewBlankDocument)
   Dok2.Activate
   Selection.WholeStory
   Selection.Paste
   MsgBox Selection.Font.Color
   ' Zwischendatei schließen:
   Dok2.Close (wdDoNotSaveChanges)
   ' In das ursprüngliche Dokument zurückkehren:
   Dok1.Activate
 End If
```

15.2.2 Das fertige Makro zur Analyse von Farben mit Ausgabe von RGB-, CMYK-, HSL- und Grauwerten sowie Prüfung des Kontrastes (auch bei Theme-Colors)

```
Option Explicit

' Für Farbanalyse und Farbenreduzierung:
Private lngGrauwert255 As Long
Private IntFarbwinkel As Integer

' Für erweiterte MsgBox mit Unicodezeichen und mehr Zeilen:
Private Declare Function MessageBoxW Lib "user32.dll" _
   (ByVal hwnd As Long, _
   ByVal lpText As Long, _
   ByVal lpCaption As Long, _
   ByVal uType As Long) As Long


Sub Farben_und_Kontrast_ausgeben_an_Markierung()
' Hanna-Chris Gast, 2015, weitgehend neu: Februar bis Juni 2017.
' Korrekturen am 26.11.2020 und 25.04.2023,
' weil der Befehl "Set Dok2 = Documents.Add"   keine Dateibezeichnung mehr zuwies.
'
' Dies Makro gibt alle Farb-Codes (auch RGB, CMYK, HSL und HSV/HSB) eines Zeichens aus.
' Zusätzlich werden auch der Grauwert (die Helligkeit) und der Kontrast angegeben.
'
' Sind mehrere Zeichen markiert, wird die Markierung auf ein Zeichen verkleinert.
' Die Farbe des Hintergrunds eines Schriftzeichens kann eine Farbmarkierung ("Hervorhebung",
' "Highlight") sein, aber auch eine Hintergrundschattierung.
' Ferner gibt es Hintergrundschattierungen von Absätzen oder von Tabellenzellen.
' Außerdem kann es "Textur" mit einer zugehörigen Vordergrundschattierung geben.
'
' Berechnung der RGB-Farbwerte sowie des Grauwertes (Helligkeit) für Text und Hintergrund.
' Es werden bei 60 Word-Farben auch die Farbnamen (englisch und deutsch) angegeben.
' Ansonsten wird die ungefähre Farbe des Farbkreisbereichs angegeben.
'
' Wenn die untersuchte Farbe keine RGB-Farben ist ("Designfarben", "Theme colours")
' etwa in docx-Dateien oder ungespeichert in doc-Dateien, gibt es keine absolute Farbe.
' Über eine temporäre Zwischendatei wird jetzt vom Makro eine entsprechende RGB-Farbe
```

```
' erzeugt und analysiert (März 2017).
'
' Bei einigen Steuerzeichen werden auf dem Bildschirm und/oder im Druck keine Farben angezeigt,
' bei Absatzmarken keine Farbmarkierung, bei Tabulator keine Textfarbe, usw.
' Hier wird deshalb auf die Angabe des Kontrastes verzichtet.
'
' Die Farbe "Cyan", 00FFFF (100 Prozent grün und 100 Prozent blau) empfinde ich
' eher als himmelblau, nicht als Türkis. Als Türkis empfinde ich (wie einige andere Menschen auch)
' nur Farben, die mehr Grünanteil als Blauanteil haben.
'
' Hinweis, wenn Mindestkontrast (Helligkeit Zeichen minus Helligkeit Hintergrund = 125 von 255)
' nicht eingehalten wird, Rangfolge: Highlight, Zeichenhintergrund, Absatz, Tabelle.
' Ferner sollte die Summe der absoluten Differenzen der drei Farbanteile (rot, grün, blau)
' mindestens 500 betragen (in diesem Makro weggelassen, da ich dies für zweitrangig halte).
'
' Mankos:
' Wenn Word bei dunklem Hintergrund (Highlight) die Schriftfarbe als Weiß darstellt,
' wird das vom Makro nicht erkannt, sondern es wird die eigentliche Textfarbe berechnet.
' Auch Farben der Änderungskennung werden nicht erkannt.
' Bei Schraffuren geben ich teilweise nur die Farbindex-Farbe an.
' Bei vertikal verbundenen  Tabellenzellen mit unterschiedlichen Zell-Schattierungen
' lässt sich mit dem Makro die Farbe nicht sicher bestimmen.
'
' Die Farbindex-Farbe entspricht leider nicht immer der empfundenen Farbe,
' deshalb gebe ich bei unbekannten Farben weitgehend den Farbkreis-Bereich an.
' Bei der Farbkreis-Einteilung richtete ich mich nach den benannten Webfarben.
'
' RGB-Formel aus http://www.office-loesung.de/ftopic421244_0_0_asc.php.
' HSL-Formel aus http://www.vb-helper.com/howto_rgb_to_hls.html
' Zahlenwert für Mindestkontrast und Formel für "Color Brightness"
' nach https://www.w3.org/TR/AERT (auch in anderen Quellen gefunden, etwa in
' https://www.xing.com/communities/posts/farbenlehre-fuer-vba-1002110463).
'
' Gewähltes Tastenkürzel:  AltGr + Num9
' ********************************************************
' Datentyp Long für Ganzzahlen von -2'147'483'648 bis 2'147'483'647
Dim lngColorText As Long
Dim lngColorHtg As Long  ' Hintergrund
Dim lngGrauwertText As Long
Dim lngGrauwertHtg As Long
Dim lngGrauwertHtg_Resultierend As Long
Dim lngColorPara As Long
'
Dim strColor As String ' Zwischenwert
Dim strMsgSteuerzeichen As String
Dim strMsgZeichen As String  ' Farbe des Zeichens
Dim strMsg1Vg As String  ' Vordergrundschattierung/Schraffur des Zeichens
Dim strMsg1Hg As String  ' Hintergrundschattierung des Zeichens
Dim strMsgAbsVg As String  ' Absatz-Schraffur
Dim strMsgAbsHg As String  ' Absatzschattierung
Dim strMsgTabVg As String  ' Tabellen-Vordergrundschattierung
Dim strMsgTabHg As String  ' Tabellen-Hintergrundschattierung
Dim strMsgKontrast As String  ' Kontrast-Meldung
Dim StrErgebnis As String
'
Dim Rng0 As Range
'
Dim Dok1 As Word.Document
Dim Dok2 As Word.Document
'  ******************************************************************
'  Fehlerbehandlung:
"On Error GoTo Zeile_Fehler1
```

```
' ********************************************************************
' Markierung auf ein einziges Zeichen einschränken:
' ********************************************************************
' (Bei unterschiedlichen Farben funktioniert die Farbbestimmung nicht)
  Selection.SetRange Start:=Selection.Start, End:=Selection.Start + 1
' Art der Markierung prüfen und bei Bildern usw. abbrechen:
  If Selection.Type < 2 Then
  ' Wenn nichts markiert ist, 1 Zeichen links vom Cursor markieren:
    Set Rng0 = Selection.Range
    Rng0.SetRange Start:=Rng0.End - 1, End:=Rng0.End
    Rng0.Select
  "ElseIf Selection.Type = 8 Then
    " MsgBox "Es ist eine frei positionierte Grafik oder deren Ankerpunkt markiert." _
    " & vbCrLf & "Bitte ändern sie die Cursorstelle! , Exclamation"
    " Exit Sub
  ' (funktioniert in Word 2010 nicht mehr)
  ElseIf Selection.Type = 7 Then
    MsgBox "Es ist Grafik oder eine Formel (InlineShape) markiert." & vbCrLf _
        & "Bitte ändern sie die Cursorstelle!", vbExclamation
    Exit Sub
  Else
   Set Rng0 = Selection.Range
  End If
  ' Shapes ausschließen:
  If Rng0.ShapeRange.Count > 0 Then
   If Rng0.ShapeRange(1).Type = msoPicture Then
     MsgBox "Es ist ein frei positioniertes Bild markiert." & vbCrLf _
     & "Bitte ändern sie zur Farbbestimmung die Cursor-Position!", vbExclamation
   Else
     MsgBox "Es ist ein Bildelement (Shape) markiert." & vbCrLf _
     & "Bitte ändern sie zur Farbbestimmung die Cursor-Position!", vbExclamation
   End If
   Exit Sub
  End If
' ****************************
' Markierung nachjustieren (bei leeren Tabellenzellen nötig):
  If Len(Rng0.Text) > 2 Then
     Rng0.Collapse Direction:=wdCollapseStart
     Rng0.SetRange Start:=Rng0.Start, End:=Rng0.Start + 1
  End If
  Rng0.Select
' Kontrolle:
  If Len(Selection) > 2 Then
    MsgBox "Die Markierung enthält zuviel Text (Feld oder leere Tabellenzelle)!", vbCritical
    Exit Sub
  ElseIf Len(Selection) = 0 Then
    MsgBox "Es ist kein Zeichen markiert!", vbCritical
    Exit Sub
  End If
' ****************
' Bei einigen Steuerzeichen sind keine Farben oder Farbmarkierungen sichtbar:
  If Asc(Rng0.Text) < 14 Then
    strMsgSteuerzeichen = vbCrLf & vbCrLf & "Achtung: Dies ist ein Steuerzeichen.  " _
    & "Bei einigen Steuerzeichen werden auf dem Bildschirm bzw. im Druck keine Farben angezeigt!"
  Else
    strMsgSteuerzeichen = ""
  End If
' ********************************************************************
' Lästiges "Flackern" des Bildschirms verhindern:
  Application.ScreenUpdating = False
' ********************************************************************
' Textfarbe ermitteln (strMsgZeichen)
```

```
'  ******************************************************************
  lngColorText = Rng0.Font.Color
  If lngColorText = -603914241 Then
     ' Korrektur für negative Zahl bei "weiß":
     lngColorText = 16777215 ' weiß
  End If
  If lngColorText = -16777216 Then
    ' Word-Farbe "automatisch":
    ' Ich setze hier bei der Textfarbe "automatisch" für die Kontrastberechnung die Farbe "schwarz"
    ' (Bei Hintergründen ist es anders, da ist die Farbe des Druckerpapier zu wählen).
    lngColorText = 0  ' als Schwarz ansetzen statt "wdColorAutomatic".
    ' Auch der Grauwert ist dann 0.
    lngGrauwertText = 0
    strMsgZeichen = "Die Farbe des markierten Textes ist: ""wdColorAutomatic"" (schwarz);" _
      & vbCrLf _
      & "RGB-Werte: Rotanteil = 0, Grünanteil = 0, Blauanteil = 0;" & vbCrLf _
      & "Als HSL-Koordinaten sind es:" & vbCrLf _
      & "Farbkreiswinkel Hue =  - ; Sättigung S = 0,00; Intensität L = 0,00." & vbCrLf _
      & "Der Grauwert (Helligkeit) ist 0 von 255,  bzw. 0 %."
    '  **********
    ElseIf lngColorText < 0 Then
    '  "Theme colours" in docx-Dateien (oder ungespeichert in sonstigen), keine absolute Farbe!
       '' MsgBox "negativer Wert!", vbCritical
       '' Exit Sub
      ' Umweg über Zwischendatei, um aus Theme-Color eine RGB-Farbe zu machen:
      On Error GoTo Zeile_Fehler2
      Rng0.Select
      Selection.Copy
      Set Dok1 = ActiveDocument
     ' Set Dok2 = Documents.Add(DocumentType:=wdNewBlankDocument, Visible:=False)
      Set Dok2 = Documents.Add
'' ***********************************************************************************
''      Hier neuer Fehler aufgetaucht(24.6.2021, Lösung 25.04.2023):
''      Der Name "Dok2" ist mit "Set Dok2 =..." nicht mehr zuweisbar, vielmehr gibt es
''      je nach Sitzungs-Vorgeschichte nur noch "Dokument2", "Dokument3" usw.
''      Aber es wird eine neue Datei erzeugt und aktiviert. Lösung:
      If Dok1.Name = ActiveDocument.Name Then
         MsgBox "Fehler im Makro!"
         Exit Sub
      Else
         Set Dok2 = ActiveDocument
      End If
'' ***************************************************************************************
      ' Falls in der Normal.dot schon ein Briefkopf vorgesehen ist, diesen löschen:
     Selection.WholeStory
      Selection.Delete
      Selection.Paste
      ' ***
      ' Textfarbe im temporären Dokument neu bestimmen:
      lngColorText = Selection.Font.Color
      strColor = Fkt_Farbname_aus_Farbcode(lngColorText)
      If strColor = "." Then
         strColor = "Farbindex (ColorIndex) = " & Rng0.Font.ColorIndex & " bzw. " _
         & Fkt_Farbname_FarbIndex(Rng0.Font.ColorIndex)
         ' *****
         ' Statt den ColorIndex besser den Farbkreisbereich angeben:
         strColor = strColor & vbCrLf & Fkt_Farbkreisbereich(lngColorText)
         ' *****
      Else
         strColor = "dies entspricht der Word-Farbe: " & strColor
      End If
      strMsgZeichen = _
```

```
        "Die Farbe des markierten Textes ist keine RGB-Farbe (Theme-Color);" & vbCrLf _
         & strColor & vbCrLf _
         & Fkt_RGB_HSL_Farbanteile(lngColorText)
        ' ***
'       Zwischendatei schließen:
        '   On Error Resume Next
        '' Dokument2.Close (wdDoNotSaveChanges) ' Es kann auch "Dokument3" usw. heißen!
        Dok2.Close (wdDoNotSaveChanges)
        On Error GoTo Zeile_Fehler2
        ' In das ursprüngliche Dokument zurückkehren:
        Dok1.Activate
        ' Der Grauwert:
        lngGrauwertText = lngGrauwert255
      ' **********
      Else
      ' normale Farben (einschließlich weiß und schwarz)
      ' ************************************
        strColor = Fkt_Farbname_aus_Farbcode(lngColorText)
        If strColor = "." Then
          strColor = "Farbindex (ColorIndex) = " & Rng0.Font.ColorIndex & " bzw. " _
          & Fkt_Farbname_FarbIndex(Rng0.Font.ColorIndex)
          ' *****
          ' Statt den ColorIndex besser den Farbkreisbereich angeben:
          strColor = strColor & vbCrLf & Fkt_Farbkreisbereich(lngColorText)
          ' *****
        Else
          strColor = "dies entspricht der Word-Farbe: " & strColor
        End If
        strMsgZeichen = "Der Farb-Code des markierten Textes ist: " & lngColorText & ";" & vbCrLf _
        & strColor & vbCrLf _
        & Fkt_RGB_HSL_Farbanteile(lngColorText)
      ' Der Grauwert:
        lngGrauwertText = lngGrauwert255
      End If
' ***********************************************************************
' Markierfarbe (Highlight) bzw. Hintergrundsschattierung (strMsg1Hg)
' ***********************************************************************
  If Rng0.HighlightColorIndex <> 0 Then
  ' ColorIndex der Markierfarbe des Ranges in Farbwerte umwandeln:
    Select Case Rng0.HighlightColorIndex
     ' (Case = 0, wenn keine Markierfarbe angewendet wurde)
     Case 1
      lngColorHtg = 0
     Case 2
      lngColorHtg = 16711680
     Case 3
      lngColorHtg = 16776960
     Case 4
       lngColorHtg = 65280
     Case 5
      lngColorHtg = 16711935 ' rosa/Magenta
     Case 6
      lngColorHtg = 255
     Case 7
      lngColorHtg = 65535
     Case 8
      lngColorHtg = 16777215 ' weiß
     Case 9
      lngColorHtg = 8388608
     Case 10
      lngColorHtg = 8421376
     Case 11
```

```
      lngColorHtg = 32768
    Case 12
      lngColorHtg = 8388736
    Case 13
      lngColorHtg = 128
    Case 14
      lngColorHtg = 32896 ' dunkelgelb
    Case 15
      lngColorHtg = 8421504
    Case 16
      lngColorHtg = 12632256
    Case Else
      ' (kommt hier nicht vor)
  End Select
  strMsg1Hg = "Die Markierfarbe (HighlightColorIndex) hat die Nummer: " _
    & Rng0.HighlightColorIndex & ";" _
  & vbCrLf _
  & "dies entspricht der Word-Farbe: " & Fkt_Farbname_FarbIndex(Rng0.HighlightColorIndex)
Else
'  Keine Farbmarkierung (Highlight) vorhanden, deshalb Hintergrundsschattierung bestimmen:
  lngColorHtg = Rng0.Font.Shading.BackgroundPatternColor
  If lngColorHtg = -603914241 Then
    ' Korrektur für negative Zahl bei "weiß" ("Theme colours"):
    lngColorHtg = 16777215 ' weiß
  End If
  ' Fallunterscheidung:
  If lngColorHtg = -16777216 Then
    ' "automatisch" = Keine Hintergrundschattierung.
    lngGrauwertHtg_Resultierend = -1
    strMsg1Hg = "Der Text-Hintergrund hat keine Farbmarkierung und keine Schattierung."
    ' Ich setze hier für die Kontrastberechnung bei Hintergrund Automatisch = weiß:
    lngColorHtg = 16777215 ' weiß
  '  ******************************
  ElseIf lngColorHtg < 0 Then
  '  "Theme colours" in docx-Dateien (oder ungespeichert in sonstigen), keine absolute Farbe!
    '' MsgBox "negativer Wert!", vbCritical
    '' Exit Sub
    ' Umweg über Zwischendatei, um aus Theme-Color eine RGB-Farbe zu machen:
    Rng0.Select
    Selection.Copy
    Set Dok1 = ActiveDocument
    Set Dok2 = Documents.Add  "(DocumentType:=wdNewBlankDocument, Visible:=False)
    "  Leider funktioniert das nicht mehr mit Namenszuweisung,
    "  Es wird nur eine neue Datei erzeugt und aktiviert, aber ohne die Bezeichnung "Dok2";
    "  deshalb die folgende Zeile ergänzt! (26.11.2020).
    Set Dok2 = ActiveDocument
    On Error GoTo Zeile_Fehler2
    Dok2.Activate
    ' Falls in der Normal.dot schon ein Briefkopf vorgesehen ist, diesen löschen:
    Selection.WholeStory
    Selection.Delete
    Selection.Paste
    Selection.WholeStory
    Selection.SetRange Start:=Selection.Start, End:=Selection.Start + 1
    lngColorHtg = Selection.Font.Shading.BackgroundPatternColor
    ' **********
    ' Farbiger Hinterrund des Zeichens:
    strColor = Fkt_Farbname_aus_Farbcode(lngColorHtg)
    If strColor = "." Then
      strColor = _
      "Farbindex (ColorIndex) = " & Rng0.Font.Shading.BackgroundPatternColorIndex _
      & " bzw. " _
```

```
        & Fkt_Farbname_FarbIndex(Rng0.Font.Shading.BackgroundPatternColorIndex)
        ' *****
        ' Statt den ColorIndex besser den Farbkreisbereich angeben:
        strColor = strColor & vbCrLf & Fkt_Farbkreisbereich(lngColorHtg)
        ' *****
      Else
        strColor = "dies entspricht der Word-Farbe: " & strColor
      End If
      strMsg1Hg = "Die Farbe des Hintergrunds ist keine RGB-Farbe (Theme-Color);" _
      & vbCrLf & strColor
      '  Zwischendatei schließen:
        Dok2.Close (wdDoNotSaveChanges)
"         On Error GoTo Zeile_Fehler1
      '  In das ursprüngliche Dokument zurückkehren:
        Dok1.Activate
    '  ******************************
    Else
      ' Farbiger Hinterrund des Zeichens (direkt auswertbarer Code):
      strColor = Fkt_Farbname_aus_Farbcode(lngColorHtg)
      If strColor = "." Then
        strColor = "Farbindex (ColorIndex) = " _
        & Rng0.Font.Shading.BackgroundPatternColorIndex & " bzw. " _
        & Fkt_Farbname_FarbIndex(Rng0.Font.Shading.BackgroundPatternColorIndex)
        ' *****
        ' Statt den ColorIndex besser den Farbkreisbereich angeben:
        strColor = strColor & vbCrLf & Fkt_Farbkreisbereich(lngColorHtg)
        ' *****
      Else
        strColor = "dies entspricht der Word-Farbe: " & strColor
      End If
      strMsg1Hg = "Der Farb-Code des Hintergrunds ist: " & lngColorHtg & ";" _
      & vbCrLf & strColor
    End If
  End If
'  ************************************
  If strMsg1Hg <> "Der Text-Hintergrund hat keine Farbmarkierung und keine Schattierung." Then
    strMsg1Hg = strMsg1Hg _
    & vbCrLf _
    & Fkt_RGB_HSL_Farbanteile(lngColorHtg)
    ' Grauwert des Zeichenhintergrunds:
    lngGrauwertHtg = lngGrauwert255
    lngGrauwertHtg_Resultierend = lngGrauwert255
  Else
    lngGrauwertHtg_Resultierend = -1
  End If
'  ********************************************************************
'  Textur und oder Vordergrundschattierung des Textes (strMsg1Vg)
'  ********************************************************************
  If Rng0.Font.Shading.ForegroundPatternColorIndex <> 0 _
  Or Rng0.Font.Shading.Texture <> 0 Then
    If Rng0.Font.Shading.Texture = 0 Then
      ' Vordergrundschattierung ohne Textur vorhanden (ist unsichtbar):
      strMsg1Vg = _
      "Das Zeichen hat Vordergrundsschattierung, ColorIndex = " _
      & Rng0.Font.Shading.ForegroundPatternColorIndex & "," & vbCrLf _
      & Fkt_Farbname_FarbIndex(Rng0.Font.Shading.ForegroundPatternColorIndex) & " " _
      & "Diese kann man aber mangels Textur nicht sehen."
    ElseIf Rng0.Font.Shading.ForegroundPatternColorIndex = 0 Then
      ' Textur ohne Vordergrundschattierung vorhanden:
      strMsg1Vg = _
      "Das Zeichen hat Textur, Code = " & Rng0.Font.Shading.Texture & "."
      lngGrauwertHtg_Resultierend = -2
```

```
      Else
        ' Textur und/oder Vordergrundschattierung vorhanden:
        strMsg1Vg = _
        "Das Zeichen hat Textur, Code = " & Rng0.Font.Shading.Texture & ", " _
        & vbCrLf & "mit Vordergrundsschattierung, ColorIndex = " _
        & Rng0.Font.Shading.ForegroundPatternColorIndex & ", " _
        & "" _
        & Fkt_Farbname_FarbIndex(Rng0.Font.Shading.BackgroundPatternColorIndex)
        lngGrauwertHtg_Resultierend = -2
      End If
    Else
      strMsg1Vg = ""
    End If
  ' ********************************************************************
  ' Textur und/oder Vordergrundschattierung des Absatzes
  ' ********************************************************************
  ' (Hier werden nur die ColorIndex-Farben ermittelt)
    If Rng0.ParagraphFormat.Shading.ForegroundPatternColorIndex <> 0 _
    Or Rng0.ParagraphFormat.Shading.Texture <> 0 Then
      If Rng0.ParagraphFormat.Shading.Texture = 0 Then
        ' Vordergrundschattierung ohne Textur vorhanden (ist unsichtbar):
        strMsgAbsVg = vbCrLf & vbCrLf _
        & "Der Absatz hat Vordergrundsschattierung, ColorIndex = " _
        & Rng0.Font.Shading.ForegroundPatternColorIndex & "," & vbCrLf _
        & "(" _
        & Fkt_Farbname_FarbIndex(Rng0.Font.Shading.BackgroundPatternColorIndex) & ")," _
        & "die man aber mangels Textur nicht sehen kann."
       ' (Keinen Einfluss auf den Kontrast)
      ElseIf Rng0.ParagraphFormat.Shading.ForegroundPatternColorIndex = 0 Then
        ' Textur ohne Vordergrundschattierung vorhanden:
        strMsgAbsVg = vbCrLf & vbCrLf _
        & "Der Absatz hat Textur, Code = " & Rng0.ParagraphFormat.Shading.Texture & "."
        ' Zeichen wegen Schraffung schwer lesbar.
        If lngGrauwertHtg_Resultierend = -1 Then lngGrauwertHtg_Resultierend = -2
      Else
        ' Textur und/oder Vordergrundschattierung vorhanden:
        strMsgAbsVg = vbCrLf & vbCrLf _
        & "Der Absatz hat Textur, Code = " & Rng0.ParagraphFormat.Shading.Texture & ", " _
        & vbCrLf & "mit Vordergrundsschattierung, ColorIndex = " _
        & Rng0.ParagraphFormat.Shading.ForegroundPatternColorIndex & ", " _
        & Fkt_Farbname_FarbIndex(Rng0.ParagraphFormat.Shading.BackgroundPatternColorIndex)
        ' Zeichen wegen Schraffung schwer lesbar.
        If lngGrauwertHtg_Resultierend = -1 Then lngGrauwertHtg_Resultierend = -2
      End If
    Else
      strMsgAbsVg = ""
    End If
  ' ********************************************************************
  ' Hintergrund-Schattierungen des Absatzes (strMsgAbsHg)
  ' ********************************************************************
    lngColorPara = Selection.ParagraphFormat.Shading.BackgroundPatternColor
    If lngColorPara = wdColorAutomatic Then
      "strMsgAbsHg = "Der Absatz hat weder Farbmarkierung noch Hintergrundschattierung."
      strMsgAbsHg = ""
    ElseIf lngColorPara = -603914241 _
    Or lngColorPara = 16777215 Then
      strMsgAbsHg = vbCrLf & "Der Absatz hat eine weiße Hintergrundschattierung."
    "ElseIf lngColorPara = 9999999 Then
    "   strMsgAbsHg = vbCrLf _
    "   & "Der Absatz hat unterschiedliche Hintergrundschattierungen (9999999)!"
    If lngGrauwertHtg_Resultierend = -1 Then lngGrauwertHtg_Resultierend = lngGrauwert255
    ' *************
```

```
  ElseIf lngColorPara < 0 Then
    ' "Theme colours" in docx-Dateien (oder ungespeichert in sonstigen Dateien),
    ' keine absolute Farbe!
    ' Umweg über Zwischendatei, um aus Theme-Color eine RGB-Farbe zu machen:
    Rng0.Paragraphs(1).Range.Copy
    Set Dok1 = ActiveDocument
    Set Dok2 = Documents.Add(DocumentType:=wdNewBlankDocument, Visible:=False)
"     On Error GoTo Zeile_Fehler2
    Dok2.Activate
    ' Falls wie bei mir in der Normal.dot schon ein Briefkopf vorgesehen ist, diesen löschen:
    Selection.WholeStory
    Selection.Delete
    Selection.Paste
    Selection.WholeStory
    Selection.SetRange Start:=Selection.Start, End:=Selection.End - 1
    lngColorHtg = Selection.Font.Shading.BackgroundPatternColor
    '  **********
    ' Farbiger Hintergrund des Absatzes:
    lngColorPara = Selection.ParagraphFormat.Shading.BackgroundPatternColor
    strColor = _
    Fkt_Farbname_aus_Farbcode(Selection.ParagraphFormat.Shading.BackgroundPatternColor)
    If strColor = "." Then
       strColor = "Farbindex (ColorIndex) = " _
       & Rng0.ParagraphFormat.Shading.BackgroundPatternColorIndex & " bzw. " _
       & Fkt_Farbname_FarbIndex(Rng0.ParagraphFormat.Shading. _
       BackgroundPatternColorIndex)
      ' *****
      ' Statt den ColorIndex besser den Farbkreisbereich angeben:
      strColor = strColor & vbCrLf & Fkt_Farbkreisbereich(lngColorPara)
      ' *****
    Else
       strColor = "dies entspricht der Word-Farbe: " & strColor
    End If
    strMsgAbsHg = vbCrLf _
      & "Der Absatz hat eine Hintergrundschattierung (Theme Color);" & vbCrLf _
      & strColor & vbCrLf _
      & Fkt_RGB_HSL_Farbanteile(Selection.ParagraphFormat.Shading.BackgroundPatternColor)
    '  Zwischendatei schließen:
      Dok2.Close (wdDoNotSaveChanges)
"       On Error GoTo Zeile_Fehler1
    '  In das ursprüngliche Dokument zurückkehren:
      Dok1.Activate
    If lngGrauwertHtg_Resultierend = -1 Then lngGrauwertHtg_Resultierend = lngGrauwert255
  '  *************
  Else
    ' Absatzschattierung hat eine RGB-Farbe:
    strColor = _
    Fkt_Farbname_aus_Farbcode(Selection.ParagraphFormat.Shading.BackgroundPatternColor)
    If strColor = "." Then
       strColor = "Farbindex (ColorIndex) = " _
       & Rng0.ParagraphFormat.Shading.BackgroundPatternColorIndex & ") bzw. " _
       & Fkt_Farbname_FarbIndex(Rng0.ParagraphFormat.Shading. _
       BackgroundPatternColorIndex)
      ' *****
      ' Statt den ColorIndex besser den Farbkreisbereich angeben:
      strColor = strColor & vbCrLf & Fkt_Farbkreisbereich(lngColorPara)
      ' *****
    Else
       strColor = "dies entspricht der Word-Farbe: " & strColor
    End If
    strMsgAbsHg = vbCrLf _
      & "Der Absatz hat die Hintergrundschattierung: " _
```

```
        & Rng0.ParagraphFormat.Shading.BackgroundPatternColor & ";" & vbCrLf _
        & strColor & vbCrLf _
        & Fkt_RGB_HSL_Farbanteile(Selection.ParagraphFormat.Shading.BackgroundPatternColor) _
        & vbCrLf & vbCrLf
      If lngGrauwertHtg_Resultierend = -1 Then lngGrauwertHtg_Resultierend = lngGrauwert255
    End If
    strMsgAbsHg = strMsgAbsHg & strMsgAbsVg
'  ********************************************************************
'  Auf Schattierungen und Schraffuren von Tabellenzellen prüfen
'  ********************************************************************
'  Vordergrunds-Schattierungen und Schraffuren von Tabellenzellen prüfen (strMsgTabVg):
    If Selection.Information(wdWithInTable) Then
      ' Hierbei darf kein Zeichen markiert sein. Vorübergehend die Markierung reduzieren:
      "Set rng1 = Selection.Range
      Selection.Collapse
      ' ************************
      On Error Resume Next
'     **************************
      If Selection.Cells.Shading.ForegroundPatternColor = -1 Then
        If Err.Number > 0 Then
          "strMsgTabVg = vbCrLf & vbCrLf _
          "& "Es ist das Steuerzeichen für Tabellenzeilen-Ende markiert."
          "Exit Sub
          Debug.Print vbCrLf & vbCrLf _
             & "Es ist das Steuerzeichen für Tabellenzeilen-Ende markiert."
          Err.Clear
          strMsgTabVg = ""
        Else
          ' "Die Markierung enthält keine bestimmbare Tabellenzellen-Vordergrundschattierung."
           strMsgTabVg = ""
        End If
      ElseIf Selection.Cells.Shading.ForegroundPatternColor = -16777216 Then
         ' keine Vordergrundschattierung.
         strMsgTabVg = ""
      '  **********
      ElseIf Selection.Cells.Shading.ForegroundPatternColor < 0 Then
      '  "Theme colours" in docx-Dateien (oder ungespeichert in sonstigen), keine absolute Farbe!
        " MsgBox "negativer Wert!", vbCritical
        " Exit Sub
        ' farbige Zellenschattierung mit negativem Zahlenwert:
        ' Umweg über Zwischendatei, um aus Theme-Color eine RGB-Farbe zu machen:
        Rng0.Cells(1).Range.Copy
        Set Dok1 = ActiveDocument
        Set Dok2 = Documents.Add(DocumentType:=wdNewBlankDocument, Visible:=False)
"         On Error GoTo Zeile_Fehler2
        Dok2.Activate
        ' Falls in der Normal.dot schon ein Briefkopf vorgesehen ist, diesen löschen:
        Selection.WholeStory
        Selection.Delete
        Selection.Paste
        Dok2.Range.Tables(1).Select
        ' **********
        ' farbige Zellen-Vordergrunds-Schattierung (nur bei Textur sichtbar):
        strColor = Fkt_Farbname_aus_Farbcode(Selection.Cells.Shading.ForegroundPatternColor)
        If strColor = "." Then
          strColor = "Farbindex (ColorIndex) = " _
          & Selection.Cells.Shading.ForegroundPatternColorIndex & " bzw. " _
          & Fkt_Farbname_FarbIndex(Selection.Cells.Shading.ForegroundPatternColorIndex)
         Else
         ' Word-Farbe gefunden:
         strColor = "dies entspricht der Word-Farbe: " & strColor
        End If
```

```
      strMsgTabVg = vbCrLf _
        & "Die Tabellenzelle hat eine Vordergrundschattierung (Theme Color);" _
        & vbCrLf _
        & strColor & vbCrLf _
        & Fkt_RGB_HSL_Farbanteile(Selection.Cells.Shading.ForegroundPatternColor)
    ' Zwischendatei schließen:
      Dok2.Close (wdDoNotSaveChanges)
"       On Error GoTo Zeile_Fehler1
    ' In das ursprüngliche Dokument zurückkehren:
      Dok1.Activate
    ' **********
     Else
      ' farbige Zellen-Vordergrunds-Schattierung (nur bei Textur sichtbar):
      strColor = Fkt_Farbname_aus_Farbcode(Rng0.Cells.Shading.ForegroundPatternColor)
      If strColor = "." Then
        strColor = "Farbindex (ColorIndex = " _
        & Rng0.Cells.Shading.ForegroundPatternColorIndex & " bzw. " _
        & Fkt_Farbname_FarbIndex(Rng0.Cells.Shading.ForegroundPatternColorIndex)
       Else
       ' Word-Farbe gefunden:
       strColor = "dies entspricht der Word-Farbe: " & strColor
      End If
      strMsgTabVg = vbCrLf _
        & "Die Tabellenzelle hat die Vordergrundschattierung: " _
        & Selection.Cells.Shading.ForegroundPatternColor & ";" & vbCrLf _
        & strColor & vbCrLf _
        & Fkt_RGB_HSL_Farbanteile(Rng0.Cells.Shading.ForegroundPatternColor)
     End If
     ' ****************************************
    ' Textur in Tabellen:
      If Selection.Cells.Shading.Texture = 9999999 _
      Or Selection.Cells.Shading.Texture = 0 Then
        If strMsgTabVg <> "" Then
          strMsgTabVg = vbCrLf & strMsgTabVg _
          & " Die Vordergrundschattierung ist mangels Textur nicht sichtbar."
        ''''''''
        ' Eventuell "strMsgTabV" im Protokoll dann weglassen?
        '' strMsgTabVg = ""
        ''''''''
        End If
      Else
        ' Die Tabellenzelle enthält Textur.
        strMsgTabVg = vbCrLf & vbCrLf & "Die Tabellenzelle enthält Textur, Code = " _
        & Selection.Cells.Shading.Texture & "." & strMsgTabVg
        '
        If lngGrauwertHtg_Resultierend = -1 Then lngGrauwertHtg_Resultierend = -2
      End If
    ' ***************************************************************
    ' Auf Hintergrunds-Schattierungen von Tabellenzellen prüfen (strMsgTabHg).
    ' ***************************************************************
    If Selection.Range.Cells(1).Range.Cells.Shading.BackgroundPatternColor = -1 Then
      If Err.Number > 0 Then
        Debug.Print Err.Description
        Err.Clear
        strMsgTabHg = vbCrLf & _
        "Es ist das Steuerzeichen für Tabellenzeilen-Ende markiert!"
      Else
       ' 23. Mai 2017
       ' Fehler bei verbundenen Tabellenzellen mit unterschiedlicher Schattierung:
        strMsgTabHg = vbCrLf _
        & "Die Tabellenzelle hat eine unbestimmbare Hintergrundschattierung (""-1""), " _
        & "vermutlich wegen verbundener Zellen." ' ???
```

```
        End If
        If lngGrauwertHtg_Resultierend = -1 Then lngGrauwertHtg_Resultierend = -3
      ElseIf Selection.Cells.Shading.BackgroundPatternColor = -16777216 Then
        ' keine Hintergrundschattierung der Tabellenzelle.
        strMsgTabHg = vbCrLf & vbCrLf & "Die Tabellenzelle hat keine Hintergrundschattierung."
      '  **********
      ElseIf Selection.Cells.Shading.BackgroundPatternColor < -1 Then
        ' farbige Zellenschattierung mit negativem Zahlenwert:
        ' "Theme colour" in docx-Dateien (oder ungespeichert in sonstigen Dateien),
        ' keine absolute Farbe!
        ' Umweg über Zwischendatei, um aus Theme-Color eine RGB-Farbe zu machen:
 Selection.Tables(1).Range.Copy
 Dim LngK As Long
 LngK = Rng0.Start  "Selection.Range.Start
 LngK = LngK - Selection.Tables(1).Range.Start
''''''''''''''''''''''
    "   rng0.Cells(1).Range.Copy
        Set Dok1 = ActiveDocument
        Set Dok2 = Documents.Add(DocumentType:=wdNewBlankDocument, Visible:=False)
       On Error GoTo Zeile_Fehler2
        Dok2.Activate
        ' Falls in der Normal.dot schon ein Briefkopf vorgesehen ist, diesen löschen:
        Selection.WholeStory
        Selection.Delete
        Selection.Paste
        "rng0.Select
''''''''''''''''''''''
Dim strFullname As String
  Dok2.SaveAs2 FileName:=Dok1.Path _
    & "\zdok2.doc", _
    FileFormat:=wdFormatDocument, LockComments:=False, _
    Password:="", AddToRecentFiles:=True, WritePassword:="", _
    ReadOnlyRecommended:=False, EmbedTrueTypeFonts:=False, _
    SaveNativePictureFormat:=False, SaveFormsData:=False, _
    SaveAsAOCELetter:=False, CompatibilityMode:=0
'  Datei kurz schließen und öffnen, um "Theme-Colors" in RGB-Farben umzuwandeln:
  strFullname = Dok2.FullName
  Dok2.Close '(wdDoNotSaveChanges)
  Dim tt As Single
    tt = Timer
    Do While Timer < tt + 0.1  'Sekunden
      DoEvents
    Loop
  Application.Documents.Open (strFullname) ' wieder öffnen.
  Set Dok2 = ActiveDocument
  Dok2.Range.Tables(1).Select
  LngK = LngK + Selection.Tables(1).Range.Start
  Selection.SetRange Start:=LngK, End:=LngK + 1
  "Set rng0 = Selection.Range
        ' **********
        If Selection.Cells(1).Shading.BackgroundPatternColor = -16777216 Then
          '  Hintergrundschattierung = "automatisch", kann aber schattiert sein.
          strMsgTabHg = vbCrLf & vbCrLf _
            & "Die Tabellenzelle hat als Hintergrundschattierung eine ""Theme colour"";" _
            & vbCrLf _
            & "Zellenschattierung = automatisch, möglicherweise " _
            & "existiert aber eine Tabellenschattierung, was den Kontrast vermindert."
          If lngGrauwertHtg_Resultierend = -1 Then lngGrauwertHtg_Resultierend = -3
        Else
          ' jetzt eine normale RGB-Farbe:
          strColor = Fkt_Farbname_aus_Farbcode(Selection.Cells(1).Shading.BackgroundPatternColor)
          If strColor = "." Then
```

```
            strColor = "Farbindex (ColorIndex) = " _
            & Selection.Cells(1).Shading.BackgroundPatternColorIndex & " bzw. " _
            & Fkt_Farbname_FarbIndex(Selection.Cells(1).Shading.BackgroundPatternColorIndex)
            ' ****
            ' Statt den ColorIndex besser den Farbkreisbereich angeben:
            strColor = strColor & vbCrLf _
            & Fkt_Farbkreisbereich(Selection.Cells(1).Shading.BackgroundPatternColor)
            ' ****
          Else
          ' Word-Farbe gefunden:
          strColor = "dies entspricht der Word-Farbe: " & strColor
          End If
          strMsgTabHg = vbCrLf & vbCrLf _
            & "Die Tabellenzelle hat als Hintergrundschattierung eine ""Theme colour"";" _
            & vbCrLf _
            & strColor & vbCrLf _
            & Fkt_RGB_HSL_Farbanteile(Selection.Cells(1).Shading.BackgroundPatternColor)
          If lngGrauwertHtg_Resultierend = -1 Then lngGrauwertHtg_Resultierend = lngGrauwert255
        End If
      ' **********
      ' Zwischendatei schließen und löschen::
        Dok2.Close (wdDoNotSaveChanges)
        Kill strFullname
        On Error GoTo Zeile_Fehler1
      ' In das ursprüngliche Dokument zurückkehren:
        Dok1.Activate
      ' **********
      Else
        ' farbige Zellenschattierung:
        strColor = Fkt_Farbname_aus_Farbcode(Selection.Cells.Shading.BackgroundPatternColor)
        If strColor = "." Then
          strColor = "Farbindex (ColorIndex) = " _
          & Selection.Cells.Shading.BackgroundPatternColorIndex & " bzw. " _
          & Fkt_Farbname_FarbIndex(Selection.Cells.Shading.BackgroundPatternColorIndex)
          ' *****
          ' Statt den ColorIndex besser den Farbkreisbereich angeben:
          strColor = strColor & vbCrLf _
          & Fkt_Farbkreisbereich(Selection.Cells.Shading.BackgroundPatternColor)
          ' ******
        Else
        ' Word-Farbe gefunden:
        strColor = "dies entspricht der Word-Farbe: " & strColor
        End If
        strMsgTabHg = "Die Tabellenzelle hat die Hintergrundschattierung: " _
          & Selection.Cells.Shading.BackgroundPatternColor & ";" & vbCrLf _
          & strColor & vbCrLf _
          & Fkt_RGB_HSL_Farbanteile(Selection.Cells.Shading.BackgroundPatternColor)
        "& Fkt_RGB_HSL_Farbanteile(Selection.Range.Cells.Shading.BackgroundPatternColor)
        If lngGrauwertHtg_Resultierend = -1 Then lngGrauwertHtg_Resultierend = lngGrauwert255
      End If
      ' ****************************************
      ' Markierung zurückstellen:
      Rng0.Select
    Else
      ' keine Tabelle:
      strMsgTabHg = ""
      strMsgTabVg = ""
    End If
  ' ********************************************************************
  ' Kontrast berechnen (strMsgKontrast)
  ' ********************************************************************
    If lngGrauwertHtg_Resultierend = -1 Then
```

```
    ' Keine Hintergrundschattierung vorhanden:
    lngGrauwertHtg_Resultierend = 255
  End If
  ' Sonderfälle ausschließen:
  If lngGrauwertHtg_Resultierend = -2 Then
    strMsgKontrast = vbCrLf & vbCrLf _
    & "Das Zeichen ist wegen Schraffur möglicherweise schwer lesbar!"
  ElseIf lngGrauwertHtg_Resultierend = -3 Then
    strMsgKontrast = vbCrLf & vbCrLf _
    & "Der Kontrast ließ sich nicht bestimmen."
  Else
    ' Mindestkontrast 125 von 255 nach https://www.w3.org/TR/AERT prüfen.
    If Abs(lngGrauwertHtg_Resultierend - lngGrauwertText) < 125 Then
      strMsgKontrast = vbCrLf & vbCrLf _
      & "Der Helligkeitsunterschied zwischen Text und Hintergrund ist: " & vbCrLf _
      & Abs(lngGrauwertHtg_Resultierend - lngGrauwertText) & " von 255 bzw. " _
      & Round(Abs(lngGrauwertHtg_Resultierend - lngGrauwertText) / 2.55, 1) & " %." _
      & vbCrLf _
      & "Dies erfüllt nicht die Anforderung an den Mindest-Kontrast (125 von 255)!"
    Else
      strMsgKontrast = vbCrLf & vbCrLf _
      & "Der Helligkeitsunterschied zwischen Text und Hintergrund ist ausreichend: " _
      & vbCrLf _
      & Abs(lngGrauwertHtg_Resultierend - lngGrauwertText) & " von 255 bzw. " _
      & Round(Abs(lngGrauwertHtg_Resultierend - lngGrauwertText) / 2.55, 1) & " %."
    End If
  End If
' Bei Steuerzeichen keinen Kontrast ausgeben:
  If strMsgSteuerzeichen <> "" Then
    strMsgKontrast = ""
  End If
' ******************************************************************
' Zusammenfassung und Ergebnis-Ausgabe:
' ******************************************************************
' Bildschirmanzeige wieder freigeben:
  Application.ScreenUpdating = True
' Das Ergebnis zusammenstellen und ausgeben:
  StrErgebnis = strMsgZeichen _
      & strMsgSteuerzeichen _
      & vbCrLf & vbCrLf _
      & strMsg1Hg _
      & vbCrLf & vbCrLf _
      & strMsg1Vg _
      & strMsgAbsHg _
      & vbCrLf & vbCrLf _
      & strMsgTabHg & strMsgTabVg _
      & strMsgKontrast
' Überschüssige Absatzmarken beseitigen:
  StrErgebnis = Replace(StrErgebnis, vbCrLf & vbCrLf & vbCrLf, vbCrLf & vbCrLf)
  StrErgebnis = Replace(StrErgebnis, vbCrLf & vbCrLf & vbCrLf, vbCrLf & vbCrLf)
  StrErgebnis = Replace(StrErgebnis, vbCrLf & vbCrLf & vbCrLf, vbCrLf & vbCrLf)
  Debug.Print StrErgebnis
' Ergebnis als erweiterte Messagebox (es sind zuviele Zeilen für eine normale MsgBox):
  StrErgebnis = MessageBoxW(0, StrPtr(StrErgebnis), StrPtr("Farbanalyse und Kontrastprüfung"), 0)
' ******************************************************************
' Am Schluss des Makros wieder aufräumen und unsichtbare Zwischendateien sichtbar machen:
  Application.Visible = True
  WordBasic.DisableAutoMacros 0
  '' Set Dok1 = Nothing
  '' Set Dok2 = Nothing
Exit Sub
' ******************************************************************
```

```
Zeile_Fehler1:
  Application.Visible = True
  WordBasic.DisableAutoMacros 0
  MsgBox "Fehler in Sub Fehlerbehandlung" & vbCrLf & "Fehlernummer: " & Err.Number & _
      vbCrLf & "Fehlerbeschreibung: " & Err.Description, vbCritical
Exit Sub
' ********************************************************************
Zeile_Fehler2:
  Application.Visible = True
  WordBasic.DisableAutoMacros 0
On Error Resume Next
  Dok2.Close (wdDoNotSaveChanges)
  Dok1.Activate
Resume
  MsgBox "Fehler in Sub Fehlerbehandlung" & vbCrLf & "Fehlernummer: " & Err.Number & _
      vbCrLf & "Fehlerbeschreibung: " & Err.Description, vbCritical
  ''  Set Dok1 = Nothing
  ''  Set Dok2 = Nothing
End Sub
' ---------------------------------------------------------------------------------------------------------
```

```
Private Function Fkt_Farbname_aus_Farbcode(lngColor As Long) As String
'  H. Chris Gast, Februar 2015, Februar 2017, 1. + 14. + 16. März 2017, Stand 9. Mai 2017.
'  Diese Funktion gibt aus der Nummer den Farbnamen zurück, sofern vorhanden.
  Dim strColor As String
'  ***************
  Select Case lngColor
    Case -16777216 ' Automatic color. Default; Bei Zeichenfarb schwarz, bei Hintergrund weiß.
        strColor = "wdColorAutomatic"
    Case 0  ' Black color.
        strColor = "wdColorBlack (schwarz)"
    ' ***********
    ' Word-Standardfarben bei Word 2010:
    Case 192
        strColor = "Standardfarbe Dunkelrot"
    "Case 255
    "     strColor = "Rot"
    Case 49407
        strColor = "Standardfarbe Orange"
    "Case 65535
    "     strColor = "Gelb"
    Case 5296274
        strColor = "Standardfarbe Hellgrün"
    Case 5287936
        strColor = "Standardfarbe Grün"
    Case 15773696
        strColor = "Standardfarbe Hellblau"
    Case 12611584
        strColor = "Standardfarbe Blau"
    Case 6299648
        strColor = "Standardfarbe Dunkelblau"
    Case 10498160
        strColor = "Standardfarbe Lila"
```

```
' ***********
' benannte Word-Farben:
Case 13421619  '  "wdColorAqua"
    strColor = "wdColorAqua (helles Cyan)"
Case 16711680  ' Blue color.
    strColor = "wdColorBlue (blau)"
Case 10053222  '  Blue-gray color.
    strColor = "wdColorBlueGray (blaugrau)"
Case 65280     '  Bright green color.
    strColor = "wdColorBrightGreen (hellgrün)"
Case 13209     '  Brown color.
    strColor = "wdColorBrown (braun)"
Case 8388608   '  Dark blue color.
    strColor = "wdColorDarkBlue (dunkelblau)"
Case 13056     '  Dark green color.
    strColor = "wdColorDarkGreen (sehr dunkles grün)"
Case 128       '  Dark red color.
    strColor = "wdColorDarkRed (dunkelrot)"
Case 6697728   ' Dark teal color.
    strColor = "wdColorDarkTeal (dunkles blaugrün)"
Case 32896     ' Dark yellow color.
    strColor = "wdColorDarkYellow (dunkelgelb)"
Case 52479     '  Gold color.
    strColor = "wdColorGold (gold)"
Case 15987699  '  Shade 05 of gray color.
    strColor = "wdColorGray05"
Case 15132390  '  Shade 10 of gray color.
    strColor = "wdColorGray10"
Case 14737632  '  Shade 125 of gray color.
    strColor = "wdColorGray125"
Case 14277081  '  Shade 15 of gray color.
    strColor = "wdColorGray15"
Case 13421772  '   Shade 20 of gray color.
    strColor = "wdColorGray20"
Case 12632256   ' Shade 25 of gray color.
    strColor = "wdColorGray25"
Case 11776947  '  Shade 30 of gray color.
    strColor = "wdColorGray30"
Case 10921638  '   Shade 35 of gray color.
    strColor = "wdColorGray35"
Case 10526880  '   Shade 375 of gray color.
    strColor = "wdColorGray375"
Case 10066329  '  Shade 40 of gray color.
    strColor = "wdColorGray40"
Case 9211020   '  Shade 45 of gray color.
    strColor = "wdColorGray45"
Case 8421504   '  Shade 50 of gray color.
    strColor = "wdColorGray50"
Case 7566195   '  Shade 55 of gray color.
    strColor = "wdColorGray55"
Case 6710886   '  Shade 60 of gray color.
    strColor = "wdColorGray60"
Case 6316128   '  Shade 625 of gray color.
    strColor = "wdColorGray625"
Case 5855577   '   Shade 65 of gray color.
    strColor = "wdColorGray65"
Case 5000268   '  Shade 70 of gray color.
    strColor = "wdColorGray70"
Case 4210752   '   Shade 75 of gray color.
    strColor = "wdColorGray75"
Case 3355443   '  Shade 80 of gray color.
    strColor = "wdColorGray80"
```

```
  Case 2500134   '  Shade 85 of gray color.
      strColor = "wdColorGray85"
  Case 2105376   '   Shade 875 of gray color.
      strColor = "wdColorGray875"
  Case 1644825   '  Shade 90 of gray color.
      strColor = "wdColorGray90"
  Case 789516 ' Shade 95 of gray color.
      strColor = "wdColorGray95"
  Case 32768      '  Green color.
      strColor = "wdColorGreen (grün)"
  Case 10040115   '  Indigo color.
      strColor = "wdColorIndigo (indigoblau)"
  Case 16751052   '  Lavender color.
      strColor = "wdColorLavender (fliederfarben)"
  Case 16737843   '  Light blue color.
      strColor = "wdColorLightBlue  (hellblau)"
  Case 13434828   '  Light green color.
      strColor = "wdColorLightGreen (sehr helles grün)"
  Case 39423  '  Light orange color.
      strColor = "wdColorLightOrange (helles orange)"
  Case 16777164  ' Helles Cyan
      strColor = "wdColorLightTurquoise (helles Cyan)"
  Case 10092543  '  Light yellow color.
      strColor = "wdColorLightYellow (hellgelb)"
  Case 52377      '  Lime color.
      strColor = "wdColorLime (gelbgrün)"
  Case 13107      '  Olive green color.
      strColor = "wdColorOliveGreen (olivgrün)"
  Case 26367      '  Orange color.
      strColor = "wdColorOrange (orange)"
  Case 16764057  '  Pale blue color.
      strColor = "wdColorPaleBlue (blassblau)"
  Case 16711935  '  Pink color.
      strColor = "wdColorPink (Magenta)"
  Case 6697881   '  Plum color.
      strColor = "wdColorPlum (pflaumenfarben)"
  Case 255        '  Red color.
      strColor = "wdColorRed (rot)"
  Case 13408767  '  Rose color.
      strColor = "wdColorRose (helles rosa)"
  Case 6723891   '  Sea green color.
      strColor = "wdColorSeaGreen (Seegrün)"
  Case 16763904  '  Sky blue color.
      strColor = "wdColorSkyBlue (himmelblau)"
  Case 10079487  '  Tan color.
      strColor = "wdColorTan (hellbraun)"
  Case 8421376   '  Teal color.
      strColor = "wdColorTeal (blaugrün)"
  Case 16776960  ' Turquoise color.
      strColor = "wdColorTurquoise (cyanblau)"
  Case 8388736   '  Violet color.
      strColor = "wdColorViolet (violett)"
  Case 16777215, -603914241    '   Weiße Farbe (2 Codes möglich).
      strColor = "wdColorWhite (weiß)"
  Case 65535    '   Standardfarbe gelb.
      strColor = "wdColorYellow (gelb)"
  Case Else
     "strColor = "Keine der Word-Farbnamen"
     strColor = ""
End Select
' ***
Fkt_Farbname_aus_Farbcode = strColor & "."
```

```
End Function


Private Function Fkt_Farbname_FarbIndex(IntFarbe As Integer)
' H. Chris Gast, August 2012, Januar 2015, Februar 2017, 16. + 20. März 2017. 3. Mai 2017.
' Dies Makro gibt die Markierfarbe (ColorIndex) an der Cursor-Stelle zweisprachig zurück.
' Dies funktioniert auch bei Steuerzeichen,
' wo die Farbe im Druck und/oder auf dem Bildschirm nicht sichtbar ist.
' Einige der deutschen Farbnamen habe ich abweichend gewählt
' entsprechend der Bezeichnung im Farbenkreis (März 2017).
' Seltsamerweise gibt es in Word 2010 die Index-Nummern 17 bis 24
' für Tabellenschattierungen bei einigen Web-Farben (entdeckt 2. Mai 2017).
Dim strFarbe As String
  '
  Select Case IntFarbe
    Case 0
      strFarbe = " wdAuto oder wdNoHighlight (keine Farbe)"
    Case 1
      strFarbe = " wdBlack (Schwarz)"
    Case 2
      strFarbe = " wdBlue (Blau)"
    Case 3
      strFarbe = " wdTurquoise (Cyan/Hellblau)"
    Case 4
      strFarbe = " wdBrightGreen (Hellgrün)"
    Case 5
      strFarbe = " wdPink (Magenta)"
    Case 6
      strFarbe = " wdRed (Rot)"
    Case 7
      strFarbe = " wdYellow (Gelb)"
    Case 8
      strFarbe = " wdWhite (Weiß)"
    Case 9
      strFarbe = " wdDarkBlue (Dunkelblau)"
    Case 10
      strFarbe = " wdTeal (Blaugrün)"
    Case 11
      strFarbe = " wdGreen (Grün)"
    Case 12
      strFarbe = " wdViolet (Violett)"
    Case 13
      strFarbe = " wdDarkRed (Dunkelrot)"
    Case 14
      strFarbe = " wdDarkYellow (Dunkelgelb)"
    Case 15
      strFarbe = " wdGray50 (Grau-50)"
    Case 16
      strFarbe = " wdGray25 (Grau-25)"
    ' ***
    ' Nur als Schattierfarben für Tabellenzellen möglich:
    Case 19
      strFarbe = " DarkBlue (Dunkelblau)"
    Case 20
      strFarbe = " DarkCyan (Dunkel-Cyan)"
    Case 21
      strFarbe = " DarkGreen (Dunkelgrün)"
    Case 22
      strFarbe = " DarkRed  (Dunkelrot)"
    Case 23
      strFarbe = " DarkGray (Dunkelgrau)"
```

```
    Case 24
      strFarbe = " LightGray (Hellgrau)"
    ' ***
    Case Else
      strFarbe = "(unbekannte oder mehrere Farbmarkierungen)"
  End Select
  Fkt_Farbname_FarbIndex = strFarbe & "."
End Function


Private Function Fkt_Farbkreisbereich(lngColor As Long) As String
' H. Chris Gast, 17. + 18. Mai 2017.
' Die Farbwinkelgrenzen sind hier zum Teil (Cyan und Pink) subjektiv.
' Der Anwender des Makros kann sie gerne für sich verändern.
' Vielleicht sollten mehr Farben als Grau-ähnlich eingestuft werden.
' *************************
  IntFarbwinkel = Fkt_Farbwinkel(lngColor)
  ' mit "Fkt_Farbwinkel" wird auch der Grauwert bestimmt.
  ' Ganz helle Farben als Weiß ansehen:
  If lngGrauwert255 > 243 Then
    Fkt_Farbkreisbereich = "(Weiß oder fast weiß)."
  Else
    ' Hintergrundschattierung  nach Farbwinkel unterscheiden:
    Select Case IntFarbwinkel
      Case Is > 700
        ' keine Farbe:
        Select Case lngGrauwert255
          Case Is > 230
            Fkt_Farbkreisbereich = "(Grau heller als Grau 10)."
          Case Is > 192
            Fkt_Farbkreisbereich = "(Grau heller als Grau 25)."
          Case Is > 153
            Fkt_Farbkreisbereich = "(Grau heller als Grau 40)."
          Case Is > 128
            Fkt_Farbkreisbereich = "(Grau heller als Grau 50)."
          Case Is > 76
            Fkt_Farbkreisbereich = "(Grau heller als Grau 70)."
          Case Is > 51
            Fkt_Farbkreisbereich = "(Grau heller als Grau 80)."
          Case Is > 25
            Fkt_Farbkreisbereich = "(Grau heller als Grau 90)."
          Case Is > 12
            Fkt_Farbkreisbereich = "(Grau heller als Grau 95)."
          Case Else
            Fkt_Farbkreisbereich = "(Schwarz oder fast schwarz)."
        End Select
        ' *****
      Case Is >= 340  ' Karmesinrot oder Pink
        If lngGrauwert255 > 100 Then
          ' pink
          Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Pink/Rosa"")."
        Else
          ' karmesinrot
          Fkt_Farbkreisbereich = "(Karmesinrot)."
        End If
      Case Is > 325   ' pink (rosa)
        Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Pink/Rosa"")."
      Case Is > 280   ' Magenta
        Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Magenta"")."
      Case Is > 250   ' violett
        Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Violett/Lila"")."
```

```
        Case Is >= 230  ' blau
          Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Blau"")"
        Case Is >= 190  ' Azurblau
          Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Azurblau")."
        Case Is >= 180  ' cyanblau
          Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Cyan")."
        Case Is > 158   ' türkisgrün
          Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Blaugrün/Türkis")."
        Case Is >= 110  ' grün
          Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Grün")."
        Case Is >= 75   ' gelbliches Grün
          Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Gelbgrün")."
        Case Is > 40    ' gelb, einschließlich Gold
          Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Gelb")."
        Case Is >= 19   ' Orange (auch Sienna-Braun)
          Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Orange")."
        Case Else       ' rot (auch rotbraun)
          Fkt_Farbkreisbereich = "(Farbkreis-Segment ""Rot")."
      End Select
    End If
End Function
' ---------------------------------------------------------------------------------------------------------------


Private Function Fkt_RGB_HSL_Farbanteile(lngColor As Long) As String
' H. Chris Gast, 2015, neu Februar 2017, 12. bis 20. März 2017, 16. Mai 2017.
' Funktion, um RGB-Farbwerte, HSL-Farbwerte (auch HSV bzw. HSB) und Grauwerte zu bestimmen.
' Außer als Text werden Grauwert und Farbwinkel als Zahlenwerte ausgegeben.
'
' RGB-Anteile aus http://www.office-loesung.de/ftopic421244_0_0_asc.php,
' HSL-Formel aus  http://www.vb-helper.com/howto_rgb_to_hls.html
' und https://en.wikipedia.org/wiki/HSL_and_HSV (März 2017),
' Formel für "Color-Brightness" in https://www.w3.org/TR/AERT.html
' entspricht "Grauwert" in https://de.wikipedia.org/wiki/Grauwert.
Dim lngRed As Long
Dim lngGreen As Long
Dim lngBlue As Long
"Dim lngGrauwert255 As Long ' (ist am Beginn des Makromoduls definiert)
Dim sgGrauwert100 As Single
'  ******
'  Farbanteile:
  lngRed = lngColor And vbRed
  lngGreen = (lngColor And vbGreen) \ &H100
  lngBlue = (lngColor And vbBlue) \ &H10000
  '' Debug.Print lngRed & ", " & lngGreen & ", " & lngBlue
'  Grauwert (Skala 0 bis 255):
  lngGrauwert255 = (0.299 * lngRed + 0.587 * lngGreen + 0.114 * lngBlue)
'  Grauwert in Prozent:
  sgGrauwert100 = 100 * Round(lngGrauwert255 / 255, 3)
'  ***************************
' HSL-Makro etwas verändert, weil das Original nicht bei mir funktionierte.
' Weitere Änderung: Für Schwarz/Grau/Weiß wird statt "0" die (unmögliche)Zahl "-1"
' gesetzt. Die aufrufende Prozedur muss dann aus der "-1" einen Strich machen.
' So kann dann "rot" von weiß/grau/schwarz unterschieden werden.
' Single =Dezimalzahl von -3,402823*10 hoch -38 bis  +3,402823*10 hoch 38;    4 Bytes.
' Double = Dezimalzahl von -1,797693*10 hoch -324 bis +1,797693*10 hoch 324;  8 Bytes.
Dim R As Single
Dim G As Single
Dim B As Single
  R = lngRed / 255
  G = lngGreen / 255
  B = lngBlue / 255
```

```
Dim max As Single ' größter RGB-Wert
Dim min As Single ' kleinster RGB-Wert
Dim H As Single ' Winkel auf dem Farbenkreis (Hue)
Dim S As Single ' Sättigung
Dim L As Single ' Helligkeit
Dim strH As String
'  Größte und kleinste RGB-Komponente bestimmen:
  max = R
  If max < G Then max = G
  If max < B Then max = B
  '
  min = R
  If min > G Then min = G
  If min > B Then min = B
  '
'  Farb-Helligkeit ("Lightness") bei HSL:
  L = (max + min) / 2
  '
'  Fallunterscheidung, schwarzweiß oder Farbe:
  If Abs(max - min) < 0.0001 Then
    ' schwarz, weiß oder grau.
    S = 0  ' Sättigung "saturation"
    H = -1 ' Farbwinkel "Hue" ist undefiniert (H = "0")
  Else
    ' farbig:
    If L <= 0.5 Then
      S = (max - min) / (max + min)
    Else
      S = (max - min) / (2 - max - min)
    End If
    ' Fallunterscheidung:
    If R = max Then
      H = 60 * (G - B) / (max - min)
    ElseIf G = max Then
      H = 60 * (2 + (B - R) / (max - min))
    ElseIf B = max Then
      H = 60 * (4 + (R - G) / (max - min))
    End If
    If H < 0 Then H = H + 360
  End If
'  Bei weiß/grau/schwarz gibt es keinen Winkel:
  If H < 0 Then
    strH = " - "
    IntFarbwinkel = 1000
  Else
    strH = Format(H, "0.0") & "°"
    IntFarbwinkel = Int(H)
  End If
'  ***************************
'  Beim HSV-System (HSB) ist nur der Winkel H identisch;
  Dim Shsv As Single  ' Saturation bei HSV
  Dim Vhsv As Single  ' Value (Brightness) bei HSV bzw. HSB
  If max = 0 Then Shsv = 0 Else Shsv = (max - min) / max
  Vhsv = max
'  ***************************
'  Ergebnis:
  Fkt_RGB_HSL_Farbanteile = _
    "RGB-Werte: Rotanteil = " & lngRed & ", Grünanteil = " _
    & lngGreen & ", Blauanteil = " & lngBlue & "," & vbCrLf _
    & "bzw. hexadezimal #" & " " & Right("0" & Hex(lngRed), 2) & " " _
    & Right("0" & Hex(lngGreen), 2) & " " & Right("0" & Hex(lngBlue), 2) & "." & vbCrLf _
    & "Als HSL-Koordinaten sind es: " & vbCrLf & "Farbkreiswinkel Hue = " & strH _
```

```
    & "; Sättigung S = " & Format(S, "0.00") _
    & "; Intensität L = " & Format(L, "0.00") & "." _
    & vbCrLf & "Das HSV-System (HSB) ergibt: " _
    & strH & "; " & Format(Shsv, "0.00") & ";  " & Format(Vhsv, "0.00") & "." _
    & vbCrLf _
    & Fkt_RGB_CMYK(lngRed, lngGreen, lngBlue) & vbCrLf _
    & "Der berechnete Grauwert (Helligkeit) ist " & lngGrauwert255 _
    & " von 255,  bzw. " & sgGrauwert100 & " %."
End Function
' ----------------------------------------------------------------------------------------------------------------
```

```
Private Function Fkt_RGB_CMYK(lngRed As Long, lngGreen As Long, lngBlue As Long) _
As String
'  H. Chris Gast, 22. März 2017.
'  Berechnung der CMYK-Farbanteile aus den RGB-Farbwerten.
'  Das Farbsystem RGB (Rot-Grün-Blau) ist eine additive Farbmischung aus
'  drei Farben, etwa für Bildschirme.
'  Das Farbsystem CMYK ist dagegen eine subtraktive Farbmischung aus vier Farben,
'  etwa für Drucker (C =Cyanblau, M = Magenta, y = Gelb und K = Schwarz).
'
'  Quellen (Abruf 22. März 2017):
'  https://fr.wikipedia.org/wiki/Quadrichromie
'  https://ru.wikipedia.org/wiki/CMYK
'  http://www.easyrgb.com/index.php?X=MATH&H=11#text11
'
'  Es gilt für die folgende Umrechnung von RGB in CMYK:
'  RGB-Werte von 0  bis 255,
'  CMYK-Werte (und CMY-Werte) von 0 bis 1.
Dim C As Single ' Cyan
Dim M As Single ' Magenta
Dim Y As Single ' Yellow (gelb)
Dim k As Single ' Key/Black (schwarz)
  C = 1 - (lngRed / 255)
  M = 1 - (lngGreen / 255)
  Y = 1 - (lngBlue / 255)
' Bei realen Farben gibt es mit drei Farben kein befriedigendes Schwarz.
' Abhilfe: Schwarze Farbe als 4. Druckerfarbe.
' Setze Schwarz als den kleinsten der drei Farbwerte:
  If C < M Then
    k = C
  Else
    k = M
  End If
  If Y < k Then k = Y
' Die drei bunten Farben dann um den Wert des Schwarz ("K") abschwächen:
   '" Lösung aus Russischer Wikipedia(und aus Französisch, Algorithme no 1):
   "  If K > 0 Then
   "     C = C - K
   "     M = M - K
   "     Y = Y - K
   "  End If
'  *************
' Lösung aus Französischer Wikipedia (Algorithme no 2) und Easy-RGB:
If k = 1 Then  ' //Black
   C = 0
   M = 0
   Y = 0
Else
   C = (C - k) / (1 - k)
```

```
  M = (M - k) / (1 - k)
  Y = (Y - k) / (1 - k)
End If
' ******************************
  Fkt_RGB_CMYK = "Die CMYK-Werte sind " _
  & Format(C, "0.00") & "; " _
  & Format(M, "0.00") & "; " & Format(Y, "0.00") & "; " & Format(k, "0.00") & "."
End Function
' ----------------------------------------------------------------------------------------------------
```

15.3 Farben durch Grautöne ersetzen (mit Mindestkontrast)

```
Sub Farben_durch_Grautöne_ersetzen_alle_StoryRanges()
'  H. Chris Gast, Januar 2015, Januar 2016, Februar + März 2017, April 2017, Stand 4. Mai 2017.
'  Alle Farben im Dokument werden durch Grautöne ersetzt, um Druckkosten zu sparen.
'
'  Farben von Text und Hintergrund werden durch die entsprechende Grautöne ersetzt,
'  aber mit Mindestkontrast
'  (Helligkeitsdifferenz > 125 von 255 nach https://www.w3.org/TR/AERT).
'
'  Absatzschattierungen werden durch einen festen hellen Grauwert (Grau 20) ersetzt.
'
'  Tabellenfeld-Schattierungen werden durch zwei Grauschattierungen ersetzt,
'  rötliche Farben etwas dunkler, sonstige Farben etwas heller (Unterprozedur).
'
'  Text in schattierten Tabellenzellen und bei Absatzschattierungen wird
'  in schwarze Buchstaben gesetzt, um den Kontrast zu erhöhen.
'
'  Schraffuren werden bei Text durch weißen Hintergrund überdeckt.
'
'  Vorteile dieses Makros:
'  "Theme-Colors" werden berücksichtig!
'  Auch Kopf- und Fußzeilen werden erfasst.
'  Die Originaldatei wird nur gespeichert, nicht verändert,
'  Die Umwandlung in Grautöne erfolgt in einer neu-erzeugten Word-Datei für den Druck.
'
'  Mankos:
'  Bei Bildern gibt es leider keinen Mindestkontrast (Unterprozedur).
'  Das Makro funktioniert nicht für die Farben der Änderungs-Kennung!
'
'  Das Makro ist nur als Muster gedacht und kann bedarfsweise abgewandelt werden.
'  Es sollte vielleicht besser für den Druck gänzlich auf Grautöne verzichtet werden:
'  es könnten etwa Grüntöne durch "kursiv", Text-Rottöne durch "Kapitälchen"
'  und Blautöne durch "doppelte Unterstreichung" ersetzt werden.
'
'  Rechenzeit für 6 total farbige Seiten am  10. April 2017 zuhause:  33,4 Sekunden.
'  Das Makro ist zeitraubend.
'  ***********************************************************
'  Zeitmessen für die Makro-Dauer in Sekunden
   Dim Zeit1 As Single    '  "Single" = Zahlen mit Dezimalen
   Dim Zeit2 As Single
   Dim Zeitdauer As Single
   Zeit1 = Timer
'  ************************
Dim RngStory As Range
Dim rngHeaderFooter As HeaderFooter
Dim shpShape As Shape        ' Textfelder (und anderes)
Dim BL_Textfeld As Boolean   ' Suche nach Textfeldern in Kopf- oder Fußzeile
Dim rngSection As Section
Dim IntAbfrage As Integer
Dim strFullname As String
'  ***********************************************************
'  Rechenzeit beschleunigen:
   Application.ScreenUpdating = False
'  ***********************************************************
'  Separate Druckdatei erzeugen:
'  ***********************************************************
'  Zuerst vorhandene Datei (unverändert) speichern:
   ActiveDocument.Save
'  Druckdatei im alten Word97-2003-Format erzwingen (Kompatibilitätsmodus),
'  um "Theme-Colors" durch analysierbare herkömmliche Farben zu ersetzen.
'  Gleichzeitig werden OMML-Gleichungen durch Grafiken ersetzt, was Druckprobleme
```

```
' einiger Formelzeichen bei den Kyocera-Druckern umgeht.
  ActiveDocument.SaveAs2 FileName:=ActiveDocument.Path _
    & "\!SW-Druckdatei_" & ActiveDocument.Name & ".doc", _
    FileFormat:=wdFormatDocument, LockComments:=False, _
    Password:="", AddToRecentFiles:=True, WritePassword:="", _
    ReadOnlyRecommended:=False, EmbedTrueTypeFonts:=False, _
    SaveNativePictureFormat:=False, SaveFormsData:=False, _
    SaveAsAOCELetter:=False, CompatibilityMode:=0
' Datei kurz schließen und öffnen, um "Theme-Colors" in RGB-Farben umzuwandeln:
  strFullname = ActiveDocument.FullName
  ActiveDocument.Close '(wdDoNotSaveChanges)
  Application.Documents.Open (strFullname) ' wieder öffnen.
' ********************************************************
  StatusBar = "Bitte warten - das Makro läuft."   ' Funktioniert nicht immer.
' Rechenzeit beschleunigen:
  Application.ScreenUpdating = False
  "Options.Pagination = False
' ********************************************************
' Alle Dokumentkomponenten außer Fuß- und Kopfzeilen:
  For Each RngStory In ActiveDocument.StoryRanges
    If RngStory.StoryType < 6 Then
      ' ohne Kopf- und Fußzeilen, die weiter unten erfasst werden:
      Call Farben_in_Grautoene_umsetzen(RngStory)
    End If
  Next RngStory
' ********************************************************
' Kopf und Fußzeilen (in allen Sections):
  For Each rngSection In ActiveDocument.Sections
    ' Kopfzeilen:
    For Each rngHeaderFooter In rngSection.Headers
      If rngHeaderFooter.LinkToPrevious = False Then
        Call Farben_in_Grautoene_umsetzen(rngHeaderFooter.Range)
        ' *****
        ' Suche nach Textfeldern in Kopf-/Fußzeilen nur ein einziges Mal durchführen!
        ' (sie werden sonst bei jedem rngHeaderFooter gefunden!)
        If BL_Textfeld = False Then
          For Each shpShape In rngHeaderFooter.Shapes
            If shpShape.Type = msoTextBox Then
              Call Farben_in_Grautoene_umsetzen(shpShape.TextFrame.TextRange)
            End If
          Next
          BL_Textfeld = True
        End If
        ' *****
      End If
    Next rngHeaderFooter
    ' ***********
    ' Fußzeilen:
    For Each rngHeaderFooter In rngSection.Footers
      If rngHeaderFooter.LinkToPrevious = False Then
        Call Farben_in_Grautoene_umsetzen(rngHeaderFooter.Range)
      End If
    Next rngHeaderFooter
  Next rngSection
' ******************************************************************
' In Seitenansicht gehen und ggf. das zweite Windows-Fenster schließen
' da im Makro Fuß- und Kopfzeilen markiert worden waren:
  If ActiveWindow.View.SplitSpecial = wdPaneNone Then
    ActiveWindow.ActivePane.View.Type = wdPrintView
  Else
    ActiveWindow.Panes(2).Close
    ActiveWindow.View.Type = wdPrintView
```

```
  End If
'  In die Hauptkomponente und an den Dokument-Anfang gehen:
  ActiveDocument.StoryRanges(1).Select
  Selection.Collapse
  Selection.HomeKey Unit:=wdStory
'  ***********************************************
  ActiveDocument.Save
  Application.ScreenUpdating = True
  "Options.Pagination = True
'  ***********************************************
'  Zeitmessen für die Makro-Dauer, eine Nachkommastelle:
  Zeit2 = Timer
  Zeitdauer = Format((Zeit2 - Zeit1), "##,##0.0")
'  ***********************************************
'  Auswertung:
  MsgBox "Umwandlung in Grautöne ist erledigt. " _
    & vbCrLf _
    & "Die Rechenzeit war " & Zeitdauer & " Sekunden."
End Sub
' ---------------------------------------------------------------------------------------------------------------------


Private Sub Farben_in_Grautoene_umsetzen(rng0 As Range)
' H. Chris Gast, Februar 2017, 3. + 21. März 2017, 10. + 18. April 2017.
' Farben von Text und Hintergrund durch Grautöne ersetzen, aber mit Mindestkontrast.
' Absatzschattierungen durch festen Grauwert ersetzen.
' Tabellenfeld-Schattierungen werden durch zwei Grauschattierungen ersetzt,
' rötliche Farben etwas dunkler, sonstige Farben etwas heller.
' Bei Absatz- und Tabellenzellenschattierungen wird die Schriftfarbe durch schwarz ersetzt.
'
' Bei Bildern leider kein Mindestkontrast.
' ***********************************************************
Dim aChr As Range
Dim rng2 As Range
Dim apara As Paragraph
'  ***********************************************
'  Bilder entfärben:
  '' Eventuell hier noch eine Abfrage einbauen?
  Call Bilder_graufaerben(rng0)
'  ***********************************************
'  Tabellen grau färben
  '(bei Text werden hierbei wegen dem Kontrast weiße Farbmarkierungen erzeugt):
  Call Tabellenzellen_grau_faerben(rng0)
'  ***********************************************
'  Absatzschattierungen:
  For Each apara In rng0.Paragraphs
    '  Vordergrundschattierung des Absatzes ggf. druch hellgrau ersetzen:
    If apara.Range.ParagraphFormat.Shading.ForegroundPatternColorIndex <> 0 _
    And apara.Range.ParagraphFormat.Shading.ForegroundPatternColorIndex <> 8 Then
      apara.Range.ParagraphFormat.Shading.ForegroundPatternColor = wdColorGray70
    End If
    ' Schraffur hinter Text durch weiße Farbmarkierung verdecken:
    ' (wird jetzt in Unterprozedur für Einzelzeichen geprüft)
    ' ***
    ' Absatz-Schattierung ggf. hellgrau (Grau 20) färben, außer bei "weiß":
    If apara.Range.ParagraphFormat.Shading.BackgroundPatternColorIndex <> 0 _
    And apara.Range.ParagraphFormat.Shading.BackgroundPatternColorIndex <> 8 Then
    '  Schattierung des Absatzes vorhanden:
      apara.Range.ParagraphFormat.Shading.BackgroundPatternColor = wdColorGray20
      "apara.Range.ParagraphFormat.Shading.Texture = wdTextureNone
    '  Text schwarz färben:
```

```
      apara.Range.Font.ColorIndex = wdAuto
    End If
  Next apara
'  ************************************************
'  Die einzelnen Buchstaben (ggf. mit Schattierungen) graufärben:
  "On Error Resume Next
  For Each aChr In rng0.Characters
    If Asc(aChr.Text) > 13 Then
    ' Nur wenn Farbe vorhanden, Unterprozedur aufrufen:
      If aChr.Font.ColorIndex <> wdAuto _
      Or aChr.HighlightColorIndex <> wdAuto _
      Or aChr.Font.Shading.BackgroundPatternColorIndex <> wdAuto _
      Or aChr.Font.Shading.ForegroundPatternColorIndex <> wdAuto _
      Or aChr.Paragraphs(1).Range.Shading.Texture <> 0 _
      Or aChr.Font.Shading.Texture <> 0 Then
        ' Prozedur für Einzelzeichen:
        Call Einzelzeichen_RGB_Farbe_in_Grau(aChr)
      End If
    End If
  Next aChr
End Sub
' ---------------------------------------------------------------------------------------------------------------


Private Sub Bilder_graufaerben(aRng As Range)
'  H. Chris Gast, 4. bis 7. Januar 2015, 18. Januar 2016, 21. Februar 2017.
'  Alle Bilder in der Dokumentkomponente entfärben.
'  Leider gibt es keinen Mindestkontrast!
'  ***************************************
Dim i As Integer  ' laufende Bildnummer
'  ***************************************
  ' a) Bilder als Inline-Shapes suchen:
  If aRng.InlineShapes.Count > 0 Then
    For i = 1 To aRng.InlineShapes.Count
      aRng.InlineShapes(i).PictureFormat.ColorType = msoPictureGrayscale
    Next i
  End If
  ' b) Bilder als Shapes suchen:
  On Error GoTo Zeile_Ende
  If aRng.ShapeRange.Count > 0 Then
    For i = 1 To aRng.ShapeRange.Count
      If aRng.ShapeRange(i).Type = msoPicture _
      Or aRng.ShapeRange(i).Type = msoLinkedPicture Then   ' 13 oder 11
        "aRng.ShapeRange(i).Select
        aRng.ShapeRange(i).PictureFormat.ColorType = msoPictureGrayscale
      End If
    Next i
  End If
'  ***************************************
Zeile_Ende:
On Error GoTo 0
End Sub
' ---------------------------------------------------------------------------------------------------------------


Private Sub Tabellenzellen_grau_faerben(rng0 As Range)
' H. Chris Gast, Januar 2015, Januar 2016, Februar 2017, 10. + 18. April 2017, 3. Mai 2017.
' Tabellenfeld-Schattierungen werden durch zwei Grauschattierungen ersetzt,
' rötliche Farben etwas dunkler, sonstige Farben etwas heller.
' Schraffuren werden auch entfärbt, bleiben aber erhalten.
'
' Bei schattierten Tabellenzellen wird ggf. der Text in schwarze Farbe geändert
```

```
' und die Zeichenschattierung gelöscht,  um einen Mindestkontrast sicherzustellen.
' Schraffuren hinter Text werden überdeckt.
' ****************************
Dim atable As Word.Table
Dim acell As Cell
"Dim c1 As Cell
' ****************************
  For Each atable In rng0.Tables
    ' Schattierungen/Schraffuren unterschiedlich ersetzen bzw. löschen:
    For Each acell In atable.Range.Cells
      ' Hinter Buchstaben keine Schraffur und keine separate Schattierung:
      ' (wird jetzt in der Unterprozedur für  Einzelzeichen erledigt)
      "If Len(acell.Range.Text) > 2 Then
      "   If acell.Range.Shading.BackgroundPatternColor <> wdColorAutomatic Then
      "      ' Bei Zellenschattierungen Schrift in schwarz:
      "      acell.Range.Font.ColorIndex = wdAuto
      "      acell.Range.HighlightColorIndex = wdAuto
      "      acell.Range.Font.Shading.BackgroundPatternColor = wdColorAutomatic
      "   End If
      "   If acell.Range.Shading.Texture <> 0 Then
      "      ' bei Schraffuren weißen Texthintergrund erzeugen:
      "      acell.Range.HighlightColorIndex = wdWhite
      "   End If
      "End If
      ' ********
      ' Vordergrundschattierung bei Schraffuren durch hellgrau (Grau 125) ersetzen:
      If acell.Range.Shading.ForegroundPatternColor <> wdColorAutomatic _
      And acell.Range.Shading.ForegroundPatternColor <> -1 Then
        acell.Range.Shading.ForegroundPatternColor = wdColorGray125
      End If
      ' ********
      ' Hintergrundschattierungen unterschiedlich ersetzen:
      If acell.Range.Shading.BackgroundPatternColor = wdColorAutomatic Then
        ' keine Zellenschattierung; nichts tun.
      ElseIf acell.Range.Shading.BackgroundPatternColorIndex = wdRed _
      Or acell.Range.Shading.BackgroundPatternColorIndex = wdDarkRed _
      Or acell.Range.Shading.BackgroundPatternColorIndex = 22 Then
        ' rötliche/rosa Schattierung durch "Grau 35" ersetzen:
        With acell.Range.Shading
          .BackgroundPatternColor = wdColorGray35
          .ForegroundPatternColor = wdColorAutomatic
          ".Texture = wdTextureNone
        End With
      Else
        ' sonstige Schattierung durch "Grau 20" ersetzen:
        With acell.Range.Shading
          .BackgroundPatternColor = wdColorGray20
          .ForegroundPatternColor = wdColorAutomatic
          ".Texture = wdTextureNone
        End With
      End If
    Next acell
  Next atable
End Sub
' ---------------------------------------------------------------------------------------------------------


Private Sub Einzelzeichen_RGB_Farbe_in_Grau(rng0)
' H. Chris Gast, Februar 2017, März 2017, April 2017, 4. Mai 2017.
' Berechnung der RGB-Farbwerte und daraus des Grauwertes für Text und Hintergrund.
' Zuweisung von Ersatz-Grauwerten für den Druck im Schwarzweiß-Drucker.
' Die Helligkeitsdifferenz zwischen Text und Hintergrund muss dabei größer als 125 sein
```

```
' (offizielles Barrierefrei-Kriterium).
' Schraffuren werden an Stellen mit Text durch weiße Farbmarkierung des Textes überdeckt.
'
' Formel aus http://www.office-loesung.de/ftopic421244_0_0_asc.php.
' kombiniert mit Formel aus https://de.wikipedia.org/wiki/Grauwert,
' stark modifiziert.
'
' Gelöstes Problem:
' Wenn die Farben nur Codes der neuen Windows-Tabelle enthalten ("Theme colours"),
' etwa in docx-Dateien (oder ungespeichert in doc-Dateien), gibt es keine absolute Farbe!
' Hier wird dann (im Testbetrieb) "blauer Text" und "gelber Hintergrund" angesetzt;
' durch das Speichern als "doc"-Datei und erneute Öffnen in der Rahmenprozedur wird
' dieses Problem jetzt aber umgangen (März 2017).
'
' Datentyp Long für Ganzzahlen von -2'147'483'648 bis 2'147'483'647
Dim lngColorText As Long
Dim lngColorText_Grau As Long
Dim lngColorHtg As Long  ' Hintergrund
Dim lngColorHtg_Grau As Long
'
Dim IntRedText As Integer
Dim IntGreenText As Integer
Dim IntBlueText As Integer
Dim IntRedHtg As Integer
Dim IntGreenHtg As Integer
Dim IntBlueHtg As Integer
Dim lngGrauwertText As Long
Dim lngGrauwertHtg As Long
'
Dim Bl_Hintergrund_Auto As Boolean
   Bl_Hintergrund_Auto = False
'  ********************************************************************
'  Schraffuren (Texturen und Vordergrundschattierungen) beseitigen
'  (außer bei Leerzeichen):
'  ********************************************************************
'  Textur bei Einzelbuchstaben durch graue Schattierung (Grau 25) ersetzen:
  If rng0.Font.Shading.ForegroundPatternColorIndex <> 0 _
  Or rng0.Font.Shading.Texture <> 0 <> 0 _
  Or rng0.Paragraphs(1).Shading.Texture <> 0 Then
    ' Leerzeichen ausnehmen:
    If rng0.Text <> " " Then
     rng0.Font.Shading.Texture = 0
     rng0.Font.Shading.ForegroundPatternColorIndex = wdWhite '= wdAuto
     rng0.Font.Shading.BackgroundPatternColorIndex = wdGray25
    End If
  End If
'  ********************************************************************
'  Markierfarbe bzw. Hintergrundschattierung
'  ********************************************************************
  If rng0.HighlightColorIndex <> 0 Then
  '  ColorIndex der Markierfarbe des Ranges in Farbwerte umwandeln,
    Select Case rng0.HighlightColorIndex
     ' (Case = 0, wenn keine Markierfarbe angewendet wurde)
     Case 1
      lngColorHtg = 0
     Case 2
      lngColorHtg = 16711680
     Case 3
      lngColorHtg = 16776960
     Case 4
       lngColorHtg = 65280
     Case 5
```

```
        lngColorHtg = 16711935 ' rosa/Magenta
      Case 6
        lngColorHtg = 255
      Case 7
        lngColorHtg = 65535
      Case 8
        lngColorHtg = 16777215 ' weiß
      Case 9
        lngColorHtg = 8388608
      Case 10
        lngColorHtg = 8421376
      Case 11
        lngColorHtg = 32768
      Case 12
        lngColorHtg = 8388736
      Case 13
        lngColorHtg = 128
      Case 14
        lngColorHtg = 32896 ' dunkelgelb
      Case 15
        lngColorHtg = 8421504
      Case 16
        lngColorHtg = 12632256
      Case Else
        ' (kommt hier nicht vor)
    End Select
  Else
  '  Hintergrundsschattierung:
    lngColorHtg = rng0.Font.Shading.BackgroundPatternColor
    ' Korrektur für negative Zahlen bei "weiß" und bei "automatisch":
    If lngColorHtg = -603914241 Then
      lngColorHtg = 16777215 ' weiß
    ElseIf lngColorHtg = -16777216 Then
      Bl_Hintergrund_Auto = True
      ' Ich setze hier bei Hintergrund Auto = weiß:
      lngColorHtg = 16777215 ' weiß
    ElseIf lngColorHtg < 0 Then
      MsgBox "keine RGB-Farbe, ""Theme-Color""! Sie wird testweise ersetzt.", vbCritical
      "Exit Sub
      ' testweise durch Hintergrund-Schattierung = gelb ersetzen:
      lngColorHtg = 65535
    End If
  "Else
  "   MsgBox "keine Farbmarkierung und keine Hintergrundsschattierung!"
  End If
'  ******
'  Die drei Farbkomponenten des Hintergrunds(rot, grün, blau):
  IntRedHtg = lngColorHtg And vbRed
  IntGreenHtg = (lngColorHtg And vbGreen) \ &H100
  IntBlueHtg = (lngColorHtg And vbBlue) \ &H10000
'  ******************************
'  Grauwert des Hintergrunds (Skala 0 bis 255)
  lngGrauwertHtg = (0.299 * IntRedHtg + 0.587 * IntGreenHtg + 0.114 * IntBlueHtg)
'  ********************************************************************
'  Zeile_Textfarbe:
'  ********************************************************************
  lngColorText = rng0.Font.Color
'  Korrektur für negative Zahlen bei "weiß" und bei "automatisch":
  If lngColorText = -603914241 Then
    lngColorText = 16777215 ' weiß
  ElseIf lngColorText = -16777216 Then
    ' Ich setze bei Textfarben "schwarz" anstelle "automatisch"!
```

```
    ' (Bei Hintergründen ist es anders, da wäre die Farbe des Druckerpapier zu wählen)
    lngColorText = 0
  ElseIf lngColorText < 0 Then
    MsgBox "keine RGB-Textfarbe, ""Theme-Color""! Sie wird testweise ersetzt.", vbCritical
    "Exit Sub
    ' testweise durch Textfarbe = blau ersetzen:
    lngColorText = 16711680
  Else
    ' nichts tun.
  End If
'  ******
'  Die drei Farbkomponenten des Textes (rot, grün, blau):
  IntRedText = lngColorText And vbRed
  IntGreenText = (lngColorText And vbGreen) \ &H100
  IntBlueText = (lngColorText And vbBlue) \ &H10000
  ' Kontrolle:
  '' Debug.Print IntRedText & ", " & IntGreenText & ", " & IntBlueText
'  ************************************
'  Grauwert (Skala 0 bis 255)
  lngGrauwertText = (0.299 * IntRedText + 0.587 * IntGreenText + 0.114 * IntBlueText)
'  ***********************************************************************
'  Farben durch Grautöne ersetzen:
'  ***********************************************************************
'  Entsprechenden Code der Ersatz-Grautöne berechnen.
'  Die Helligkeitsdifferenz zwischen Text und Hintergrund muss größer als 125 sein
'  (offizielles Barrierefrei-Kriterium nach https://www.w3.org/TR/AERT).
  If Bl_Hintergrund_Auto = True Then
    ' nur Textfarbe ändern:
    ' Kontrast wird etwas größer als nötig (255 - 115 = 140):
    If lngGrauwertText > 115 Then lngGrauwertText = 115
  ElseIf lngGrauwertHtg < lngGrauwertText Then
    ' heller Text auf dunklem Hintergrund.
    ' Hintergrund ggf. dunkler, Text ggf. heller (190 - 62 = 128):
    If lngGrauwertHtg > 62 Then lngGrauwertHtg = 62
    If lngGrauwertText < 190 Then lngGrauwertText = 190
  Else
    ' Normalfall: dunkler Text auf hellem Hintergrund.
    ' Hintergrund ggf. heller, Text ggf. dunkler (190 - 62 = 128):
    If lngGrauwertHtg < 190 Then lngGrauwertHtg = 190
    If lngGrauwertText > 62 Then lngGrauwertText = 62
  End If
  lngColorHtg_Grau = lngGrauwertHtg + lngGrauwertHtg * &H100 + lngGrauwertHtg * &H10000
  lngColorText_Grau = lngGrauwertText + lngGrauwertText * &H100 + lngGrauwertText * &H10000
'  ************************************
'  Dem Zeichen den errechneten Textgrauwert und ggf. Hintergrunds-Grauwert zuweisen:
  If Bl_Hintergrund_Auto = True Then
    ' Kein Highlight und keine Hintergrundschattierung.
    rng0.Font.Color = lngColorText_Grau
    rng0.Font.Shading.ForegroundPatternColorIndex = 0
    '' Rng0.Font.Shading.Texture = 0
  Else
    rng0.Font.Color = lngColorText_Grau
    rng0.HighlightColorIndex = 0
    rng0.Font.Shading.BackgroundPatternColor = lngColorHtg_Grau
    rng0.Font.Shading.ForegroundPatternColorIndex = 0
    '' Rng0.Font.Shading.Texture = 0
  End If
'  ***********************************************************************
End Sub
'-----------------------------------------------------------------------------------------------------------
```

15.4 Farben entfernen

(Für das folgende Makro habe ich mir ein Tastenkürzel zugeordnet)

```
Sub Farbmarkierungen_in_Markierung_beseitigen()
'  Dipl.-Ing. Hanna-Chris Gast, Stand 19. März 2019.
'
Dim aTable As Table
'  ****************
'  Farbmarkierung beseitigen, die eine "Hervorhebung" ist:
   Selection.Range.HighlightColorIndex = wdNoHighlight
'
'  Farbmarkierung beseitigen, die eine "Schattierung" bzw. "Hintergrundsfarbe" ist:
   With Selection.Font.Shading
      .Texture = wdTextureNone
      .ForegroundPatternColor = wdColorAutomatic
      .BackgroundPatternColor = wdColorAutomatic
   End With
'
'  Farbmarkierung beseitigen, die auf Absatzeigenschaften beruht:
   With Selection.ParagraphFormat.Shading
      .Texture = wdTextureNone
      .ForegroundPatternColor = wdColorAutomatic
      .BackgroundPatternColor = wdColorAutomatic
   End With
'
'  Farbmarkierung eines Tabellenfelds beseitigen.
    If Selection.Information(wdWithInTable) And Selection.Type <> 2 Then
       With Selection.Cells.Shading
       .Texture = wdTextureNone
       .ForegroundPatternColor = wdColorAutomatic
       .BackgroundPatternColor = wdColorAutomatic
         End With
    Else
      ' 4. Juli 2018:
       For Each aTable In Selection.Range.Tables
           With aTable.Range.Cells.Shading
             .Texture = wdTextureNone
             .ForegroundPatternColor = wdColorAutomatic
             .BackgroundPatternColor = wdColorAutomatic
           End With
       Next aTable
   End If
End Sub


Sub Farbmarkierungen_und_Schattierungen_in_allen_StoryRanges_entfernen()
' H. Chris Gast, Stand 18. Januar 2016.
' Entferne die Farbmarkierungen (Highlights) in allen Dokument-Komponenten,
' ferner alle Hintergrunds- und Vordergrunds-Schattierungen und Texturen.
'
' Ferner werden (wahlweise) in Tabellen die Schattierungen von Tabellenzellen
' entfernt. Es werden dabei auch Tabellen in Fußnoten berücksichtigt.
'
' Jede Dokumentkomponente soll nur ein einziges Mal aufgesucht werden.
' ********************
Dim RngStory As Range
Dim rngSection As Section
```

```
Dim rngHeaderFooter As HeaderFooter
Dim shpShape As Shape        ' Textfelder (und anderes)
Dim BL_Textfeld As Boolean   ' Suche nach Textfeldern in Kopf- oder Fußzeile
Dim atable As Word.Table
Dim IntAbfrage As Integer
Dim i As Integer: i = 0
'  ****************************************************************************
'  Abfrage, ob Tabellenfelder-Schattierungen entfernt werden sollen.
  ' Diese Abfrage ist lästig, wenn es keine solchen gibt, deshalb nur fragen, wenn es dies gibt:
  For Each RngStory In ActiveDocument.StoryRanges
    For Each atable In RngStory.Tables
      "MsgBox atable.Range.Cells.Shading.Texture = wdTextureNone
      "MsgBox atable.Range.Cells.Shading.ForegroundPatternColor = wdColorAutomatic
      "MsgBox atable.Range.Cells.Shading.BackgroundPatternColor = wdColorAutomatic
       If atable.Range.Cells.Shading.Texture = wdTextureNone = True _
       And atable.Range.Cells.Shading.ForegroundPatternColor = wdColorAutomatic = True _
       And atable.Range.Cells.Shading.BackgroundPatternColor = wdColorAutomatic = True Then
          " MsgBox "Keine Schattierung in Tabelle"
       Else
          " MsgBox "Schattierung in Tabelle!"
         i = i + 1
       End If
    Next atable
  Next RngStory
  "MsgBox i & " Tabelle(n) hat/haben Schattierung in einigen Zellen"
  If i > 0 Then
    IntAbfrage = MsgBox("Sollen auch Tabellenfelder in " & i & " Tabellen entfärbt werden?", _
              vbYesNo, "Auch Tabellenfeld-Schattierungen?")
  Else
    ' Wenn keine Tabelle im Haupttext, keine Abfrage nötig.
    IntAbfrage = 7
  End If
'  ****************************************************************************
'  Alle (wirklich alle) Dokumentkomponenten durchsuchen, aber nur je einmal:
'  ****************************************************************************
'  Alle Dokumentkomponenten außer Fuß- und Kopfzeilen:
  For Each RngStory In ActiveDocument.StoryRanges
    If RngStory.StoryType < 6 Then
      ' ohne Kopf- und Fußzeilen, die weiter unten erfasst werden:
      ''''''''''''''''''''''''
      "MsgBox rngStory.StoryType
      RngStory.HighlightColorIndex = wdNoHighlight
      RngStory.Font.Shading.BackgroundPatternColor = wdColorAutomatic
      RngStory.Font.Shading.ForegroundPatternColor = wdColorAutomatic
      ' Farbmarkierung von Absatzformatierungen beseitigen:
      With RngStory.ParagraphFormat.Shading
        .Texture = wdTextureNone
        .ForegroundPatternColor = wdColorAutomatic
        .BackgroundPatternColor = wdColorAutomatic
      End With
      ' ********
      ' Farbmarkierung von Tabellenfeldern beseitigen:
      If IntAbfrage = 6 Then
        If RngStory.Tables.Count > 0 Then
          For Each atable In RngStory.Tables
            With atable.Range.Cells.Shading
              .Texture = wdTextureNone
              .ForegroundPatternColor = wdColorAutomatic
              .BackgroundPatternColor = wdColorAutomatic
            End With
          Next atable
        End If
```

```
      End If
      '''''''''''''''''''''''''
    End If
  Next RngStory
'  ***********************************************
'  Kopf und Fußzeilen (in allen Sections):
  For Each rngSection In ActiveDocument.Sections
    ' Kopfzeilen:
    For Each rngHeaderFooter In rngSection.Headers
      If rngHeaderFooter.LinkToPrevious = False Then
        rngHeaderFooter.Range.HighlightColorIndex = wdNoHighlight
        rngHeaderFooter.Range.Font.Shading.BackgroundPatternColor = wdColorAutomatic
        rngHeaderFooter.Range.Font.Shading.ForegroundPatternColor = wdColorAutomatic
        '''''''''''''''''''
         With rngHeaderFooter.Range.ParagraphFormat.Shading
          .Texture = wdTextureNone
          .ForegroundPatternColor = wdColorAutomatic
          .BackgroundPatternColor = wdColorAutomatic
        End With
        '''''''''''''''''''
        ' Suche nach Textfeldern in Kopf-/Fußzeilen nur ein einziges Mal durchführen!
        ' (sie werden sonst bei jedem rngHeaderFooter gefunden!)
        If BL_Textfeld = False Then
           For Each shpShape In rngHeaderFooter.Shapes
             If shpShape.Type = msoTextBox Then
                shpShape.TextFrame.TextRange.HighlightColorIndex = wdNoHighlight
                shpShape.TextFrame.TextRange.Font.Shading.BackgroundPatternColor _
                = wdColorAutomatic
                shpShape.TextFrame.TextRange.Font.Shading.ForegroundPatternColor _
                = wdColorAutomatic
             End If
           Next
           BL_Textfeld = True
        End If
        '''''''''''''''''''
      End If
    Next rngHeaderFooter
    ' **********
    ' Fußzeilen:
    For Each rngHeaderFooter In rngSection.Footers
      If rngHeaderFooter.LinkToPrevious = False Then
        rngHeaderFooter.Range.HighlightColorIndex = wdNoHighlight
        rngHeaderFooter.Range.Font.Shading.BackgroundPatternColor = wdColorAutomatic
        rngHeaderFooter.Range.Font.Shading.ForegroundPatternColorIndex = wdColorAutomatic
      End If
    Next rngHeaderFooter
  Next rngSection
'  **********************
End Sub
' ------------------------------------------------------------------------------------------------------
```

Analog dazu auch die Textfarben und ggf. die Farben von Bildern in allen Dokumentkomponenten entfernen:

```
Sub Farben_von_Text_und_ggf_Bildern_in_allen_Story_Ranges_beseitigen()
' H. Chris Gast, 4. bis 7. Januar 2015.
' Alle Farben vom Text und ggf. von Bildern in allen Dokumentkomponenten beseitigen:
' Funktioniert in Word 2000 und in Word 2010.
' Makro erfasst nicht nur Hauptkomponente, sondern auch Fußnoten, Textfelder usw.
'
' Rahmenmakro, das jede Dokumentkomponente und jedes Textfeld usw. aufruft:
```

```
' **********
Dim RngStory As Range
Dim rngSection As Section
Dim rngHeaderFooter As HeaderFooter
Dim shpShape As Shape        ' Textfelder (und anderes)
Dim BL_Textfeld As Boolean   ' Suche nach Textfeldern in Kopf- oder Fußzeile
Dim IntBilderAbfrage As Integer
'  ***************************************************************************
'  Separate Druckdatei erzeugen:
  ActiveDocument.SaveAs FileName:=ActiveDocument.Path _
                        & "\!SW-Druckdatei_" & ActiveDocument.Name
'  ***************************************************************************
'  Abfrage, ob Bilder auch entfärbt werden sollen:
  IntBilderAbfrage = MsgBox("Sollen auch Bilder entfärbt werden?", _
                vbYesNo, "Auch Bilder?")
'  ***************************************************************************
'  Alle (wirklich alle) Dokumentkomponenten durchsuchen, aber nur je einmal:
'  ***************************************************************************
'  Alle Dokumentkomponenten außer Fuß- und Kopfzeilen:
  For Each RngStory In ActiveDocument.StoryRanges
    If RngStory.StoryType < 6 Then
      ' ohne Kopf- und Fußzeilen, die weiter unten erfasst werden:
      Call Text_und_Bildfarben_in_Range(RngStory, IntBilderAbfrage)
    End If
  Next RngStory
'  ***********************************************
'  Kopf und Fußzeilen (in allen Sections):
  For Each rngSection In ActiveDocument.Sections
    ' Kopfzeilen:
    For Each rngHeaderFooter In rngSection.Headers
      If rngHeaderFooter.LinkToPrevious = False Then
        Call Text_und_Bildfarben_in_Range(rngHeaderFooter.Range, IntBilderAbfrage)
        '''''''''''''''''''''
        ' Suche nach Textfeldern in Kopf-/Fußzeilen nur ein einziges Mal durchführen!
        ' (sie werden sonst bei jedem rngHeaderFooter gefunden!)
        If BL_Textfeld = False Then
          For Each shpShape In rngHeaderFooter.Shapes
            If shpShape.Type = msoTextBox Then
              Call Text_und_Bildfarben_in_Range(shpShape.TextFrame.TextRange, _
              IntBilderAbfrage)
            End If
          Next
          BL_Textfeld = True
        End If
        '''''''''''''''''''''
      End If
    Next rngHeaderFooter
    ' **********
    ' Fußzeilen:
    For Each rngHeaderFooter In rngSection.Footers
      If rngHeaderFooter.LinkToPrevious = False Then
        Call Text_und_Bildfarben_in_Range(rngHeaderFooter.Range, IntBilderAbfrage)
      End If
    Next rngHeaderFooter
  Next rngSection
End Sub
' ---------------------------------------------------------------------------------------------------------


Private Sub Text_und_Bildfarben_in_Range(aRng As Range, IntBilderAbfrage As Integer)
'  H. Chris Gast, 4. bis 7. Januar 2015.
'  Texte und ggf. Bilder in Dokumentkomponente entfärben:
```

```
'  ***************
'  Alle Schriftzeichen in Schwarz setzen:
   aRng.Font.Color = wdColorAutomatic
'  ********************************************************
'  Bilder in Grautöne, wenn MsgBox-Abfrage "ja" ist:
   If IntBilderAbfrage <> 6 Then Exit Sub
   Dim i As Integer    ' laufende Bildnummer
   ' a) Bilder als Inline-Shapes suchen:
   If aRng.InlineShapes.Count > 0 Then
     For i = 1 To aRng.InlineShapes.Count
       aRng.InlineShapes(i).PictureFormat.ColorType = msoPictureGrayscale
     Next i
   End If
   ' b) Bilder als Shapes suchen:
   On Error GoTo Zeile_Ende
   If aRng.ShapeRange.Count > 0 Then
     For i = 1 To aRng.ShapeRange.Count
       If aRng.ShapeRange(i).Type = msoPicture _
       Or aRng.ShapeRange(i).Type = msoLinkedPicture Then    ' 13 oder 11
         "aRng.ShapeRange(i).Select
         aRng.ShapeRange(i).PictureFormat.ColorType = msoPictureGrayscale
       End If
     Next i
   End If
Zeile_Ende:
End Sub
' ---------------------------------------------------------------------------------------------------------
```

15.5 Spielereien mit Farbmarkierungen

Ich verwende zum Markieren (Highlight) gerne die Farben gelb, hellgrün (wdBrightGreen) und türkis/hellblau (wdTurquoise) zum Markieren. Auch "Grau-25%" ist für den Druck noch geeignet, notfalls auch noch dunkelgelb. Alle andern Markier-Farben sind für den Druck **zu dunkel** und **beeinträchtigen die Lesbarkeit**.

Ich suchte mir deshalb als Ersatz füreine rote Markierung die Hintergrunds-Schattierungsfarbe Hellrosa aus und schrieb mir dafür ein Makro (auf Anfrage lieferbar). Schattierungen werden aber bei Überschriften leider nicht in das Inhaltsverzeichnis übernommen, da sie keine Farbmarkierung ("Hervorhebungsfarbe", "HighligtColor"), sondern eine Hintergrunds-Schattierung ist ("Selection.Font.Shading.BackgroundPatternColor = wdColorRose").

Als Tastenkürzel für Farbmarkierungen wählte ich mir AltGr + Zahl vom Nummernfeld rechts auf der Tastatur. Die Zahl entspricht dabei dem mir als E-Technikerin vertrauten Farbencode von Widerständen. So brauchte ich mir nichts neu zu merken (AltGr + Num4 für gelb, AltGr + Num5 für grün, AltGr + Num6 für blau usw.).

```
Sub Farbmarkierung_Gelb()
' Hanna-Chris Gast, Stand  19. März 2019.
' Dieses Makro markiert den markierten Text gelb
' und bei fehlender Textmarkierung in Tabellen das ganze Tabellenfeld.
' Gewähltes Tastenkürzel AltGr + Num4
'
'  Vorher im markierten Bereich alle Highlight- Farben entfernen
   Selection.Range.HighlightColorIndex = wdNoHighlight
```

```
'  Farbmarkierung beseitigen, die eine "Schattierung" bzw. "Hintergrundsfarbe" ist:
  With Selection.Font.Shading
    .Texture = wdTextureNone
    .ForegroundPatternColor = wdColorAutomatic
    .BackgroundPatternColor = wdColorAutomatic
  End With
'  ******
'  Tabellenfelder markieren, wenn nicht Text in nur einer Tabellenzelle markiert ist:
  If Selection.Information(wdWithInTable) And Selection.Type <> 2 Then
  "And Selection.Text Like "*" & "[!" & ChrW(7) & "-" & ChrW(13) & "]" & "*" = False Then
    ' Tabellenfeld mit gelber Hintergrunds-Schattierung versehen:
    Selection.Cells.Shading.BackgroundPatternColor = wdColorYellow
  Else
    ' Außerhalb einer Tabelle oder bei markiertem Text nur die markierte Textstelle gelb markieren:
    Selection.Range.HighlightColorIndex = wdYellow
    Options.DefaultHighlightColorIndex = wdYellow    ' nicht nötig, aber nützlich.
  End If
End Sub
```

Alle Zeichen in der "Private Use Area" von Unicode sowie alle Windows-Sonderzeichen, die **weder Symbol noch Wingdings sind**, werden mit dem folgenden Makro rot markiert. Das Problem war: Manche Windows-Sonderzeichen sind bei "Suchen-Ersetzen" nur als Klammer "(" zu finden.

Nachteil: Dieses Makro funktioniert leider ab Word 2003 nicht mehr für PUA-Zeichen mit kombinierenden Zeichen (griechische Buchstaben in Symbol mit Akzent in Unicode-Schriftart), was aber so gut wie nie vorkommt!

```
Sub Rot_markieren_alle_unerlaubten_Sonderzeichen_im_PUA()
' H. Chris Gast, März 2012.
' Dies Makro markiert alle Zeichen im PUA-Bereich rot (außer bei Fonts Symbol und Wingdings)
' (PUA = Private Use Area bei Unicode-Zeichen):
Dim chara
Dim fs
Dim ys
    For Each chara In ActiveDocument.Characters
      If chara.Text Like "[" & ChrW(57344) & "-" & ChrW(63743) & "]" Then
        chara.Select
        With Dialogs(wdDialogInsertSymbol)
          fs = .Font
        End With
          If fs = "Symbol" Then
          '     Symbol soll bleiben!
          ElseIf fs = "Wingdings" Then
          '    Wingdings soll bleiben!
          Else
            Selection.Range.HighlightColorIndex = wdRed
          End If
      End If
     'Rot markieren  von Windows-Sonderzeichen außer Symbol und Wingdings:
      If chara.Text = "(" Then
        chara.Select
        With Dialogs(wdDialogInsertSymbol)
          fs = .Font
          ys = .CharNum
        End With
        If fs <> "(normaler Text)" And fs <> "" _
            Or ys < 0 And chara.Text = "(" Then
          ys = (ys + 4096 + 61440)
          If fs = "Symbol" Then
            ' Call Symbol_U(ys)
```

```
            ElseIf fs = "Wingdings" Then
              ' Call Wingdings_U(ys)
            Else
              Selection.Range.HighlightColorIndex = wdRed
            End If
          End If
        End If
'
      Next chara
      '
End Sub
```

Mit dem folgenden Makro werden alle Zeichen mit höheren Unicode-Ebenen rot markiert:

```
Sub Rot_markieren_Unicode_höhere_Ebenen()
    ' Rot markieren von Zeichen  mit Unicodes höherer Ebenen (nach Surrogat-Paar suchen):
    Options.DefaultHighlightColorIndex = wdRed
    Selection.Find.ClearFormatting
    Selection.Find.Replacement.ClearFormatting
    With Selection.Find
      .Text = "[" & ChrW(55296) & "-" & ChrW(56319) & "][" _
                & ChrW(56320) & "-" & ChrW(57343) & "]"
      .Replacement.Text = ""
      .Replacement.Font.Color = wdColorGreen
      .Replacement.Highlight = True
      .Forward = True
      .Wrap = wdFindContinue
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = True
      .MatchSoundsLike = False
      .MatchAllWordForms = False
    End With
    Selection.Find.Execute Replace:=wdReplaceAll
    Options.DefaultHighlightColorIndex = wdYellow
End Sub
```

15.6 Schriftfarbe ändern

In Word 2010 ist es schwierig, Texte mit kräftig-blauer Farbe zu schreiben (statt einfach-blau). Dieses kräftige Blau lässt sich im Schwarzweiß-Druck auch besser lesen. Dafür habe ich mir ein Makro geschrieben und kann es mit einem Icon im Menüband (Ribbon) aufrufen.

```
Sub Farbe_Blau()
' H. Chris Gast, 31. August 2015, 27. Juni 2017.
' Textfarbe blau zuordnen.
' In Word 2010 ist es schwierig, die Grundfarbe blau zu erwischen.
  Selection.Font.ColorIndex = wdBlue
  ' Farbmarkierung (Hervorhebung) löschen:
  Selection.Range.HighlightColorIndex = wdNoHighlight
End Sub
' ----------------------------------------------------------------------------------------------------
```

15.7 Suche nach der nächsten farbigen Stelle im Text

```
Sub nächste_farbige_Stelle_suchen()
'  H. Chris Gast, Januar bis April 2015, September 2016, März 2017, Stand 21. April 2017.
'  Suche von der Cursorstelle aus die nächste Stelle (Text oder Tabellenzelle) mit Farbe.
'  Für die Analyse der gefundenen Farbe oder Farben ist ggf. ein separates Makro aufzurufen.
'
'  Findet auch die Farbe "Schwarz", die genauso wie "wdColorAutomatic" aussieht,
'  und findet auch weiße Schattierung, die wie "wdColorAutomatic" aussieht.
'  Ggf. ist das Makro zu modifizieren.
'
'  Durchsucht keine Bilder!
'
'  Dieses Makro sucht separat nach Schattierungen von Tabellenzellen
'  und vergleicht dann die Orte der Fundstellen.
'  Die gleichzeitige Suche in nur einer Suchschleife würde das Makro zu langsam machen.
'
'  Das Makro brauchte in Word 2000 zuviel Zeit (58 Sekunden für 27 Seiten);
'  schnellere Lösung mit "Suchen-Ersetzen" und "Animation":
'  Durchsuchen von 27 Seiten in 2 Sekunden statt in 58 Sekunden!
'  Diese Makroversion ist allerdings nur noch für Word 2010 und höher geeignet.
'
'  Gewähltes Tastenkürzel:  Alt+Strg + Shift + F
'  ***************************************
Dim achr As Range  ' Bereich des jeweiligen Zeichens
Dim rng0 As Range  ' Bereich des restlichen Dokuments ab Cursorstelle
   Set rng0 = Selection.Range
   rng0.SetRange Start:=rng0.End + 1, End:=ActiveDocument.Range.End
Dim Rng1 As Range   ' Bereich der Fundstelle eines Zeichens mit Farbe
Dim rng2 As Range   ' Bereich zwischen Start und Textfundstelle
Dim Textfund As Boolean
Dim para As Word.Paragraph
Dim atable As Table
Dim acell As Cell
Dim Msg1 As String
   Msg1 = ""
Dim afield As Field
'  ***************************************
'  Zeitmessen für die Makro-Dauer in Sekunden
  Dim Zeit1 As Single
  Dim Zeit2 As Single
  Dim Zeitdauer As Single
  Zeit1 = Timer
'  ***************************************
'  Makro beschleunigen:
  Application.ScreenUpdating = False
'  ***************************************
'  Suche nach Farbe bei Buchstaben
'  (auch nach "Schwarz" und "Weiß):
'  ***************************************
  '''  Für Word 2000:
  ''   ' (brauchte fast 1 Minute für 30 Seiten!):
  ''   ' Man kann in Word 2000 zwar nach Schriftfarbe und "Highlight" suchen,
  ''   ' aber die Suche nach "Shading" funktionierte nicht in Word 2000.
  ''   ' Deshalb die (langsamere) Lösung mit "For Each...Characters".
  ''   ' Dabei wird hier gleichzeitig auch nach "Highlight" und nach Schriftfarbe gesucht.
  ''   ' ****
  ''   ' Felder ausschließen:
  ''   ActiveWindow.View.ShowFieldCodes = True
  ''   '
  ''   For Each achr In rng0.Characters
  ''      If achr.HighlightColorIndex <> wdAuto _
```

```
"     Or achr.Font.Shading.BackgroundPatternColor <> wdColorAutomatic _
"     Or achr.Font.Shading.ForegroundPatternColor <> wdColorAutomatic _
"     Or achr.ParagraphFormat.Shading.BackgroundPatternColor <> wdColorAutomatic _
"     Or achr.Font.Color <> wdColorAutomatic Then
"       ' Hier wurde eine farbige Text-Stelle gefunden.
"       ' Felder (Inhaltsverzeichnis) ausschließen:
"       If achr.Fields.Count = 0 Then
"         Textfund = True
"         achr.Select
"         ActiveWindow.View.ShowFieldCodes = False
"         Set Rng1 = Selection.Range
"         GoTo Zeile_Tabellensuche
"         Exit For
"       End If
"     End If
"   Next achr
"   ' Wenn keine Buchstabenfarben im restlichen Dokument gefunden:
"   Textfund = False
"   ActiveWindow.View.ShowFieldCodes = False
"   Set Rng1 = Selection.Range
' ************************************
' Für Word 2010 (schnellere Lösung):
  rng0.Select
' Zuerst farblose Textzeichen (wdColorAutomatic) mit Animation markieren:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  Selection.Find.Replacement.Font.Animation = wdAnimationLasVegasLights
  With Selection.Find
    .Text = ""
    .Font.Color = wdColorAutomatic
    .Highlight = False
    .Font.Shading.BackgroundPatternColor = wdColorAutomatic
    .Font.Shading.ForegroundPatternColor = wdColorAutomatic
    ".ParagraphFormat.Shading.BackgroundPatternColor = wdColorAutomatic
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
' ****************
' Felder animieren:
  For Each afield In rng0.Fields
    afield.Result.Font.Animation = wdAnimationLasVegasLights
    "afield.Code.Font.Animation = wdAnimationLasVegasLights
  Next afield
' ****************
' Das erste Zeichen ohne Animation (= Textzeichen mit Farbe) finden:
  rng0.Select
Zeile_Weitersuchen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    ".Text = ""
    .Text = "^?"  ' dauert länger, ist aber genauer.
    .Font.Animation = wdAnimationNone
    .Replacement.Text = ""
```

```
      .Forward = True
      ".Wrap = wdFindContinue
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
    End With
    Selection.Find.Execute
    If Selection.Find.Found Then
      ' Fehlerhafte Fundstelle ausschließen:
      If Selection.Range.HighlightColorIndex = wdAuto _
      And Selection.Font.Shading.BackgroundPatternColor = wdColorAutomatic _
      And Selection.Font.Shading.ForegroundPatternColor = wdColorAutomatic _
      And Selection.ParagraphFormat.Shading.BackgroundPatternColor = wdColorAutomatic _
      And Selection.Font.Color = wdColorAutomatic Then
        Selection.Collapse Direction:=wdCollapseEnd
        Selection.MoveRight
        GoTo Zeile_Weitersuchen
      Else
       ' Farbige Stelle gefunden:
        Textfund = True
       Set Rng1 = Selection.Range  ' farbige Stelle
      End If
    Else
      Set Rng1 = Selection.Range    ' gesamter Suchbereich
      Textfund = False
    End If
'  ****************
'  Absätze mit Highlight oder Schattierung
Set rng2 = Selection.Range
rng2.SetRange Start:=rng0.Start, End:=Rng1.End - 1
  ' die Suche kann lange dauern, deshalb prüfen, ob sich die Suche lohnt.
  If rng2.HighlightColorIndex = 9999999 _
  Or rng2.ParagraphFormat.Shading.BackgroundPatternColor = 9999999 _
  Or rng2.ParagraphFormat.Shading.ForegroundPatternColor = 9999999 Then
    ' Absatz  mit Highlight oder Schattierung suchen:
    For Each para In rng2.Paragraphs
      If para.Range.HighlightColorIndex <> wdAuto _
      Or para.Range.ParagraphFormat.Shading.BackgroundPatternColor <> wdColorAutomatic _
      Or para.Range.ParagraphFormat.Shading.ForegroundPatternColor <> wdColorAutomatic Then
        ' Hier wurde ein schattierter Absatz gefunden.
        If para.Range.Fields.Count = 0 Then
          Set Rng1 = para.Range
          Rng1.Select
          Textfund = True
          Exit For
        End If
      End If
    Next para
  End If
'  ****************
'  "Las-Vegas-Animation" überall wieder beseitigen:
  ActiveDocument.Range.Font.Animation = wdAnimationNone
'  ************************************
Zeile_Tabellensuche:
'  ************************************
'  Suche nach Schattierungen in Tabellenzellen, (auch!) ohne Text:
  For Each atable In rng0.Tables
    If Textfund = False Then
      ' Alle Tabellenzellen prüfen:
```

```
      For Each acell In atable.Range.Cells
        If acell.Range.Start < rng0.Start Then
          ' nicht durchsuchen
        ElseIf acell.Range.Shading.BackgroundPatternColor <> wdColorAutomatic _
        Or acell.Range.Shading.ForegroundPatternColor <> wdColorAutomatic Then
          ' schattierte Tabellenzelle gefunden.
          acell.Range.Select
          ActiveWindow.ScrollIntoView Selection.Range
          Msg1 = "Hier ist eine farbige Stelle (Tabellenzellenschattierung)."
          GoTo Zeile_Ergebnis
        End If
      Next acell
    Else
      ' Bei jeder Tabellenzelle auch prüfen, ob sie hinter der Textfundstelle liegt:
      For Each acell In atable.Range.Cells
        If acell.Range.Start < rng0.Start Then
          ' nicht durchsuchen
        ElseIf acell.Range.Shading.BackgroundPatternColor <> wdColorAutomatic _
        Or acell.Range.Shading.ForegroundPatternColor <> wdColorAutomatic Then
          ' schattierte Tabellenzelle gefunden.
          If acell.Range.Start > Rng1.Start Then
            ' Tabellenfundstelle liegt hinter Textfundstelle,
            ' Textfundstelle ausgeben:
            Rng1.Select
            ActiveWindow.ScrollIntoView Selection.Range
            Msg1 = "Hier ist eine farbige Stelle (Text)."
            GoTo Zeile_Ergebnis
          Else
            ' Tabellenfundstelle liegt vor Textfundstelle,
            ' Tabellenfundstelle ausgeben:
            acell.Range.Select
            ActiveWindow.ScrollIntoView Selection.Range
            Msg1 = "Hier ist eine farbige Stelle (Tabellenzellenschattierung)."
          End If
          GoTo Zeile_Ergebnis
          Exit For
        End If
      Next acell
    End If
  Next atable
' ************************************
Zeile_Ergebnis:
' ************************************
' Fundstelle auf dem Bildschirm anzeigen:
  Application.ScreenUpdating = True
  ActiveWindow.ScrollIntoView Selection.Range
  ' Fundstelle nicht so extrem weit oben auf dem Bildschirm:
  ActiveWindow.ScrollIntoView Selection.Previous(wdLine, 1), True
' ************************************
' Zeitmessen für die Makro-Dauer, eine Nachkommastelle:
  Zeit2 = Timer
  Zeitdauer = Format((Zeit2 - Zeit1), "##,##0.0")
' ************************************
' Ergebnis mit Rechenzeit als MsgBox ausgeben:
  If Msg1 <> "" Then
    ' Tabellenzellenschattierungen gefunden:
    MsgBox Msg1 & vbCrLf _
    & "Die Rechenzeit war " & Zeitdauer & " Sekunden."
    "Call Farben_und_Grauwert_ausgeben_an_Markierung
  ElseIf Textfund = True Then
    ' keine Tabellenzellenschattierungen gefunden:
    MsgBox "Hier ist eine farbige Stelle im Text." & vbCrLf _
```

```
    & "Die Rechenzeit war " & Zeitdauer & " Sekunden."
    "Call Farben_und_Grauwert_ausgeben_an_Markierung
  Else
    ' Weder Farbe im Text noch Tabellenschattierung gefunden:
    Selection.Collapse
    MsgBox "Keine (weitere) Fundstelle für Farben in Text oder Tabellen." & vbCrLf _
    & "Die Rechenzeit war " & Zeitdauer & " Sekunden."
  End If
End Sub
' ----------------------------------------------------------------------------------------------------
```

Finden der jeweils nächsten roten Markierung

Mit dem folgenden Hilfsmakro kann man bei umfangreichen Dateien schnell die nächste rote Markierung ("Highlight") finden.

```
Sub nächste_Farbmarkierung_rot_oder_rosa_suchen()
' H. Chris Gast, Stand 27. April 2015.
' Nur nächste rote oder rosa Farbmarkierung suchen, aber keine andere.
' Stürzt nicht mehr ab, wenn nichts gefunden wird,
' Makro bleibt nicht mehr am Dokumentende hängen.
'
' Gewähltes Tastenkürzel:  AltGr + Shift + R
'
  Selection.Collapse Direction:=wdCollapseEnd
Zeile_Suchbeginn:
  Selection.Find.ClearFormatting
  Selection.Find.Highlight = True
  With Selection.Find
    .Text = ""
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindStop   'wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute
  If Selection.Find.Found = True Then
   If Selection.Range.HighlightColorIndex > 99 Then Selection.Collapse
'
   If Selection.Range.HighlightColorIndex = wdRed _
   Or Selection.Range.HighlightColorIndex = wdPink Then
    " Selection.Collapse Direction:=wdCollapseEnd
     Exit Sub
   ElseIf Selection.End = ActiveDocument.Range.End Then
     MsgBox "Keine bzw. keine weitere Fundstelle für rote oder rosa Markierungen!"
     Exit Sub
   Else
     Selection.Collapse Direction:=wdCollapseEnd
     Selection.MoveRight Unit:=wdCharacter, Count:=1
     GoTo Zeile_Suchbeginn
   End If
  Else
    MsgBox "Keine bzw. keine weitere Fundstelle für rote oder rosa Markierungen!"
  End If
  Selection.MoveRight Unit:=wdCharacter, Count:=1
```

```
End Sub
' ----------------------------------------------------------------------------------------------------------------------
```

```
Sub hellrosa_Schattierung_suchen()
'  H. Chris Gast, 11. Januar 2015.
'  Suche die nächste Textstelle mit einer von zwei bestimmten hellrosa Farbschattierungen.
'  Funktioniert auch bei Word 2000.
'  Will man alle hellrosa-Töne finden, muss man den "ColorIndex = 5" abfragen, dann
'  findet das Makro aber auch Farbtöne, die "Magenta" sind, was hier aber nicht gesucht wird.
'  **********************
Dim aChr As Range
Dim Rng0 As Range
  Set Rng0 = Selection.Range
  Rng0.SetRange Start:=Rng0.End, End:=ActiveDocument.Range.End
  '
  For Each aChr In Rng0.Characters
''   If Selection.Font.Shading.BackgroundPatternColorIndex = 5 Then
    If aChr.Font.Shading.BackgroundPatternColor = 16764159 _
    Or aChr.Font.Shading.BackgroundPatternColor = wdColorRose Then
      aChr.Select
      MsgBox "Hier wurde eine hell-rosa Schattierung gefunden!"
      Exit Sub
    End If
  Next aChr
  MsgBox "Keine (weiteren) hellrosa Textstellen im Text."
End Sub
```

16 Sprache zuweisen für die Rechtschreibprüfung

In Word funktioniert die Spracherkennung bei der Rechtschreibprüfung nur mangelhaft. Nach einem Fremdwort oder nach einem Formelzeichen will Word plötzlich das nächste Wort nach Italienischer oder einer mir unbekannten afrikanischen Sprache prüfen.

In Word 2000 ist der Pfad, um einem markierten Text eine Sprache zuzuweisen, aufwendig (zuviele Klicks nötig). Deshalb bastelte ich mir (mit dem Makrorekorder und Nachbearbeitung) Makros für die drei häufigsten Sprachen, mit denen ich zu tun habe.

Einfach fraglichen Text markieren, und den für das jeweilige Makro festgelegte Tastenkombination anklicken:

```
Sub Deutsch()
' Makro aufgezeichnet am 2007-12-06 von H. Chris GAST
'  Festgelegtes Tastenkürzel  Strg + 1
   Selection.LanguageID = wdGerman
   Selection.NoProofing = False
   Application.CheckLanguage = True
End Sub

Sub Englisch()
' Makro aufgezeichnet am 2007-12-06 von H. Chris GAST
'  Festgelegtes Tastenkürzel  Strg + 5
   Selection.LanguageID = wdEnglishUK
   Selection.NoProofing = False
   Application.CheckLanguage = True
End Sub

Sub französisch()
' Makro aufgezeichnet am 2007-12-06 von H. Chris GAST
'  Festgelegtes Tastenkürzel  Strg + 4
   Selection.LanguageID = wdFrench
   Selection.NoProofing = False
   Application.CheckLanguage = True
End Sub
```

Ich habe mir im Lauf der Zeit noch viele anderen nützlichen (und weniger nützlichen) Makros geschrieben, aber da empfehle ich jedem, sich nur das als Makro aufzuschreiben, was er für nötig hält.

WICHTIG: Vor dem Schreiben eines neuen Makros sollte man sich vielleicht eine Sicherheitskopie von der **Normal.dot** machen, normalerweise zu finden unter:
C:\Dokumente und Einstellungen\...<Benutzername>...\Anwendungsdaten\Microsoft\Vorlagen.
oder
C:\Users\...<Benutzername>...\AppData\Roaming\Microsoft\Vorlagen

17 Per VBA auf den Explorer zugreifen

17.1 Standard-Ordner ausgeben und ggf. im Explorer den Pfad öffnen

Man kann mittels VBA den Explorer öffnen, und zwar mit einem Shell-Befehl. Hierbei bewirkt der Schalter "/e", dass der Explorer zweispaltig geöffnet wird.

Im Folgenden wird der Speicher-Ordner für Word-Dokumente ("…doc" oder "…docx") gesucht.

```
Sub Explorer_Pfad_Dateispeicherort_ausgeben()
  ' Ordner für "Dokumente" im Direktfenster angeben.
   Debug.Print Options.DefaultFilePath(wdDocumentsPath)
   MsgBox Options.DefaultFilePath(wdDocumentsPath)
End Sub
```

Im Folgenden wird der Ordner für Dokumentvorlagen ("…dotm" usw.) gesucht:

```
Sub Explorer_Pfad_öffnen_Word_Dokumentvorlagen_Ordner()
  ' Ordner der Dokumentvorlagen im Direktfenster angeben und im Explorer öffnen.
Dim Pfad1 As String
   Pfad1 = Options.DefaultFilePath(wdUserTemplatesPath)
   Debug.Print Pfad1
   Shell "Explorer.exe /e, " & Pfad1, vbNormalFocus
End Sub
```

Der "Word-Startup-Ordner eignet sich für Dateien mit Makros, die bei Starten von Word geladen werden sollen:

```
Sub Explorer_Pfad_öffnen_Word_STARTUP_Ordner()
  ' Ordner "…word\startup" im Direktfenster angeben und im Explorer öffnen.
Dim Pfad1 As String
   Pfad1 = Options.DefaultFilePath(wdStartupPath)
   Debug.Print Pfad1
   Shell "Explorer.exe /e, " & Pfad1, vbNormalFocus
End Sub
```

Bevorzugten Bilderpfad fest einstellen

Beim Einfügen von Bildern mit dem Tastenkürzel "Alt + E, G, A" (oder über Menü) wird im Explorer der von Microsoft vorgebene Ordner "Meine Bilder" geöffnet. Dies lässt sich in **Word 2010** in den Optionen nicht mehr verstellen.

Abhilfe:
Es wird zuerst der gewünschte Pfad per Hand aus dem Explorer ausgelesen, zum Bespiel bei mir zuhause ist dieser: "D:\transfer"
Dann gibt man im Direktfenster (Aufrufen mit Alt + F11) das folgende ein (rot = Beispiel):

```
Application.Options.DefaultFilePath(wdPicturesPath) = "D:\transfer"
```

Mehr dazu siehe http://www.siebener-kurier.de/chris-aufsaetze/Word-Probleme.pdf [3].

17.2 Makro zum Ändern von Dateinamen in allen Dateien eines Ordners

Das folgende Makro erlaubt es, im Explorer in den Dateinamen eines Ordners Zeichenfolgen zu ersetzen. **Voreingestell**t ist im folgenden Makro das Ersetzen von Leerzeichen durch Unterstriche.

Damit dieses Makro auch in Word 2000 funktioniert, habe ich auf den "FilePicker" verzichtet und den normalen Befehl für das Öffnen von Dateien verwendet. Man muss eine beliebige Datei des Zielordners öffnen (aber keinen Hyperlink!), die sich dann aber gleich wieder schließt, nachdem der Pfad ausgelesen wurde.

Kleines Problem: Geöffnete Word-Dateien können grundsätzlich nicht umbenannt werden, sondern müssten vorher geschlossen werden. Ein Schließen aller betroffenen Word-Dateien per Makro ist zwar möglich, jedoch nicht ratsam. Ist die Word-Datei mit dem Makro in dem betreffenden Ordner enthalten, würde spätestens durch das Schließen dieser Datei das Makro abgebrochen.

Achtung! Dateinamen von von Programmen dürfen nicht geändert werden. Voreingestellt ist deshalb, dass das Makro keine Dateinamen in Laufwerk "c" ändert. Für Anwender, die Dateinamen auf Laufwerk "c" ändern wollen, ist die Sicherheitsabfrage entsprechend zu ändern.

```
Sub Dateinamen_im_Explorer_ändern()
' H. Chris Gast, erstellt November 2012, geändert 2013, Stand 5. Mai 2014.
' Word 2000 bis Word 2010, Windows XP und Windows 7.
' Dieses Makro ersetzt in allen Dateien eines Ordners einzelne Zeichenfolgen in den Dokumentnamen.
' Voreingestellt (default) ist: Ersetze Leerzeichen durch Unterstriche.
'
' Mankos:
' Bei geöffneten Word-Dateien kann der Dateiname nicht geändert werden.
' Durchsucht keine Unterordner.
' *****************
Dim Pfad1 As String
Dim Term1 As String  ' zu ersetzende Schriftzeichen
Dim Term2 As String  ' Ersatztext
Dim Name1 As String
Dim Name2 As String
Dim i As Integer: i = 0  ' Durchsuchte Dateien
Dim j As Integer: j = 0  ' Zu ändernde Dateien
Dim L As Integer: L = 0  ' misslungene Änderungen
Dim Endung As String
Dim Dok1 As String
Dim Antwort As Integer      ' MsgBox-Antwort
'  ********************************************************************
'  Eingabe des zu durchsuchenden Ordners,
'  Eingabe von Term1 (zu ersetzende Zeichen) und Term2 (Ersatztext),
'  und Sicherheitskontrolle.
'  ********************************************************************
  '''  Inputbox für "Pfad eingeben" (mit Defaultwert = normaler Ordner für Word-Dateien):
  '''   Pfad1 = InputBox("Bitte zu durchsuchendes Explorer-Verzeichnis angeben:", _
  '''                 "Dateinamen ändern", Options.DefaultFilePath(wdDocumentsPath))
  '''   If Pfad1 = "" Then
  '''      Antwort = MsgBox("Die Eingabe stimmte nicht. Das Makro wird abgebrochen.", _
  '''                     vbOKOnly, "Dateinamen ändern")
  '''      Exit Sub
  '''   End If
'  Ein Auswählen des Ordners mittels InputBox fand ich zu umständlich.
'  Ich wählte deshalb die Auswahlfunktion von "Dialogs(wdDialogFileOpen)".
'  ****************
  Antwort = MsgBox("Bitte im folgenden Dialog den gewünschten Ordner aufsuchen und dort " _
      & "irgendeine Datei anklicken." & vbCrLf _
```

```
      & vbCrLf _
      & "Achtung: Geöffnete Word-Dateien können nicht umbenannt werden!", , "Dateinamen ändern")
' ****************
  ' Mit dem Dialog "Datei öffnen" den Explorer-Ordner wählen, und  eine Datei anklicken:
  '' Application.FileDialog(msoFileDialogFilePicker) ' geht nicht in Word 2000.
  '' Application.Dialogs(wdDialogFileOpen).Show 'macht in Word 2010 bei zip-Dateien Absturz.
' Lösung mit dem alten WordBasic (macht nur bei Hyperlinks Ärger):
  On Error GoTo Zeile_Fehler_wegen_gewählter_Datei
  With Dialogs(wdDialogFileOpen)
   If .Display Then
      Pfad1 = WordBasic.FilenameInfo$(.Name, 5)
      Dok1 = WordBasic.FilenameInfo$(.Name, 3)
    Else
      Antwort = MsgBox("Abbruch des Makros. Es wurde keine Datei angeklickt!", _
                 vbOKOnly, "Dateinamen ändern")
      Exit Sub
    End If
  End With
  On Error GoTo 0
' *****************
  Name1 = Dir(Pfad1 & Application.PathSeparator & "*.*")
' *****************
' Sicherheitsabfrage:
' Wenn in Laufwerk "c", dann nur in Unterordnern von "c:/Users":
  If LCase(Left(Pfad1, 1)) = "c" Then
     If InStr(LCase(Pfad1), LCase("c:\users")) <> 1 Then
        MsgBox "Dieses Makro darf nur auf Dateien des Anwenders, nicht auf " _
             & "die Software in Laufwerk 'c' angewendet werden!", vbCritical, "ACHTUNG!"
        Exit Sub
     End If
  End If
' ****************
' Inputbox für zu ändernde Schriftzeichen in den Dateinamen eingeben:
  Term1 = InputBox("Zu Ersetzendes angeben" & vbCrLf & "(Vorgabewert: 'Leerzeichen'), " _
              & vbCrLf & " dabei Groß- und Kleinbuchstaben unterscheiden;" _
              & vbCrLf & vbCrLf & "Ordner: " & vbCrLf & "'" & Pfad1 & "'.", _
              "Dateinamen ändern", " ")
  If Term1 = "" Then Exit Sub
' ****************
' Inputbox für Ersatz-Schriftzeichen in den Dateinamen:
  Term2 = InputBox("Ersatz angeben (Vorgabewert: 'Unterstrich')", "Dateinamen ändern", "_")
  If Term2 = "" Then
    Antwort = MsgBox("Es wurde nichts eingegeben, das Makro wird beendet.", , "Abbruch")
    Exit Sub
  End If
' ***********************************************************************
' Das "eigentliche" Makro:
' ***********************************************************************
' Schleife bilden:
  On Error GoTo Zeile_Dateiname_nicht_änderbar
  Do While Name1 <> ""
    i = i + 1  ' gefundene Ordner
    Name2 = Replace(Name1, Term1, Term2)
    If Name2 <> Name1 Then
       j = j + 1  ' zu ändernde Ordner
       Name Pfad1 & Application.PathSeparator & Name1 As _
          Pfad1 & Application.PathSeparator & Name2
    End If
    ' Nächste Datei abfragen:
    Name1 = Dir
  Loop
' ***********************************************************************
```

```
'  Protokoll des Ergebnis:
'  ********************************************************************
  If L = 0 Then
    Antwort = MsgBox("Es wurden " & i & " Dateinnamen geprüft und " & j _
      & " Dateinnamen geändert.", vbOKOnly, "Ergebnis")
  ElseIf L = 1 Then
    Antwort = MsgBox("Es wurden " & i & " Dateinnamen geprüft und " & j - L _
      & " Dateinnamen geändert." _
      & vbCrLf & "Eine Datei ließ sich nicht ändern, vielleicht, weil sie geöffnet ist.", _
      vbOKOnly, "Ergebnis")
  Else
    Antwort = MsgBox("Es wurden " & i & " Dateinnamen geprüft und " & j - L _
      & " Dateinnamen geändert." _
      & vbCrLf & L & " Dateien ließen sich nicht ändern, vielleicht, weil sie geöffnet sind.", _
      vbOKOnly, "Ergebnis")
  End If
' ****************************
' Ende des Makros:
Exit Sub
'  ********************************************************************
Zeile_Fehler_wegen_gewählter_Datei:
  MsgBox "Es wurde keine Datei oder ein Link ins Internet gewählt." & vbCrLf _
    & "Bitte Makro neu starten und echte Datei wählen."
  Exit Sub
'  ***************************
Zeile_Dateiname_nicht_änderbar:
  Antwort = MsgBox("Bei der Datei '" & Name2 & "' lässt sich der Dateiname nicht ändern!" _
    & vbCrLf & "Möglicherweise ist diese Datei geöffnet.", vbOKOnly, "Dateinamen ändern")
    L = L + 1
  Resume Next
End Sub
```

Will man auch die Unterordner erfassen, empfehle ich die Unterprozedur von Josef Ehrensberger, Westendorf (Tirol, Austria) aufzurufen, zu finden zum Beispiel unter:
http://www.ms-office-forum.net/forum/showthread.php?t=266393

****

Das folgende Makro listet alle Dateien und Unterordner eines gewählten Explorer-Ordners aus. Die vorhandenen Beispiele im Internet versagten bei mir, wenn ich die Unterordner auflisten wollte.
Für GetAttr(Pfad1 & Name1) erhalte ich auch "8208" oder "48" oder "21" statt "16", wie es die Beispiele und die "Hilfe" angeben; die erhaltene Zahl enthält nämlich auch andere Attribute, die hinzuaddiert sind).

```
' Für eine erweiterte MsgBox mit Unicodezeichen am Anfang des Makromoduls:
Private Declare Function MessageBoxW Lib "user32.dll" _
   (ByVal hWnd As Long, _
   ByVal lpText As Long, _
   ByVal lpCaption As Long, _
   ByVal uType As Long) As Long
'

Sub Explorer_Dateien_und_Unterordner_mit_Dir_auflisten()
' H. Chris Gast, Oktober 2013, 30. März 2014, 13. Januar 2016.
' Makro ist zusammengefügt aus zwei Makros von November 2012, modifiziert.
```

```
' Word 2000 bis Word 2010.
' Das Makro listet alle Dateinamen und Unterordner-Namen im angegebenen Ordner
' in einer erweiterten Messagebox an und zählt sie dabei.
' Wahlweise (oder bei zuviel Fundstellen für eine erweiterte Messagbox)
' erfolgt die Auflistung in einer Datei.
'
' Da der "FilePicker" in Word 2000 nicht funktioniert, wird hier
' die alte Word-Basic-Funktion verwendet.
'
' Für Unterordner ergaben sich unterschiedliche Attributsnummern.
' Hier muss ich unterschiedliche Werte abfragen.
' *********************
Dim Pfad1 As String
Dim Name1 As String
Dim Ergebnis1 As String: Ergebnis1 = ""
Dim i As Integer:  i = 0
Dim Endung As String
Dim oShell As Object: Set oShell = CreateObject("WScript.Shell")
Dim Msg0 As String
Dim Msg1 As String
Dim Msg2 As String
Dim strMsgText As String
Dim IntAbfrage As Integer
'  ********************************************************
  "Pfad1 = Fkt_Filepicker  ' Filepicker geht nicht in Word 2000!
'  ********************************************************
'  Ordner in Windows auswählen:
  Msg0 = "Bitte im folgenden Dialog einen Ordner wählen, eine der Dateien " _
      & " (außer Links) anklicken und ""Öffnen"" drücken."
  oShell.Popup Msg0, 1, "Windows-Ordner durchsuchen", 0 ' vbInformation
   Set oShell = Nothing
 "  Set Dok1 = ActiveDocument
' ***************************************************************
'  Pfad auslesen mit WordBasic:
On Error Resume Next
  With Application.Dialogs(wdDialogFileOpen)
    If .Display Then
       Pfad1 = WordBasic.FilenameInfo$(.Name, 5)
     Else
       MsgBox "Es wurde keine Datei gewählt. Gegebenenfalls nochmals probieren!"
       Exit Sub
     End If
  End With
  '
  If Err.Number > 0 Then
    ' MsgBox "Fehler " & Err.Number & vbCr & Err.Description, _
          vbOKOnly, "Fehlermeldung"
    MsgBox "Es wurde möglicherweise ein Link statt einer Datei ausgewählt. " _
          & vbCrLf & "Das Makro wird beendet."
      Exit Sub
  End If
On Error GoTo 0
'  ********************************************************
'  Dateien im Ordner auflisten:
'  ********************************************************
'  Die erste Fundstelle wird vor der Schleife ermittelt:
  "Name1 = Dir(Pfad1 & Application.PathSeparator & "*")
  '(Nur normale Dateien mit irgendeiner Endung)
  Name1 = Dir(Pfad1 & Application.PathSeparator & "*.*")
'  **********
'  Schleife durch alle Dateien des ausgewählten Explorer-Ordners:
  Do While Name1 <> ""     ' Schleife beginnen.
```

```
    Ergebnis1 = Ergebnis1 & Name1 & vbCrLf
    i = i + 1
    Name1 = Dir  ' nächsten Namen abfragen:
  Loop
'  ***********
  If i = 1 Then
    ' (bei i = 0 hätte man den Ordner nicht wählen können)
    Msg1 = "Der Ordner " & vbCrLf & """" & Pfad1 & """" & vbCrLf _
       & "enthält " & i & ChrW(160) & "eine Datei:" _
       & vbCrLf & vbCrLf & Ergebnis1
    'MsgBox Msg1
  Else
    Msg1 = "Der Ordner " & vbCrLf & """" & Pfad1 & """" & vbCrLf _
       & "enthält " & i & ChrW(160) & "Dateien:" _
       & vbCrLf & vbCrLf & Ergebnis1
    "MsgBox Msg1
  End If
'  ********************************************************
'  Unterverzeichnisse im Ordner auflisten:
'  ********************************************************
  Dim Ergebnis2 As String: Ergebnis2 = ""
  Dim j As Integer: j = 0       ' Anzahl der Fundstellen
'  *********
  If Right(Pfad1, 1) <> "\" Then Pfad1 = Pfad1 & "\"
  If Dir(Pfad1, vbDirectory) = "" Then
     MsgBox "Ordner existiert nicht!"
     Exit Sub
  End If
  '
  Name1 = Dir(Pfad1, vbDirectory)    ' Ersten Eintrag abrufen.
  Do While Name1 <> ""    ' Schleife beginnen.
    ' Aktuelles und übergeordnetes Verzeichnis ignorieren:
    If Name1 <> "." And Name1 <> ".." Then
      ' Eintrag nur anzeigen, wenn es sich um ein Verzeichnis ("16") handelt:
      ' (vbDirectory = 16)
      "If GetAttr(Pfad1 & Name1) = 8208 _
      "   Or GetAttr(Pfad1 & Name1) = 16 Then
      "If GetAttr(Pfad1 & Name1) = 8208 _
      "Or GetAttr(Pfad1 & Name1) = 48 _
      "Or GetAttr(Pfad1 & Name1) = 16 Then
      If GetAttr(Pfad1 & Name1) Mod 32 >= 16 Then
        Ergebnis2 = Ergebnis2 & Name1 & vbCrLf
        j = j + 1
      End If
    End If
    Name1 = Dir    ' Nächsten Eintrag abrufen.
  Loop
  '
  If j = 0 Then
     Msg2 = "Der Ordner " & vbCrLf & """" & Pfad1 & """" & vbCrLf _
          & "enthält keine Unterverzeichnisse."
  ElseIf j = 1 Then
      Msg2 = "Der Ordner " & vbCrLf & """" & Pfad1 & """" & vbCrLf _
          & "enthält  nur ein Unterverzeichnis, dieses ist: " & vbCrLf & vbCrLf _
          & Ergebnis2
  Else
     Msg2 = "Der Ordner " & vbCrLf & """" & Pfad1 & """" & vbCrLf _
          & "enthält " & j & ChrW(160) & "Unterverzeichnisse, diese sind: " & vbCrLf & vbCrLf _
          & Ergebnis2
  End If
   " MsgBox Msg2
  "Abfrage = MsgBox(Msg2 & vbCrLf & vbCrLf & "Wollen Sie das gesamte Ergebnis als Datei?", _
```

```
                vbYesNoCancel + vbDefaultButton2, "Makro für Dir-Befehl-Anwendung")
  ' Bei "Ja" kommt "6" heraus.
'  ****************************
  strMsgText = Msg1 & vbCrLf & vbCrLf & Msg2 & vbCrLf & vbCrLf _
  & "Wollen Sie das gesamte Ergebnis als Datei?"
  If i + j < 40 Then
    ' (Auch eine erweiterte MsgBox kann nicht beliebig viele Zeilen darstellen)
    IntAbfrage = MessageBoxW(0, StrPtr(strMsgText), StrPtr("Dateien im Ordner"), 260)
  Else
    IntAbfrage = 6
  End If
  If IntAbfrage <> 6 Then
    ' End  (beendet Programm, auch übergeordnete Prozeduren)
    Exit Sub
  End If
'  ************************************************************
'  Ergebnis in eine neue Datei:
  Documents.Add  'Type:=wdNewBlankDocument
  Selection.WholeStory
  Selection.Collapse Direction:=wdCollapseEnd
  Selection.TypeParagraph
  Selection.TypeText Text:="Ergebnis der Analyse eines Windows-Ordners"
  Selection.Paragraphs.Style = wdStyleHeading2
  Selection.TypeParagraph
'  ********
'  Absatzabstand für folgenden Text auf Null stellen:
  With Selection.ParagraphFormat
    .Alignment = wdAlignParagraphLeft
    .SpaceBefore = 0
    .SpaceAfter = 0
    .SpaceAfterAuto = False
    .Alignment = wdAlignParagraphLeft
  End With
  Selection.Font.Size = 10
  Selection.Font.Name = "Arial"
'  Die Auswertung:
  Selection.TypeText Text:=Msg1 & vbCrLf & vbCrLf & Msg2
End Sub
' ---------------------------------------------------------------------------------------------------------------


''Private Function Fkt_Filepicker()
''Dim strPfad As String
''    With Application.FileDialog(msoFileDialogFilePicker)
''        .Title = "Bitte Datei wählen"
''        .Filters.Clear
''        If .Show = -1 Then
''            strPfad = .SelectedItems(1)
''        End If
''    End With
''    'MsgBox strPfad
''    Fkt_Filepicker = strPfad
''End Function
''' ---------------------------------------------------------------------------------------
```

17.3 Per Makro im Explorer die AutoKorrektur-Dateien wie "MSO1031.acl" finden

Für die Übertragung von Tastenkürzeln und (deutschen) AutoKorrekturen biete ich ein Formblatt mit eingebauten Makros an, siehe [4]:
http://www.siebener-kurier.de/chris-aufsaetze/Formblatt-Tastenkuerzel+Autokorrekturen_uebertragen.doc.

In Word werden einfache AutoKorrekturen nicht mehr in der Normal.dot, sondern in separaten Dateien gespeichert mit der Dateiendung ".acl" (Auto Correct List files). Für **Windows XP** lautet der Pfad wie folgt:

C:\Dokumente und Einstellungen\<Benutzername>\Anwendungsdaten\Microsoft\Office.

In Window 7 kann der Pfad wie folgt lauten:

C:\Users\<Benutzername>\AppData\Roaming\Microsoft\Office.

Für jeden Benutzer gibt es einen eigenen Satz von ".acl-"Dateien, und für jede Sprache gibt es eine eigene Datei, selbst für amerikanisches, australisches und britisches Englisch gibt es verschiedene ".acl-"Dateien.

Nur Einträge mit Formatierungen, Bildern oder Sonderzeichen wie Smileys werden weiterhin in der Normal.dot gespeichert.

Will man auch AutoKorrekturen für andere Sprachen (Englisch, Französisch usw.) übertragen, empfehle ich, die entsprechenden ".acl"-Dateien wie "MSO1033.acl" für Englisch (USA) auf einen Stick zu speichern und in den neuen Computer zu übertragen. Oft hat man mehr .acl-Dateien, als man überhaupt Sprachen verwendet. Das folgende Makro gibt weitgehend an, für welche Sprache welche ".acl"-Datei dient. Dabei ist zu beachten, dass die acl-Datei für British-Englisch möglicherweise nur einen Querverweis auf die acl-Datei für US-Englisch enthält, usw.

Die ".acl"-Dateien mit Angabe des Ordners im Explorer sowie der zugehörigen Sprachen findet man mit dem folgenden Makro:

```
Sub acl_Dateien_suchen_und_auflisten_und_erklären()
' H. Chris Gast, 13. Nov. 2012, 16. Dezember 2012, 7. November 2015.
' Word 2000 bis Word 2010, getestet Windows XP und Windows 7.
' Sucht die für die AutoKorrektur nötigen ".acl"-Dateien und gibt ggf. die zugehörige Sprache an.
' zum Beispiel "MSO1031.acl    Deutsch (Deutschland)".
Dim Pfad1 As String
Dim Name1 As String
Dim Rng0 As Range
Dim i As Integer:  i = 0
Dim Abfrage As Integer
'  ***************************
'  Pfad zu den acl-Dateien bestimmen:
  ' Gesuchter Pfad: C:\Users\ ...\AppData\Roaming\Microsoft\Office
  On Error Resume Next
  '1. Versuch:
  Pfad1 = Replace(Options.DefaultFilePath(wdUserTemplatesPath), "vorlagen", "Office")
  ' Erste acl-Datei abrufen.
  Name1 = Dir(Pfad1 & Application.PathSeparator & "*.acl")
  ' ***
  If Err.Number > 0 Or Name1 = "" Then
    ' 2. Versuch:
    Pfad1 = Replace(Options.DefaultFilePath(wdUserTemplatesPath), "templates", "Office")
    Name1 = Dir(Pfad1 & Application.PathSeparator & "*.acl")
  End If
  If Err.Number > 0 Or Name1 = "" Then
    ' Nicht gefunden:
    MsgBox "acl-Dateien nicht gefunden!" & vbCrLf _
```

```
        & "Versuchen Sie es im Explorer unter:" & vbCrLf _
        & Replace(Options.DefaultFilePath(wdUserTemplatesPath), _
        "vorlagen (oder templates)", "Office")
      Exit Sub
    End If
'  ***************************
'  Neues Dokument erstellen:
    Application.Documents.Add
    Selection.WholeStory
    Selection.Collapse Direction:=wdCollapseEnd  ' (Briefkopf mit Datum soll ggf. erhalten bleiben)
    Selection.TypeParagraph
    Selection.TypeParagraph
    Set Rng0 = Selection.Range
'  ***************************
'  Schleife über alle ".acl"-Dateien mit abgefangenen Fehlern:
    Do While Name1 <> ""    ' Schleife beginnen.
      If Name1 = "" Then Exit Do
      Selection.TypeText Text:=vbCrLf & Name1 & "    "
      On Error Resume Next
'     Den Sprachcode aus "Name1" der Markierung zuordnen:
      Selection.LanguageID = Val(Mid(Name1, 4))
'     Aus der zugeordneten Sprache den Sprachnamen entnehmen,
'     aber nur dann, wenn die Zuordnung wirklich funktioniert hat:
      If Selection.LanguageID = Val(Mid(Name1, 4)) Then
         Selection.TypeText Text:=Languages(Selection.LanguageID)
         ' Zum Beispiel ergibt die Zahl "1031": "Deutsch, Deutschland".
       Else
         Selection.TypeText Text:="((Sprache ließ sich nicht aufrufen!))"
      End If
      If Err.Number = 5941 Then
        If Mid(Name1, 4, 4) = 127 Then
          Selection.TypeText Text:="""Mathe-AutoKorrektur-Liste"""
        Else
         Selection.TypeText Text:="((Dieser Nummer ist keine Sprache zugeordnet))"
        End If
        On Error GoTo 0
      ElseIf Err.Number > 0 Then
        Selection.TypeText Text:=vbCrLf & "unklarer Fehler " & Err.Number
        On Error GoTo 0
      End If
      i = i + 1
      ' nächsten Namen abfragen:
      Name1 = Dir
    Loop
'  ***********************************
'  Wieder als Sprache "Deutsch" einstellen und Hinweise geben:
    Selection.LanguageID = 1031
    Selection.TypeParagraph
    Selection.TypeParagraph
    Selection.TypeText Text:="Sie können die AutoKorrektur-Dateien (""*.acl"")  " _
            & "auf einen USB-Stick speichern und in einen neuen Computer übertragen." _
            & vbCrLf & vbCrLf _
            & "ACHTUNG: AutoKorrekturen mit Bildern, Sonderzeichen oder Formatierungen werden " _
            & "nicht hier, sondern Sprach-unabhängig in der Normal.dot gespeichert."
'  ***********************************
'  An den Anfang des Dokuments gehen und Zählung ausgeben:
    Rng0.Select
    Selection.ParagraphFormat.Alignment = wdAlignParagraphLeft
    Selection.TypeText Text:="Es gibt im Ordner " & vbLf & """" & Pfad1 & """" & vbLf & "insgesamt " _
                & i & " AutoKorrektur-Dateien mit der Endung "".acl"":" & vbCrLf & vbCrLf
'  ***********************************
'  Abfrage, ob der Explorer-Ordner mit den ".acl"-Dateien geöffnet werden soll:
```

```
  Abfrage = MsgBox("Es gibt im Ordner """ & Pfad1 & """ insgesamt " & i & " AutoKorrektur-Dateien." _
            & vbCrLf & vbCrLf _
            & "Wollen Sie diesen Explorer-Ordner öffnen?", vbYesNoCancel + vbDefaultButton2)
  ' Bei "Ja" kommt "6" heraus, bei "Nein" kommt "7" heraus, bei Abbrechen eine "2".
  If Abfrage = 6 Then
    Shell "Explorer.exe /e, " & Pfad1, vbNormalFocus
  End If
End Sub
' ----------------------------------------------------------------------------------------------------
```

Ich würde nur AutoKorrekturen für Sprachen übertragen, die ich auch brauche.

Hinweis: Wenn die Spracherkennung die Sprache falsch erkennt, werden auch die falsche AutoKorrekturen verwendet. Aus dem deutschen Verb "*werden*" wird dann das englische Substantiv "*warden*". Siehe dazu auch
"Einige Probleme mit Word und deren Lösung"
Als Download: www.siebener-kurier.de/chris-aufsaetze/Word-Probleme.pdf.

17.4 Alle geöffneten Word-Dateien auflisten

Das folgende Makro ist noch unbefriedigend!

Im Falle von mehreren Word-Instanzen, etwa durch gleichzeitiges Öffnen im Explorer können nicht alle Pfade ermittelt werden!

Vielleicht kann mir hier einer der Leser weiterhelfen?
E-Mail: hcgast "ät" siebener-kurier.de *(dabei "ät" durch "@" ersetzen).*

```
Sub alle_geöffneten_Worddateien_auflisten()
' H. Chris Gast, Juni 2018, 9. Juli 2018.
' Das Makro listet die Namen alle geöffneten Word-Dateien auf.
' Dies funktioniert auch, wenn mehrere Word-Instanzen geöffnet sind.
' Funktioniert zuhause und im Büro (Word 2010 unter Windows 7).
'
' Bei der Auflistung mit Pfad-Angabe gibt es Probleme, wenn mehrere Instanzen von Word
' geöffnet sind (gekennzeichnet durch "???").
' *****************
Dim Int_Einfach_zählen As Integer
Dim myTask As Task
Dim strFundliste As String: strFundliste = ""
Dim strEinzelfund As String: strEinzelfund = ""
Dim strDateiname As String: strDateiname = ""
Dim i As Integer: i = 0
''''''''''''''
Dim strDokFullname() As String
Dim strFundliste_Fullname As String: strFundliste_Fullname = ""
Dim IntAnzeige As Integer
Dim Dok() As Word.Document
'  ************************************
  Int_Einfach_zählen = Application.Windows.Count
'  Funktioniert falsch, wenn mehrere Word-Instanzen geöffnet sind!
'  Abhilfe:
```

```
'  Quelle: http://www.office-loesung.de/ftopic214123_0_0_asc.php,Lisa, 2008,
'  stark modifiziert für das Zählen und Auflisten von Word-Dateien.
  For Each myTask In Tasks
    "Debug.Print myTask.Name
    If InStr(myTask.Name, "Microsoft Word") > 0 Then
      strEinzelfund = myTask.Name
      If InStr(strEinzelfund, ".doc") > 0 Then
        ' Word-Dokument:
        strDateiname = Left(strEinzelfund, InStr(strEinzelfund, ".doc") + 4)
        ' Wenn noch nicht erfasst, in Liste der Word-Dokumente aufnehmen:
        If InStr(strFundliste, strDateiname) = 0 Then
          i = i + 1
          strFundliste = strFundliste & vbCrLf & "    " & strDateiname
          '''''''''''''
          myTask.Activate  ' funktioniert nicht bei unterschiedlichen Instanzen!
          ReDim Preserve strDokFullname(i)
          strDokFullname(i) = ActiveDocument.FullName
          If InStr(strFundliste_Fullname, strDokFullname(i)) > 0 Then
            ' falsche aktive Datei:
            strDokFullname(i) = "???unbekannter Pfad???" & "/" & Trim(strDateiname)
          End If
          strFundliste_Fullname = strFundliste_Fullname & vbCrLf & "    " & strDokFullname(i)
          '''''''''''''
        End If
      ElseIf InStr(strEinzelfund, ".dot") > 0 Then
         ' Dokument-Vorlage (dot, dotm, dotx):
        strDateiname = Left(strEinzelfund, InStr(strEinzelfund, ".dot") + 4)
        If InStr(strFundliste, strDateiname) = 0 Then
          "j = j + 1
          "strVorlagenliste = strVorlagenliste & vbCrLf & strDateiname
          i = i + 1
          strFundliste = strFundliste & vbCrLf & "    " & strDateiname
          '''''''''''''
          myTask.Activate  ' funktioniert nicht bei unterschiedlichen Instanzen!
          ReDim Preserve strDokFullname(i)
          strDokFullname(i) = ActiveDocument.FullName
          If InStr(strFundliste_Fullname, strDokFullname(i)) > 0 Then
            strDokFullname(i) = "???unbekannter Pfad???" & "/" & Trim(strDateiname)
          End If
          strFundliste_Fullname = strFundliste_Fullname & vbCrLf & "    " & strDokFullname(i)
          '''''''''''''
        End If
      Else
        ' Ungespeichertes Word-Dokument ("Dokument1" usw.)
        strDateiname = Left(strEinzelfund, InStr(strEinzelfund, " ") + 0)
        ' zeigte auch gelöschte ungespeicherte Dateien an wie z.B. "Dokument 6"!
        ' Abhilfe (das geht möglicherweise nur bei mir so?):
        If InStr(strEinzelfund, "[Kompatibilitätsmodus]") > 0 Then
          ' noch existierende Word-Datei:
          If InStr(strFundliste, strDateiname) = 0 Then
            i = i + 1
            strFundliste = strFundliste & vbCrLf & "    " & strDateiname
            ReDim Preserve strDokFullname(i)
            ' Hierbei kann es keinen Pfad geben:
            strDokFullname(i) = strDateiname
            strFundliste_Fullname = strFundliste_Fullname & vbCrLf & "    " & strDateiname
          End If
        Else
          ' (keine noch geöffnete Datei)
        End If
      End If
    End If
```

```
  Next myTask
  strFundliste = "Es wurde(n) " & i & " geöffnete Word-Datei(en) gefunden: " _
    & vbCrLf & strFundliste
  ' ************************************
  ' Auflistung ohne Pfade:
  If Int_Einfach_zählen < i Then
    ' Mehrere Word-Instanzen:
    IntAnzeige = MsgBox(strFundliste & vbCrLf & vbCrLf _
    & "Die Dateien befinden sich (teils) in unterschiedlichen Instanzen!", vbSystemModal, _
    "Auflistung aller geöffneten Word-Dateien")
    ' Anmerkung: Hierbei erzwingt "vbSystemModal" bei mehreren Instanzen
    ' ein Öffnen der MsgBox im Vordergrund.
  Else
    ' Alle Word-Dateien in nur einer Instanz, aber es sind einige ungespeicherte dabei:
    IntAnzeige = MsgBox(strFundliste, vbSystemModal, _
    "Auflistung aller geöffneten Word-Dateien")
  End If
'  ***************************************
'  Angabe mit Pfaden im Direktfenster (noch mit Problemen):
  strFundliste_Fullname = "Es wurde(n) " & i & " geöffnete Word-Datei(en) gefunden: " _
    & vbCrLf & strFundliste_Fullname
  IntAnzeige = MsgBox(strFundliste_Fullname, vbApplicationModal, "Liste mit Pfaden")
  ''Debug.Print strFundliste_Fullname
End Sub
' ---------------------------------------------------------------------------------------------------------
```

18 Name und Datum im Briefkopf automatisch einfügen mit Feiertagsangabe

Das folgende Makro erzeugt einen Briefkopf, der nicht nur das aktuelle Datum enthält, sondern in einer zweiten Zeile angibt, welcher Feiertag gerade ist. Nur ein Teil dieser Feiertage ist ein gesetztlicher Feiertag, aber der Anwender kann die Liste für seine Zwecke anpassen.

Statt "Vorname + Name" ist der jeweilige Name einzufügen.

Installation: Klicke Alt + F11, dann öffnet sich der Makro-Explorer (Makro-Editor), siehe folgendes Bild. Die neue Prozedur "*AutoNew()*" mit den zugehörigen beiden "*Private Functions*" ist in der Normal.dot nicht in einem der Module, sondern in "ThisDocument" unterzubringen.

Problem: Bei Windows 7 funktioniert "Sub AutoNew()" nicht, wenn noch keine anderen Word-Dateien geöffnet sind. **Lösung** für Windows 7 durch Ergänzung einiger Makrozeilen in der AutoExec, siehe: http://www.siebener-kurier.de/chris-aufsaetze/Word-Probleme.pdf, dort in Abschnitt 6.2.

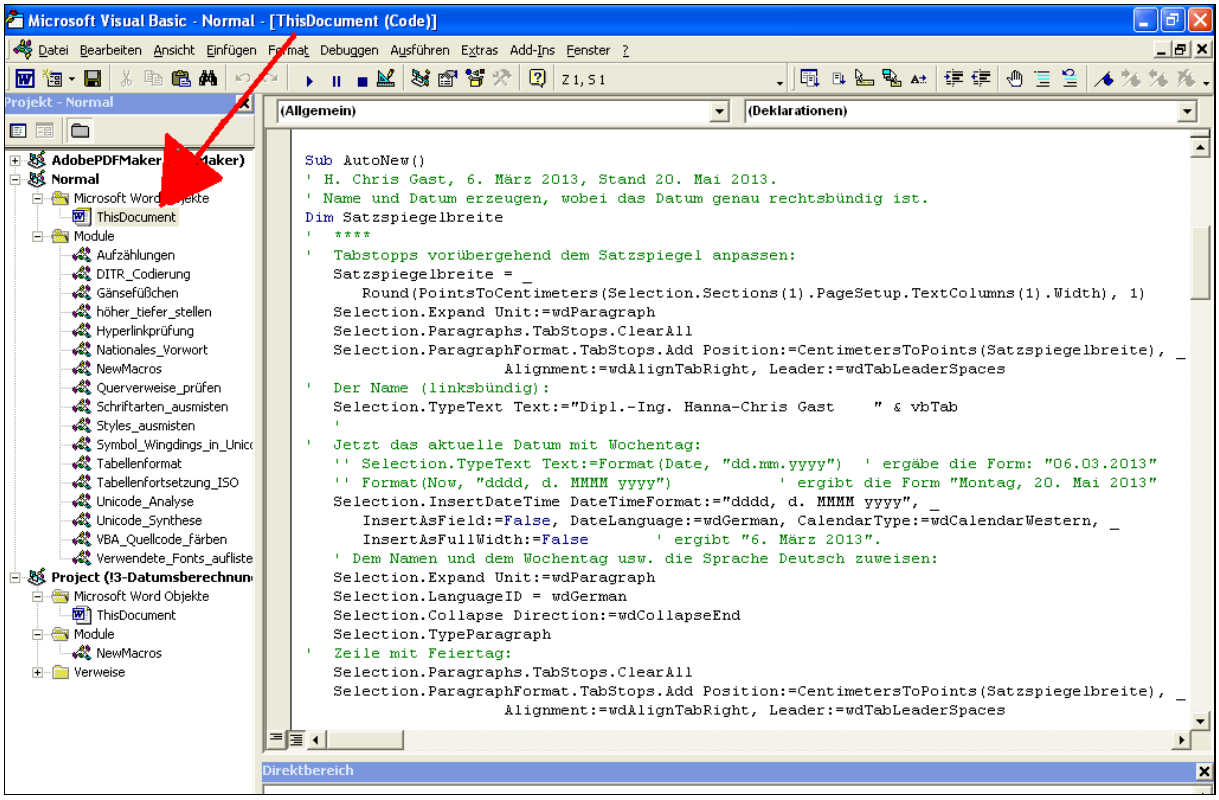


Bild 18.1: Makro-Explorer mit "Sub AutoNew()" im Modul "ThisDocument"

```
Sub AutoNew()
' H. Chris Gast, 6. März 2013, 20. Mai 2013, Stand 26. März 2014.
' Name und Datum erzeugen, wobei das Datum genau rechtsbündig ist.
' In einer zweiten Zeile wird ggf. der Name  des jeweiligen Feiertags angegeben.
Dim Satzspiegelbreite
'  ****
'  Nur bei leerem Dokument, welches auf der Normal.dot basiert, dieses Makro ausführen:
  If ActiveDocument.AttachedTemplate <> Application.NormalTemplate _
  Or Len(ActiveDocument.Range) > 1 Then Exit Sub
' ****************************************************************
'  Tabstopps vorübergehend dem Satzspiegel anpassen:
   Satzspiegelbreite = _
```

```
    Round(PointsToCentimeters(Selection.Sections(1).PageSetup.TextColumns(1).Width), 1)
  Selection.Expand Unit:=wdParagraph
  Selection.Paragraphs.TabStops.ClearAll
  Selection.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(Satzspiegelbreite), _
              Alignment:=wdAlignTabRight, Leader:=wdTabLeaderSpaces
'  Der Name (links oben):
  Selection.TypeText Text:="Vorname + Name      " & vbTab
  '
'  Jetzt das aktuelle Datum mit Wochentag:
  '' Selection.TypeText Text:=Format(Date, "dd.mm.yyyy")  ' ergäbe die Form: "06.03.2013"
  '' Format(Now, "dddd, d. MMMM yyyy")          ' ergibt die Form "Montag, 20. Mai 2013"
  Selection.InsertDateTime DateTimeFormat:="dddd, d. MMMM yyyy", _
    InsertAsField:=False, DateLanguage:=wdGerman, CalendarType:=wdCalendarWestern, _
    InsertAsFullWidth:=False       ' ergibt "6. März 2013".
  ' Dem Namen und dem Wochentag usw. die Sprache Deutsch zuweisen:
  Selection.Expand Unit:=wdParagraph
  Selection.LanguageID = wdGerman
  Selection.Collapse Direction:=wdCollapseEnd
  Selection.TypeParagraph
'  Zeile mit Feiertag:
  Selection.Paragraphs.TabStops.ClearAll
  Selection.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(Satzspiegelbreite), _
              Alignment:=wdAlignTabRight, Leader:=wdTabLeaderSpaces
  Selection.TypeText Text:="          "
  Selection.TypeText Text:=vbTab
  Selection.TypeText Text:=Feiertagsname_vollständig(Format(Now, "dd.mm.yyyy"))
  Selection.TypeParagraph
  Selection.TypeParagraph
'  Tabstopps zurückstellen:
  Selection.ParagraphFormat.Reset
End Sub
' ***********************************************

Private Function Feiertagsname_vollständig(Xdat As Date) As String
' H. Chris Gast, April 2013, neu erstellt 20. Mai 2013 (Pfingstmontag).
' Ermitteln, ob Datum "Xdat" ein Feiertag ist und wie er heißt:
' Quellen: http://de.wikibooks.org/wiki/Algorithmensammlung:_Kalender:_Feiertage)
' ************
 Dim IntJahr As Integer
 Dim i As Long
 Dim Osterdatum As Date
 Dim txtName(24) As String
 Dim Feiertage(24) As Date
 Feiertagsname_vollständig = ""   ' Wird nicht definiert, da es im Function-Name vorkommt.
' ****************
' Das Jahr ("IntJahr") aus dem übergebenen Datum ermitteln:
 IntJahr = CInt(Format(Xdat, "yyyy"))
' Bewegliche Feiertage anhand des Osterdatums ermitteln:
 Osterdatum = Ostertermin_Berechnung(IntJahr)
'  ***************************************
 Select Case Xdat
    Case CDate("01.01." & IntJahr)
       Feiertagsname_vollständig = "Neujahr"
    Case CDate("06.01." & IntJahr)
       Feiertagsname_vollständig = "Epiphanias (Heilige Drei Könige)"
    Case (Osterdatum) - 48
       Feiertagsname_vollständig = "Rosenmontag (in Berlin kein Feiertag)"
    Case (Osterdatum) - 46
       Feiertagsname_vollständig = "Aschermittwoch (in Berlin kein Feiertag)"
    Case (Osterdatum) - 3
       Feiertagsname_vollständig = "Gründonnerstag"
    Case (Osterdatum) - 2
```

```
        Feiertagsname_vollständig = "Karfreitag"
    Case Osterdatum
        Feiertagsname_vollständig = "Ostersonntag"
    Case (Osterdatum) + 1
        Feiertagsname_vollständig = "Ostermontag"
    Case CDate("01.05." & IntJahr)
        Feiertagsname_vollständig = "1.Mai"
    Case (Osterdatum) + 39
        Feiertagsname_vollständig = "Himmelfahrt"
    Case (Osterdatum) + 49
        Feiertagsname_vollständig = "Pfingstsonntag"
    Case (Osterdatum) + 50
        Feiertagsname_vollständig = "Pfingstmontag"
    Case (Osterdatum) + 60
        Feiertagsname_vollständig = "Frohnleichnam (nur katholische Gegenden)"
    Case CDate("15.08." & IntJahr)
        Feiertagsname_vollständig = "Maria Himmelfahrt"
    Case CDate("03.10." & IntJahr)
        Feiertagsname_vollständig = "Tag der deutschen Einheit"
    Case CDate("31.10." & IntJahr)
        Feiertagsname_vollständig = "Reformationstag (nur Brandenburg usw.)"
    Case CDate("01.11." & IntJahr)
        Feiertagsname_vollständig = "Allerheiligen (nur katholische Gegenden)"
    ' Für den Buß- und Bettag sind vom 1. Weihnachtstag 32 Tage und je nach
    ' Wochentag des Weihnachtstages noch einige weitere Tage abzuziehen:
    Case CDate("25.12." & IntJahr) - 32 - Weekday(CDate("25.12." & IntJahr), vbMonday)
        Feiertagsname_vollständig = "Buß- und Bettag (nur noch in Sachsen Feiertag)"
    Case CDate("25.12." & IntJahr) - 21 - Weekday(CDate("25.12." & IntJahr), vbMonday)
        Feiertagsname_vollständig = "1. Advent"  ' 21 Tage vor Weihnachten + Werktage
    Case CDate("25.12." & IntJahr) - 22 - Weekday(CDate("25.12." & IntJahr), vbMonday)
        Feiertagsname_vollständig = "Vorabend des 1. Advent (Apfelkerzen vorbereiten!)"
    Case CDate("24.12." & IntJahr)
        Feiertagsname_vollständig = "Heilig-Abend"
    Case CDate("25.12." & IntJahr)
        Feiertagsname_vollständig = "1. Weihnachtstag"
    Case CDate("26.12." & IntJahr)
        Feiertagsname_vollständig = "2.Weihnachtstag"
    Case CDate("31.12." & IntJahr)
        Feiertagsname_vollständig = "Silvester"
    '  *****
    Case Else
        Feiertagsname_vollständig = ""
End Select
End Function


Private Function Ostertermin_Berechnung(X As Integer) As Date
' H. Chris Gast, April 2013, Stand: 13. Mai 2013.
' Gleichungen aus einer Berechnungs-Anleitung der PTB nach Gauß und Lichtenberg,
' http://www.ptb.de/cms/fachabteilungen/abt4/fb-44/ag-441/darstellung-der-gesetzlichen-zeit/
'       wann-ist-ostern.html
'  Die CDate-Funktion funktioniert erst ab der Jahreszahl "100".
' Dieses Makro eignet sich nur für den Gregorianischen Kalender ab 1583.
Dim K As Integer
Dim M As Integer
Dim S As Integer
Dim A As Integer
Dim D As Integer
Dim R As Integer
Dim OG As Integer   ' Märzdatum des Ostervollmonds
Dim SZ As Integer   ' Datum des ersten Sonntags im März
```

```
Dim OE As Integer
Dim OS As Integer   ' Datum des Ostersonntags (ab 1. März gezählt)
'  ****************
  ' Gültigkeitsklärung:
  If X < 1583 Then
    MsgBox "Vor 1582 galt der Gregorianische Kalender noch nicht, " & vbCrLf _
        & "nach welchem dieses Makro das Osterdatum berechnet."
    End
  ElseIf X < 1700 Then
    MsgBox "Achtung! Der Gregorianische Kalender gilt für alle deutschen Gebiete erst ab 1700!"
  End If
'  ****************
'  Berechnung:
  K = Int(X / 100)
  M = 15 + Int((3 * K + 3) / 4) - Int((8 * K + 13) / 25)
  S = 2 - Int((3 * K + 3) / 4)
  A = X Mod 19
  D = (19 * A + M) Mod 30
  R = Int(D / 29) + (Int(D / 28) - Int(D / 29)) * Int(A / 11)
  OG = 21 + D - R
  SZ = 7 - (X + Int(X / 4) + S) Mod 7
  OE = 7 - (OG - SZ) Mod 7
  OS = OG + OE
''  Ostertermin_Berechnung = DateAdd("d", OS, "1. März 2000")
''  Ostertermin_Berechnung = DateAdd("yyyy", (X - 2000), Ostertermin_Berechnung)
  If OS < 32 Then
    Ostertermin_Berechnung = CDate(OS & ". " & "März " & X)
  Else
    Ostertermin_Berechnung = CDate((OS - 31) & ". " & "April " & X)
  End If
End Function
```

Die Funktion "Private Function Feiertagsname..." kann man auch dazu verwenden, ein Formular für Urlaubsanträge zu erstellen, auf dem automatisch die Anzahl der Werktage berechnet wird unter Berücksichtigung der Feiertage. Ein Beispiel ist hier unter:

http://www.siebener-kurier.de/chris-aufsaetze/Formblatt-Urlaubsantragsformular.dot

19 Makros zum Auflisten aller normalen Fonts und Sonderschriftarten im Dokument (alle Dokumentkomponenten)

Es werden mit dem folgenden Makro sämtliche Dokumentkomponenten (StoryRanges) und alle Kopf- und Fußzeilen sowie alle Textfelder abgesucht.

Die normalen Schriftarten werden dadurch gefunden, dass zuerst geprüft wird, ob der gesamte Bereich eine Schriftart hat. Wenn nicht, wird weiter unterteilt (Absatz, Satz, Wort, Zeichen).

Die vorhandenen Makros aus dem Internet waren alle nicht in der Lage, die **Windows-Sonderschriftarten** wie Symbol und Wingdings zu finden, wenn diese über "Einfügen → Symbole" eingefügt worden waren, oder etwa die Smileys, wenn diese über das Tastenkürzel mittels AutoKorrektur "**: -)**" usw. eingefügt wurden. Für diese Sonderschriftarten reicht in der Regel die Suche nach Zeichen im PUA-Codebereich ("Private Use Area") aus, aber es gab auch einen Fall , in der selbst dieses nichts half.
Einzige gefundene Lösung:
Jedes Zeichen im gesamten Dokument (ausnahmslos) wird abgefragt, ob es eine "*Klammer auf*" ist oder im PUA-Bereich (Privat Use Area) liegt.

```
If chara.Text = "(" Or chara.Text Like "[" & ChrW(61472) & "-" & ChrW(61695) & "]" Then
```

Dann wird mit

```
With Dialogs(wdDialogInsertSymbol)
  fs = .Font
  ys = .CharNum
End With
```

die Schriftart und der Windows-interne Zeichencode abgefragt. Damit werden die echten "Klammer auf" von den Windows-Sonderzeichen unterschieden und wieder aussortiert.

```
Option Explicit

' für normale Schriftarten
Private strFontliste As String
Private strComputerfonts As String
Private strFehlendeFonts As String
'
' für Sonderschriftarten:
Private Symbolanzahl As Integer
Private Wingdingsanzahl As Integer
Private Sonstigesanzahl As Integer
Private Unerkanntesanzahl As Integer
Private strSonderzeichenliste As String
'
' für das Ergebnis:
Private Ergebnisprotokoll As String
' ************************************************


Sub alle_Fonts_im_Dokument_auflisten_gründlich()
'  Neuere Version--Erfasst auch verkorkste Windows-Zeichen.
'  Dipl.-Ing. Hanna-Chris Gast, 14. April 2014, Stand 23. April 2014.
'  Rahmenprozedur (nach Muster eines meiner Makros von September 2011).
'  Das vorübergehende Abschalten der Bildschirm-Aktualisierung beschleunigt das Makro.
'  Die Rückkehr an die ursprüngliche Cursorstelle funktioniert nur in der Hauptkomponente.
' ***************************
```

```
Dim rng0 As Range
Dim Abfrage As Integer
Dim Dateiname As String
Dim Seitenanzahl As String
' ********
'    Zeitmessen für die Makromodul-Dauer
     Dim Zeit1, Zeit2, Zeitdauer
     Zeit1 = Timer
' *********
' Zuerst die vorhandene Bildschirmansicht speichern und ggf. Bildschirm auf "Normalansicht" stellen:
  Dim Ansichtstyp1 As Integer
  Ansichtstyp1 = (ActiveWindow.View.Type)
  If ActiveWindow.View.Type = wdNormalView = False Then
     ActiveWindow.View.Type = wdNormalView
  End If
' Bildschirmaktualisierung während des Programms abstellen.
  Application.ScreenUpdating = False
  Options.Pagination = False
' Ursprüngliche Markierung merken:
  Set rng0 = Selection.Range
' ************************************************************
' zuerst alle_Markierfarben beseitigen im Hauptteil, in Fußnoten und in Kopfzeilen  usw.
  "   Dim Rng1 As Range
  "   For Each Rng1 In ActiveDocument.StoryRanges
  "       Rng1.HighlightColorIndex = wdNoHighlight
  "       Rng1.Font.Shading.BackgroundPatternColor = wdColorAutomatic
  "       Rng1.Font.Color = wdColorAutomatic
  "   Next Rng1
  "' obiges erreicht nicht die Kopf- und Fußzeilen von weiteren Sections)
  Call Farbmarkierungen_beseitigen
' ************************************************************
' Die eigentlich Prozedur aufrufen:
  StatusBar = "    Bitte warten - das Makro läuft."
  Call alle_StoryRanges_Fonts_suchen
' ************************************************************
' Markierfarbe wieder zurückstellen:
  Options.DefaultHighlightColorIndex = wdYellow
' Alte Markierung wieder herstellen:
  rng0.Select
' Ansichtstyp zurückstellen, Seitenumbrüche und Bildschirm wieder anzeigen:
  ActiveWindow.View.Type = Ansichtstyp1
  Options.Pagination = True
  Application.ScreenUpdating = True
' Alte Markierung wieder herstellen:
  rng0.Select
' ************************************
' Im Suchen/Ersetzen-Menü alles zurückstellen
   Selection.Find.ClearFormatting
   Selection.Find.Replacement.ClearFormatting
   With Selection.Find
      .Text = ""
      .Replacement.Text = ""
      .Forward = True
      .Wrap = wdFindContinue
      .Format = False
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
   End With
' ***************************************
```

```
'  Zeitmessen für die Makromodul-Dauer mit drei Nachkommastellen:
   Zeit2 = Timer
   Zeitdauer = Format((Zeit2 - Zeit1), "##,##0.000")
'  ****************************************
      "  MsgBox "Die Windows-Sonderzeichen wurden farbig markiert." _
      "      & vbCrLf & "Die Rechenzeit betrug " & Zeitdauer & " Sekunden."
'  Ergebnis ausgeben mit Abfrage, ob es in einer Datei ausgegeben werden soll:
   Ergebnisprotokoll = Ergebnisprotokoll _
         & vbCrLf & vbCrLf & "Die Rechenzeit betrug " & Zeitdauer & " Sekunden."
'  Abfrage, ob es ausgedruckt werden soll:
   Abfrage = MsgBox(Ergebnisprotokoll _
          & vbCrLf & vbCrLf & "Wollen Sie das Ergebnis als Datei?", _
          vbYesNoCancel + vbDefaultButton2, "Auflistung der Schriftarten")
'  Bei "Ja" kommt "6" heraus, bei "Nein" kommt "7" heraus, bei Abbrechen eine "2".
   If Abfrage = 6 Then
      Dateiname = ActiveDocument.Name
      Seitenanzahl = "Das Dokument hat " & Selection.Information(wdNumberOfPagesInDocument) _
                & " Seiten."
      ' Anzeige in separater Datei:
      Documents.Add DocumentType:=wdNewBlankDocument
      ' falls in der Normal.dot schon Name und Datum vorgesehen waren, dieses stehe lassen:
      Selection.EndKey Unit:=wdStory
      ' Absatzabstand auf Null stellen und Einrückung:
      With Selection.ParagraphFormat
"         .Alignment = wdAlignParagraphLeft
"         .LeftIndent = CentimetersToPoints(0.5)
"         .FirstLineIndent = CentimetersToPoints(-0.5)
         .SpaceBefore = 0
         .SpaceAfter = 0
         .SpaceAfterAuto = False
         .Alignment = wdAlignParagraphLeft
      End With
"      Selection.Font.Name = "Arial"
"      Selection.Font.Size = 10
      ' *********
      Selection.TypeParagraph
      Selection.TypeParagraph
      Selection.TypeParagraph
      Selection.Font.Bold = True
      Selection.TypeText Text:="Schriftarten in """ & Dateiname & """:"
      Selection.Font.Bold = False
      Selection.TypeParagraph
      Selection.TypeText Text:=Ergebnisprotokoll
      Selection.TypeParagraph
      Selection.TypeParagraph
      Selection.TypeText Text:=Seitenanzahl
   End If
End Sub


Private Sub alle_StoryRanges_Fonts_suchen()
' H. Chris Gast, 4. April 2014, Stand 7. April 2013.
' Suche in allen Dokument-Komponenten nach den (normalen) Schriftarten.
' Für Sonderschriftarten siehe separates Makro.
' Jede Dokumentkomponente soll nur ein einziges Mal aufgesucht werden.
' ********************
Dim rngStory As Range
Dim rngSection As Section
Dim rngHeaderFooter As HeaderFooter
Dim astrFontList() As String
Dim shpShape As Shape        ' Textfelder und anderes
```

```
Dim i As Integer            ' Section ("Abschnitt")
Dim BL_Textfeld As Boolean   ' Suche nach Textfeldern in Kopf- oder Fußzeile
Dim aFont As Variant ' Object
Dim tt As Single
Dim strSonderzeichenhinweis As String
' *******************
' Am Anfang die Speicher zurückstellen:
  strFontliste = ""
  BL_Textfeld = False
  ' für Sonderzeichensuche:
  Symbolanzahl = 0
  Wingdingsanzahl = 0
  Sonstigesanzahl = 0
  Unerkanntesanzahl = 0
  strSonderzeichenliste = ""
' ***********************************************
' Fonts im gesamten Dokument auflisten, die auch der Computer hat:
  strComputerfonts = ""
  strFehlendeFonts = ""
  For Each aFont In Application.FontNames()
    strComputerfonts = strComputerfonts & ";" & aFont
  Next aFont
' ****************************************************************************
' Alle (wirklich alle) Dokumentkomponenten durchsuchen, aber nur je einmal:
' ****************************************************************************
' Alle Dokumentkomponenten außer Fuß- und Kopfzeilen:
  For Each rngStory In ActiveDocument.StoryRanges
    If rngStory.StoryType < 6 Then
      '(die Trennlinie über Fußnoten soll nicht erfasst werden)
      Call GetUsedFontsInRange(rngStory)
    End If
  Next
' ***************************************************
' Kopf und Fußzeilen (in allen Sections):
  For Each rngSection In ActiveDocument.Sections
    ' Kopfzeilen:
    For Each rngHeaderFooter In rngSection.Headers
      If rngHeaderFooter.LinkToPrevious = False Then
        Call GetUsedFontsInRange(rngHeaderFooter.Range)
        '''''''''''''''''''''
        ' Suche nach Textfeldern in Kopf-/Fußzeilen nur ein einziges Mal durchführen!
        ' (sie werden sonst bei jedem rngHeaderFooter gefunden!)
        If BL_Textfeld = False Then
          For Each shpShape In rngHeaderFooter.Shapes
            If shpShape.Type = msoTextBox Then
              Call GetUsedFontsInRange(shpShape.TextFrame.TextRange)
            End If
          Next
          BL_Textfeld = True
        End If
        '''''''''''''''''''''
      End If
    Next
    ' ***
    ' Fußzeilen:
    For Each rngHeaderFooter In rngSection.Footers
      If rngHeaderFooter.LinkToPrevious = False Then
        Call GetUsedFontsInRange(rngHeaderFooter.Range)
      End If
    Next
  Next
' ****************************************************************************
```

```
' Schriftartenliste nachbearbeiten:
  '  Überflüssige Trennzeichen beseitigen:
  If Left(strFontliste, 1) = ";" Then
     strFontliste = Mid(strFontliste, 2)
    '' strFontliste = Replace(strFontliste, ";", "; ")
  End If
  ' alphabetisch sortieren:
   astrFontList = Split(strFontliste, ";")
   WordBasic.SortArray astrFontList()
   strFontliste = VBA.Join(astrFontList(), vbLf)
'  ****************************************
'  Liste der Fehlenden Fonts nachbearbeiten:
  Dim strWarnung As String: strWarnung = ""
  If Len(strFehlendeFonts) > 2 Then
     ' Überflüssige Trennzeichen beseitigen:
     If Left(strFehlendeFonts, 1) = ";" Then
        strFehlendeFonts = Mid(strFehlendeFonts, 2)
     End If
     astrFontList = Split(strFehlendeFonts, ";")
     WordBasic.SortArray astrFontList()
     strFehlendeFonts = VBA.Join(astrFontList(), "; ")
'
     strWarnung = vbCrLf & vbCrLf _
        & "Achtung: Folgende verwendete Fonts fehlen im Computer: " & vbCrLf _
        & strFehlendeFonts & vbCrLf _
        & "Diese sind im Text rot markiert."
  End If
'  ****************************************
'  Windows-Sonderschriftarten nachbearbeiten:
  strSonderzeichenhinweis = Fkt_Sonderschriftarten_nachbearbeiten
'  ****************************************
'  An den Dateianfang gehen der Hauptkomponente:
  ActiveDocument.StoryRanges(1).Select
  Selection.Collapse
  If ActiveWindow.View.SplitSpecial = wdPaneNone Then
     ActiveWindow.ActivePane.View.Type = wdPrintView
  Else
     ActiveWindow.View.Type = wdPrintView
  End If
  Selection.HomeKey Unit:=wdStory
'  **************************
'  Etwas warten, damit alle obigen Befehle erledigt sind:
  tt = Timer
  Do While Timer < tt + 0.5  'Sekunden
     DoEvents
  Loop
'  **************************
'  Ergebnis ausgeben:
  Ergebnisprotokoll = "Das gesamte Dokument enthält folgende (normale) Schriftarten: " & vbCrLf _
     & vbCrLf _
     & strFontliste & "." _
     & strWarnung _
     & strSonderzeichenhinweis
End Sub
```

```
Private Sub GetUsedFontsInRange(ByVal StoryRng As Range)
' aus: http://www.vb-fun.de/cgi-bin/loadframe.pl?ID=vb/tipps/tip0367.shtml
' von einer "Angie", 2003, hier stark vereinfacht.
' Die jeweilige Dokumentkomponente nach Fonts durchsuchen. Wenn die
' Dokumentkomponente unterschiedliche Fonts hat, Suche weiter unterteilen.
  Dim para    As Paragraph
  Dim rngPara As Range
  Dim rngSent As Range
  Dim rngWord As Range
  Dim rngChar As Range
'  **********
  If StoryRng.Font.Name = "" Then
    For Each para In StoryRng.Paragraphs
      Set rngPara = para.Range
      If rngPara.Font.Name = "" Then
        For Each rngSent In rngPara.Sentences
          If rngSent.Font.Name = "" Then
            For Each rngWord In rngSent.Words
              If rngWord.Font.Name = "" Then
                For Each rngChar In rngWord.Characters
                  Call SaveFontName(rngChar)
                Next rngChar
              Else
                Call SaveFontName(rngWord)
              End If
            Next rngWord
          Else
            SaveFontName rngSent
          End If
        Next rngSent
      Else
        Call SaveFontName(rngPara)
      End If
    Next para
  Else
    Call SaveFontName(StoryRng)
  End If
' ****************
' Da diese Prozedur keine Windows-Sonderschriftarten erkennt,
' wird hierfür eine spezielle Prozedur aufgerufen:
  Call Sonderschriftartensuche(StoryRng)
End Sub


Private Sub SaveFontName(ByVal rngRange As Range)
' 4. April 2014,7. April 2014, Stand 23. April 2014.
' Vorgegebenen Bereich nach Schriftart abfragen:
Dim strFontName As String
' **************
   strFontName = rngRange.Font.Name
   If Len(strFontName) < 2 Then
      Exit Sub
   End If
'  *************
'  Speichern, wenn noch nicht in Liste:
   If InStr(strFontliste, strFontName) = 0 Then
       strFontliste = strFontliste & ";" & strFontName
       If InStr(strComputerfonts, strFontName) = 0 Then
          strFehlendeFonts = strFehlendeFonts & ";" & strFontName
          rngRange.HighlightColorIndex = wdRed
          ' rngRange.Paragraphs(1).Range.HighlightColorIndex = wdRed
```

```
    End If
  End If
End Sub


Private Sub Sonderschriftartensuche(ByVal Rng1 As Range)
' Erstellt von  Dipl.-Ing. Hanna-Chris Gast, 3. bis 14. April 2014, Stand 23. April 2014.
' Neuer Ansatz übernommen aus altem Makro zur Ersetzung von Sonderzeichen
' von 2009 bis 2011.
' Funktioniert von Word 2000 bis Word 2010, in Windows XP und in Windows 7.
' Die ersetzen Zeichen werden farbig markiert.
' Findet auch Windows-Sonderzeichen, bei denen alle bisherigen Makros versagten.
'
Dim chara As Variant
Dim i As Integer
Dim fs As String
Dim ys As Long
Dim Dateiende As Boolean
'
' Vorab in allen Dokument-Komponenten Schriftart "SymbolPS" durch "Symbol" ersetzen
' (dies ist für Computer ohne diese Schriftart nützlich):
   Rng1.Find.ClearFormatting
   Rng1.Find.Replacement.ClearFormatting
   Rng1.Find.Font.NameAscii = "SymbolPS"
   Rng1.Find.Replacement.Font.Name = "Symbol"
   With Rng1.Find
      .Text = ""
      .Replacement.Text = ""
      .Forward = True
      .Wrap = wdFindContinue
      .Format = True     ' Wichtig!
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
   End With
   Rng1.Find.Execute Replace:=wdReplaceAll
'  ************
   StatusBar = "    Bitte warten - das Makro läuft."
'  ******************************************************************************************
'  Problem: Es gibt Fälle, in denen mit "Suchen" die Sonderschriftarten nicht gefunden
'  werden.
'  Einzige Lösung:
'  Jedes Zeichen des ganzen Dokuments wird mittels "Einfügen-Symbol" danach abgefragt,
'  ob es ein Windows-Sonderzeichen ist:
'  ******************************************************************************************
'  Die eigentliche Suche nach Windows-Sonderzeichen:
   i = 0
   For Each chara In Rng1.Characters
     If chara.Text = "(" Or chara.Text Like "[" & ChrW(61472) & "-" & ChrW(61695) & "]" Then
       ' (hier ist eine Sonderschriftart)
       chara.Select
       With Dialogs(wdDialogInsertSymbol)
         fs = .Font
         ys = .CharNum
       End With
       ' Schriftart (Font) bestimmen:
       If fs <> "(normaler Text)" And fs <> "" _
       Or ys < 0 And chara.Text = "(" Then
         ' (Windows-Sonderzeichen)
```

```
        ys = (ys + 4096 + 61440)
        If fs = "Symbol" Or fs = "SymbolPS" Then
          Selection.Range.HighlightColorIndex = wdDarkYellow  ' dunkelgelb
          Symbolanzahl = Symbolanzahl + 1
        ElseIf fs = "Wingdings" Then
          Selection.Range.HighlightColorIndex = wdTeal  ' Mittelblau
          Wingdingsanzahl = Wingdingsanzahl + 1
        Else
          ' sonstige Sonderschriftart:
          Selection.Range.HighlightColorIndex = wdPink   ' dunkles Rosa
          Sonstigesanzahl = Sonstigesanzahl + 1
        End If
        '''''''''
        If InStr(strSonderzeichenliste, fs) = 0 Then
          strSonderzeichenliste = strSonderzeichenliste & ";" & fs
        End If
      '''   Else
        '''     '' MsgBox "Achtung, unerkannte Schriftart!"
        '''       fs = "???"
        '''       Selection.Range.HighlightColorIndex = wdRed   ' rot
        '''       Unerkanntesanzahl = Unerkanntesanzahl + 1
        '''     ' ((färbt mir fälschlich alle "Klammer-Auf"-Zeichen!))
      End If
    End If
  Next chara
' *************************
' kombinierte Zeichen suchen und rot markieren, wenn es Windows-Sonderschriftarten sind
' (diese zusätzliche Abfrage ist nicht bei Word 2000, aber ab Word 2003 nötig):
' *************************
  Rng1.Select
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "?[" & ChrW(768) & "-" & ChrW(879) & "]"
    .Replacement.Text = ""
    .Highlight = False
    .Forward = True
    .MatchWildcards = True  ' ist hier unbedingt nötig!!!
    Do While .Execute      ' (tue, solange etwas gefunden wird)
      If Dialogs(wdDialogInsertSymbol).Font <> "(normaler Text)" Then
          Selection.Range.HighlightColorIndex = wdRed
          Unerkanntesanzahl = Unerkanntesanzahl + 1
      End If
    Loop
  End With
' ************************************
' und nun Windows-Sonderzeichen mit 2 diakritischen Zeichen (für Word 2003 aufwärts):
''   Selection.HomeKey Unit:=wdStory
  Rng1.Select
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "?[" & ChrW(768) & "-" & ChrW(879) & "][" & ChrW(768) & "-" & ChrW(879) & "]"
    .Replacement.Text = ""
    .Highlight = False
    .Forward = True
    .MatchWildcards = True  ' ist hier unbedingt nötig!!!
    Do While .Execute      ' (tue, solange etwas gefunden wird)
      If Dialogs(wdDialogInsertSymbol).Font <> "(normaler Text)" Then
          Selection.Range.HighlightColorIndex = wdRed
          Unerkanntesanzahl = Unerkanntesanzahl + 1
      End If
```

```
    Loop
  End With
'  ******************
  ' Restliche (unzulässige?) Zeichen in der Private-Use-Area rot markieren,
  ' aber keine normale "Klammer auf" wie in der Schleife oben:
  Selection.HomeKey Unit:=wdStory
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "[" & ChrW(61473) & "-" & ChrW(61695) & "]"
    .Replacement.Text = ""
    .Highlight = False
    .Forward = True
    .MatchWildcards = True  ' ist hier unbedingt nötig!!!
    Do While .Execute      ' (tue, solange etwas gefunden wird)
      fs = Dialogs(wdDialogInsertSymbol).Font
      '  Abfrage, ob Symbol, Wingdings oder sonstige Sonderschrift:
      '  Wenn Schriftart unklar, Fragezeichen setzen und rot markieren.
      If fs = "(normaler Text)" Or fs = "" Then
        '' MsgBox "Achtung!"
        fs = "???"
        Selection.Range.HighlightColorIndex = wdRed   ' rot
        Unerkanntesanzahl = Unerkanntesanzahl + 1
      End If
      '''''''''
      If InStr(strSonderzeichenliste, fs) = 0 Then
        strSonderzeichenliste = strSonderzeichenliste & ";" & fs
      End If
      '''''''''
    Loop
  End With
End Sub



Private Function Fkt_Sonderschriftarten_nachbearbeiten()
' H. Chris Gast, 3 bis. 7. April 2014, 10. April 2014, Stand 23. April 2014.
' Sonderschriftartenliste sortieren und Ergebnisprotokoll erstellen:
Dim astrFontList() As String
Dim Warnung2 As String
Dim strSonstige_Sonderschriften As String
'  ****************
'  Warnung vor unerkannten Zeichen (mit Sonderschriftarten):
  If Unerkanntesanzahl > 0 Then
    Warnung2 = vbCrLf & vbCrLf & "Achtung, es gibt " & Unerkanntesanzahl _
          & " Textstellen mit einer unbekannten Sonderschriftart," _
          & " die hier rot markiert ist!"
    If strSonderzeichenliste = "" Then
      Fkt_Sonderschriftarten_nachbearbeiten = Warnung2
      Exit Function
    End If
  Else
    Warnung2 = ""
  End If
'  ***************
'  Gibt es überhaupt (erkannte) Windows-Sonderschriftarten?
  If strSonderzeichenliste = "" Then
      Fkt_Sonderschriftarten_nachbearbeiten = vbCrLf & vbCrLf _
      & "Das Dokument enthält keine Windows-Sonderschriftarten." _
      & Warnung2
    Exit Function
  End If
```

```
' ****************
' Es gibt erkannte Windows-Sonderschriftarten:
' Überflüssige Trennzeichen beseitigen:
  If Left(strSonderzeichenliste, 1) = ";" Then
    strSonderzeichenliste = Mid(strSonderzeichenliste, 2)
  End If
' alphabetisch Sortieren:
  astrFontList = Split(strSonderzeichenliste, ";")
  WordBasic.SortArray astrFontList()
  strSonderzeichenliste = VBA.Join(astrFontList(), vbLf)
' ****************
' Sonstige Sonderschriftarten nur auflisten, wenn > 0:
  If Sonstigesanzahl > 0 Then
   strSonstige_Sonderschriften _
       = vbCrLf & "und " & Sonstigesanzahl & " sonstige Windows-Sonderzeichen."
  Else
   strSonstige_Sonderschriften = "."
  End If
' ****************
' Ergebnisprotokoll erstellen:
  Fkt_Sonderschriftarten_nachbearbeiten = vbCrLf & vbCrLf _
       & "Das Dokument enthält folgende Sonderschriftarten: " & vbCrLf _
       & strSonderzeichenliste _
       & vbCrLf & vbCrLf & "Häufigkeit der Sonderschriftarten: " & vbCrLf _
       & Symbolanzahl & " Symbol-Zeichen, " _
       & Wingdingsanzahl & " Wingdings-Zeichen" _
       & strSonstige_Sonderschriften _
       & Warnung2
End Function


Private Sub Farbmarkierungen_beseitigen()
' H. Chris Gast, 6. April 2012, neu 7. April 2013.
' Suche in allen Dokument-Komponenten nach Farbmarkierungen (Highlights),
' jetzt sogar in Textfeldern von Kopf- bzw. Fußzeilen.
' Jede Dokumentkomponente soll nur ein einziges Mal aufgesucht werden.
' ********************
Dim rngStory As Range
Dim rngSection As Section
Dim rngHeaderFooter As HeaderFooter
Dim shpShape As Shape        ' Textfelder (und anderes)
Dim BL_Textfeld As Boolean   ' Suche nach Textfeldern in Kopf- oder Fußzeile
' ***************************************************************************
' Alle (wirklich alle) Dokumentkomponenten durchsuchen, aber nur je einmal:
' ***************************************************************************
' Alle Dokumentkomponenten außer Fuß- und Kopfzeilen:
  For Each rngStory In ActiveDocument.StoryRanges
    If rngStory.StoryType < 6 Then
      '(die Trennlinie über Fußnoten soll nicht erfasst werden)
      rngStory.HighlightColorIndex = wdNoHighlight
      rngStory.Font.Shading.BackgroundPatternColor = wdColorAutomatic
    End If
  Next rngStory
' ***********************************************
' Kopf und Fußzeilen (in allen Sections):
  For Each rngSection In ActiveDocument.Sections
    ' Kopfzeilen:
    For Each rngHeaderFooter In rngSection.Headers
      If rngHeaderFooter.LinkToPrevious = False Then
        rngHeaderFooter.Range.HighlightColorIndex = wdNoHighlight
        rngHeaderFooter.Range.Font.Shading.BackgroundPatternColor = wdColorAutomatic
```

```
        '''''''''''''''''''''
        ' Suche nach Textfeldern in Kopf-/Fußzeilen nur ein einziges Mal durchführen!
        ' (sie werden sonst bei jedem rngHeaderFooter gefunden!)
        If BL_Textfeld = False Then
          For Each shpShape In rngHeaderFooter.Shapes
            If shpShape.Type = msoTextBox Then
              shpShape.TextFrame.TextRange.HighlightColorIndex = wdNoHighlight
              shpShape.TextFrame.TextRange.Font.Shading.BackgroundPatternColor _
                    = wdColorAutomatic
            End If
          Next
          BL_Textfeld = True
        End If
        '''''''''''''''''''''
      End If
    Next rngHeaderFooter
    ' **********
    ' Fußzeilen:
    For Each rngHeaderFooter In rngSection.Footers
      If rngHeaderFooter.LinkToPrevious = False Then
        rngHeaderFooter.Range.HighlightColorIndex = wdNoHighlight
        rngHeaderFooter.Range.Font.Shading.BackgroundPatternColor = wdColorAutomatic
      End If
    Next rngHeaderFooter
  Next rngSection
End Sub
```

20 Prüfung der gesamten Datei auf Darstellbarkeit mit "Cambria"

Es ist unangenehm, wenn in einer gedruckten Broschüre oder in einem frisch gedruckten Buch irgendwo ein Fragezeichen oder ein "leeres Kästchen"[13] statt der gewünschten Sonderzeichen stehen. Beim Korrekturlesen übersieht man so etwas leicht, wenn sehr viele Seiten zu prüfen sind.

Das folgende Makro prüft das ganze Dokument und färbt alle Zeichen ein, die nicht mit Cambria oder Cambria Math darstellbar sind. Mit der Suchfunktion von Word kann man recht leicht alle Farbmarkierungen finden.

Ferner machten vor einigen Jahren einige Steuerzeichen für Schreibrichtungswechsel Ärger bei der Konvertierung in Pdf mittel des damaligen Adobe-Acrobats. Deshalb lasse ich das Makro auch diese einfärben.

"Courier New" ist eine nichtproportionale Schriftart und soll nicht in Cambria umgewandelt werden. Der Anwender des Makro kann bei Bedarf das Makro anpassen und andere Schriftarten von der Konvertierung ausnehmen.

Mit dem folgenden Makro werden alle Zeichen geprüft, ob sie mit Cambria oder Cambria-Math darstellbar sind. Nicht-darstellbare Zeichen werden rot markiert und gezählt. Durch eine Unterprozedur werden auch alle Zeichen der Windows-Sonderschriftarten Symbol, Wingdings usw. erfasst, dunkelrosa gefärbt und mitgezählt.

— Es werden alle mit Cambria **nicht**-darstellbare Unicode-Zeichen rot markiert, auch wenn kein Kästchen oder Fragezeichen zu sehen ist, sondern ein Zeichen in einer anderen Schriftart wie Arial Unicode.
— Windows-Sonderschriftarten wie Symbol und Wingdings werden dunkelrosa markiert.
— Steuerzeichen für die Schreibrichtung werden lila markiert, bzw. in Tabellenfeldern werden die Tabellenfelder rosa schattiert.
— "Courier New" wird gelb markiert.

Der Rest der Zeichen wird in Cambria (bzw. Cambria Math) umgesetzt. Es wird hierbei nicht geprüft, ob andere Unicode-Fonts als Cambria verwendet worden waren.

Zeichen höherer Unicode-Ebenen können mit der Abfrage der Glyph-Darstellbarkeit mittels API nicht geprüft werden. Da aber bei Cambria Math die darstellbaren Zeichen der höheren Unicode-Ebene 1 lückenlos zusammenhängen, sind sie durch eine einfache Abfrage erfassbar.

Will man Zeichen der Sonderschriftarten "Symbol" und "Wingdings" durch Unicode-Zeichen ersetzen, kann man das Makro aus folgendem Aufsatz verwenden.
Beschreibung: www.siebener-kurier.de/chris-aufsaetze/Unicode_Word-Makros.pdf
Download der Makros: http://www.siebener-kurier.de/chris-aufsaetze/Unicode-Makros.zip

```
Option Explicit

Private Const TWIPSPERINCH = 1440
Private Const HORZRES = 8
Private Const VERTRES = 10
Private Const LOGPIXELSX = 88
Private Const LOGPIXELSY = 90
'
#If Win64 Then
```

[13] Leere Kästchen (*en: Blank box, blank place-holder, blank rectangle, box*) sind Platzhalter, die angezeigt werden, wenn die gewählte Schriftart (Font) das Zeichen nicht darstellen kann.

```
  Private Declare PtrSafe Function GetDC Lib "user32" (ByVal hwnd As LongPtr) As LongPtr

  Private Declare PtrSafe Function GetWindowDC Lib "user32" ( _
    ByVal hwnd As LongPtr) As LongPtr

  Private Declare PtrSafe Function CreateCompatibleDC Lib "gdi32" ( _
    ByVal hDC As LongPtr) As LongPtr

  Private Declare PtrSafe Function ReleaseDC Lib "user32" ( _
    ByVal hwnd As LongPtr, ByVal hDC As LongPtr) As Long

  Private Declare PtrSafe Function DeleteDC Lib "gdi32" (ByVal hDC As LongPtr) As Long

  Private Declare PtrSafe Function GetDeviceCaps Lib "gdi32" ( _
    ByVal hDC As LongPtr, ByVal nIndex As Long) As Long
#Else

  Private Declare Function GetDC Lib "user32" (ByVal hwnd As Long) As Long

  Private Declare Function GetWindowDC Lib "user32" (ByVal hwnd As Long) As Long

  Private Declare Function CreateCompatibleDC Lib "gdi32" (ByVal hDC As Long) As Long

  Private Declare Function ReleaseDC Lib "user32" ( _
    ByVal hwnd As Long, ByVal hDC As Long) As Long

  Private Declare Function DeleteDC Lib "gdi32" (ByVal hDC As Long) As Long

  Private Declare Function GetDeviceCaps Lib "gdi32" ( _
    ByVal hDC As Long, ByVal nIndex As Long) As Long
#End If
'
'used with fnWeight
Private Const FW_DONTCARE = 0
Private Const FW_THIN = 100
Private Const FW_EXTRALIGHT = 200
Private Const FW_LIGHT = 300
Private Const FW_NORMAL = 400
Private Const FW_MEDIUM = 500
Private Const FW_SEMIBOLD = 600
Private Const FW_BOLD = 700
Private Const FW_EXTRABOLD = 800
Private Const FW_HEAVY = 900
Private Const FW_BLACK = FW_HEAVY
Private Const FW_DEMIBOLD = FW_SEMIBOLD
Private Const FW_REGULAR = FW_NORMAL
Private Const FW_ULTRABOLD = FW_EXTRABOLD
Private Const FW_ULTRALIGHT = FW_EXTRALIGHT
'
'used with fdwCharSet
Private Const ANSI_CHARSET = 0
```

```
 Private Const DEFAULT_CHARSET = 1
 Private Const SYMBOL_CHARSET = 2
 Private Const SHIFTJIS_CHARSET = 128
 Private Const HANGEUL_CHARSET = 129
 Private Const CHINESEBIG5_CHARSET = 136
 Private Const OEM_CHARSET = 255
'
'used with fdwOutputPrecision
 Private Const OUT_CHARACTER_PRECIS = 2
 Private Const OUT_DEFAULT_PRECIS = 0
 Private Const OUT_DEVICE_PRECIS = 5
'
'used with fdwClipPrecision
 Private Const CLIP_DEFAULT_PRECIS = 0
 Private Const CLIP_CHARACTER_PRECIS = 1
 Private Const CLIP_STROKE_PRECIS = 2
'
'used with fdwQuality
 Private Const DEFAULT_QUALITY = 0
 Private Const DRAFT_QUALITY = 1
 Private Const PROOF_QUALITY = 2
'
'used with fdwPitchAndFamily
 Private Const DEFAULT_PITCH = 0
 Private Const FIXED_PITCH = 1
 Private Const VARIABLE_PITCH = 2
'
'
#If Win64 Then

   Private Declare PtrSafe Function MulDiv Lib "kernel32" ( _
      ByVal nNumber As Long, ByVal nNumerator As Long, _
      ByVal nDenominator As Long) As Long


   Private Declare PtrSafe Function CreateFont Lib "gdi32" Alias "CreateFontA" ( _
      ByVal H As Long, ByVal W As Long, ByVal E As Long, ByVal O As Long, _
      ByVal W As Long, ByVal i As Long, ByVal u As Long, ByVal S As Long, _
      ByVal c As Long, ByVal OP As Long, ByVal CP As Long, ByVal Q As Long, _
      ByVal PAF As Long, ByVal F As String) As LongPtr


   Private Declare PtrSafe Function SelectObject Lib "gdi32" ( _
      ByVal hDC As LongPtr, ByVal hObject As LongPtr) As LongPtr


   Private Declare PtrSafe Function DeleteObject Lib "gdi32" ( _
      ByVal hObject As LongPtr) As Long
 #Else


   Private Declare Function MulDiv Lib "kernel32" ( _
      ByVal nNumber As Long, ByVal nNumerator As Long, _
      ByVal nDenominator As Long) As Long


   Private Declare Function CreateFont Lib "gdi32" Alias "CreateFontA" ( _
      ByVal nHeight As Long, ByVal nWidth As Long, ByVal nEscapement As Long, _
      ByVal nOrientation As Long, ByVal fnWeight As Long, _
      ByVal fdwItalic As Boolean, _
      ByVal fdwUnderline As Boolean, ByVal fdwStrikeOut As Boolean, _
      ByVal fdwCharSet As Long, _
```

```
      ByVal fdwOutputPrecision As Long, ByVal fdwClipPrecision As Long, _
      ByVal fdwQuality As Long, ByVal fdwPitchAndFamily As Long, _
      ByVal lpszFace As String) As Long


   Private Declare Function SelectObject Lib "gdi32" ( _
      ByVal hDC As Long, ByVal hObject As Long) As Long


   Private Declare Function DeleteObject Lib "gdi32" (ByVal hObject As Long) As Long
 #End If
'
'!!F-
 Public Type WCRANGE
   wcLow As Integer   'Low Unicode code point in the range of supported Unicode code points.
   cGlyphs As Integer 'Number of supported Unicode code points in this range.
 End Type
'
Public Type GLYPHSET
   cbThis As Long          'The size, in bytes, of this structure.
   flAccel As Long         'Flags describing the maximum size of the glyph indices.
   cGlyphsSupported As Long 'The total number of Unicode code points supported in the font.
   cRanges As Long         'The total number of Unicode ranges in ranges.
   aRANGE() As WCRANGE     'Array of Unicode ranges that are supported in the font.
End Type
'!!F+
'
Private Const GGI_MARK_NONEXISTING_GLYPHS As Long = &H1
Private Const GDI_ERROR As Long = &HFFFFFFFF
Private Const INVALIDGLYPHINDEX As Integer = &HFFFF
'
'
#If Win64 Then

   Private Declare PtrSafe Function GetFontUnicodeRanges Lib "gdi32.dll" ( _
      ByVal hDC As LongPtr, lpGS As Any) As Long


   Private Declare PtrSafe Sub CopyMemory Lib "kernel32" Alias "RtlMoveMemory" ( _
      lpDest As Any, lpSource As Any, ByVal nCount As Long)

#Else


   Private Declare Function GetFontUnicodeRanges Lib "gdi32.dll" ( _
      ByVal hDC As Long, lpGS As Any) As Long
''   Private Declare Function GetGlyphIndices Lib "gdi32.dll" Alias "GetGlyphIndicesA" ( _
''      ByVal hDC As Long, ByVal lpStr As String, ByVal lpStrLen As Long, _
''      ByVal pGI As Long, ByVal Flags As Long) As Long


   Private Declare Sub CopyMemory Lib "kernel32" Alias "RtlMoveMemory" ( _
      lpDest As Any, lpSource As Any, ByVal nCount As Long)
#End If
```

```
Sub Cambria_Unzulaessige_Zeichen_färben_Ganze_Datei()
' H. Chris Gast, 19. bis 24. November 2015, 1. + 8. + 9. Dezember 2015. 12. +13. Januar 2017.
' Prüfen, ob alle Zeichen des Dokuments mit Cambria bzw. mit Cambria Math darstellbar sind.
'
' Unicode-Zeichen, die nicht mit Cambria oder Cambria Math darstellbar sind, werden rot markiert.
' Windows-Sonderfonts wie Symbol und Wingdings werden in einer Unterprodur
' erfasst, separat gezählt und dunkelrosa gefärbt.
' Zeichen der Schriftart "Courier New" werden gelb markiert,
' Zeichen der Schriftart "BSsymb9" und BSsymb10" dunkelgelb.
' Bidirektionale Steuerzeichen werden violett markiert,
' bzw. in Tabellenfeldern werden diese rosa schattiert.
'
' Funktioniert in Word 2000 unter Windows XP und in Word 2010 unter Windows 7.
' Ermittlung des Data-Array aus Excel-Makro von Andreas Killer, Jan. 2014,
' aber für Word 2000 angepasst.
' http://answers.microsoft.com/en-us/office/forum/office_2010-customize/
' check-if-a-unicode-character-is-available-in-the/492757a5-b965-4f91-b340-855725cd8307?auth=1
'
' Zeichen höherer Unicode-Ebenen werden anhand der Cambria-Math-Liste
' aus Fileformat.Info geprüft, weil hier die API-Funktion bei mir nicht funktionierte.
'
' Windows-Sonderfonts wie Symbol und Wingdings werden in einer Unterprodur
' erfasst und separat gezählt und dunkelrosa gefärbt.
'
'  Alle Zeichen, die ok sind, werden in Cambria umgewandelt.
'
' Rechenzeit ca. 1 Sekunde pro Seite.
' ****************
Dim GS1 As GLYPHSET  ' für Cambria
Dim GS2 As GLYPHSET  ' für Cambria Math
Dim i As Long        ' Zeichen-Nr. in der Markierung
Dim k As Integer     ' Anzahl Symbolzeichen und Zeichen in der Private Use Area
Dim Data(31 To 65535, 1 To 6)
Dim strFontname1 As String
Dim strFontname2 As String
Dim jF As Integer     ' Anzahl der nicht-darstellbaren Zeichen
Dim jS As Integer     ' Anzahl der Symbol- und Wingdings-Zeichen
Dim jsur As Integer   ' Anzahl der gefunden Cambria_Math-Zeichen höherer Ebenen
Dim Dok1 As Word.Document
Dim Dok2 As Word.Document  ' unsichtbare temporäre Zwischendatei
Dim strFund As String
Dim aRange As Range
Dim Rng2 As Range
Dim Rng3 As Range
Dim IntLänge As Long  ' Länge der Markierung
Dim LngCode  As Long  ' Dezimalcode des jeweiligen Zeichens
Dim LngSur1 As Long   ' Dezimalcode des eventuellen linken Surrogats
Dim LngSur2 As Long   ' Dezimalcode des eventuellen rechten Surrogats
Dim tt As Single      ' Zeit für Warteschleifen
Dim kstory As Integer
Dim strWarnung_Bidi As String
'  ************************************
On Error GoTo Zeile_Fehler
'  ***********************************************************************
'  Zeitmessen für die Makro-Dauer in Sekunden
  Dim Zeit1 As Single    '  "Single" = Zahlen mit Dezimalen
  Dim Zeit2 As Single
  Dim Zeitdauer As Single
  Zeit1 = Timer
'  ***********************************************************************
'  Unsichtbares Zwischen-Dokument erstellen für Schriftartzuweisung
  '(dies Vermeidet Probleme, wenn Cursor gerade auf Zeichen steht, die nicht mit Cambria
```

```
  ' darstellbar sind):
  Set Dok1 = ActiveDocument
  WordBasic.DisableAutoMacros 1   ' Disables auto macros
  Set Dok2 = Documents.Add(DocumentType:=wdNewBlankDocument, Visible:=False)
  Dok2.Activate
  Selection.WholeStory
'  Get the font:
  strFontname1 = "Cambria"
  strFontname2 = "Cambria Math"
  ' ***********************************
  ' Fonts vorübergehend der Markierung zuweisen in Zwischendokument(Mindestlänge nötig):
  ' ***
  If Len(Selection) < 2 Then Selection.TypeText Text:="Testtext fuer Font."
   Selection.WholeStory
  ' Ersten Font zuweisen:
  Selection.Range.Font.Name = strFontname1
  ' **************
  ' Warteschleife:
  tt = Timer
  Do While Timer < tt + 0.1  'Sekunden
    DoEvents
  Loop
  ' **************
  If Selection.Range.Font.NameAscii <> strFontname1 Then
    MsgBox "Einlesen misslungen!"
    Exit Sub
  End If
  ' Einlesen in Glyphset 1:
  GS1 = GetGLYPHSET(Selection.Range.Font)
'  ************************************
  ' zweiten Font zuweisen:
  Selection.Range.Font.Name = strFontname2
  ' **************
  ' Warteschleife:
   tt = Timer
   Do While Timer < tt + 0.1  'Sekunden
      DoEvents
   Loop
  ' **************
  If Selection.Range.Font.NameAscii <> strFontname2 Then
    MsgBox "Einlesen misslungen!"
    Exit Sub
  End If
  ' Einlesen in Glyphset 2:
  GS2 = GetGLYPHSET(Selection.Range.Font)
'  ************************************
'  Zwischendatei schließen und zu untersuchendes Dokument wieder aktivieren:
  Dok2.Close (wdDoNotSaveChanges)
  Dok1.Activate
'  ***********************************************************************
'  Create the output
  Dim nbeide As Integer
  Dim nja    As Integer
  Dim nok As Integer
  For i = 32 To UBound(Data)
    Data(i, 1) = i
    Data(i, 2) = "U+" & Right$("0000" & Hex(i), 4)
    Data(i, 3) = ChrW$(i)
    Data(i, 4) = IIf(HuntGLYPHSET(i, GS1), "jaC", "--")
    Data(i, 5) = IIf(HuntGLYPHSET(i, GS2), "jaM", "--")
    If Data(i, 4) = "jaC" And Data(i, 5) = "jaM" Then
      Data(i, 6) = "ja12!"
```

```
      nbeide = nbeide + 1
    ElseIf Data(i, 4) = "jaC" Or Data(i, 5) = "jaM" Then
      Data(i, 6) = "ok"
      nok = nok + 1
    Else
      Data(i, 6) = "----"
    End If
  Next i
  '''''''''''''''''''''''''
  '  Vergeblicher Test für höhere Unicodes:
  ''   MsgBox "Zeichen 119807 " & HuntGLYPHSET(119807, GS2) ' eigentlich nicht darstellbar
  ''   MsgBox "Zeichen 119808 " & HuntGLYPHSET(119808, GS2) ' eigentlich darstellbar
  ''   MsgBox "Zeichen 119819 " & HuntGLYPHSET(120819, GS2) ' eigentlich darstellbar
  ''   MsgBox "Zeichen 119829 " & HuntGLYPHSET(120820, GS2) ' eigentlich darstellbar
  ' ergibt immmer falsch! Dagegen
  ''     MsgBox "Zeichen 228 " & HuntGLYPHSET(228, GS2)
  ' ergibt wahr
' ***********************************************************************
' ***********************************************************************
' H. Chris Gast, 19.11.2015 bis 24.11.2015.
'  Änderungsverfolgung abstellen:
  Selection.NoProofing = True
  Application.CheckLanguage = False
'  ***************************
'  Rechenzeit beschleunigen, Bildschirmaktualisierung abstellen:
  StatusBar = "Bitte warten - das Makro läuft."          '  Geht nicht immer. .
'  Problem:  Die StatusBar-Anzeige wird sehr schnell wieder überschrieben.
  ''Application.DisplayStatusBar = False              ' geht nicht in Word 2010.
  Application.ScreenUpdating = False
  Options.Pagination = False
'  ******************************
'  Scrollbars ausblenden:
  With ActiveWindow
    .DisplayHorizontalScrollBar = False
    .DisplayVerticalScrollBar = False
  End With
'  ******************************
'  Warnung vor Bidirektionalen Steuerzeichen:
  strWarnung_Bidi = ""
'  ************************************
'' Jedes Zeichen der Markierung prüfen:
''''''''''''''''''''''''''''''''''''''''''''''''
'  Das ganze Dokument prüfen (ohne Kopf- und Fußzeilen)!
'''  Hauptkomponente + Fußnoten + Textfelder erfassen:
  For kstory = 1 To 5
     On Error Resume Next
     ActiveDocument.StoryRanges(kstory).Select
     If Err.Number > 0 Then
       'MsgBox "StoryRange " & kstory & " gibt es nicht!"
       Err.Clear
     Else
       On Error GoTo Zeile_Fehler
       ''''''''''''''''''''''''''''''''''''''''''''''''
    '  Die StoryRange markieren:
       Set aRange = Selection.Range
       Set Rng2 = Selection.Range
       Set Rng3 = Selection.Range
    '  Alle Farbmarkierungen beseitigen:
       Selection.Range.HighlightColorIndex = wdAuto
    '  Windows-Sonderschriftarten wie Symbol und Wingdings werden in einem
    '  separaten Makro dunkelrosa gefärbt und gezählt:
       jS = jS + Fkt_Windows_Sonderschrift(Selection.Range)
```

```
  aRange.Select
'  ******************************************************************
  IntLänge = Len(aRange)
  For i = 1 To IntLänge
    "If i Mod (100) = 0 Then
    "   StatusBar = "Das Makro ist gerade bei dem Wert " & i & "."
    "End If
    Rng2.SetRange Start:=aRange.Start + i - 1, _
                  End:=aRange.Start + i
    If Rng2.HighlightColorIndex = wdPink Then
      ' ist bereits als Symbol-Zeichen erfasst
      "MsgBox "hier dunkelrosa!"
    Else
      '  ****************************************************
      ' Dezimaler Code bestimmen:
      '"If AscW(Mid(aRange, i, 1)) > 255 Then
      '"LngCode = Fkt_AscXL(Mid(aRange, i, 1)) ' geht so nicht!
      ' Da der Code bei "AscW(Zeichen)" in VBA als Datentyp "Integer" ausgegeben wird,
      ' der nur bis 32767 positive Zahlenwerte ermöglicht, waren einige "Verrenkungen"
      ' in der Programmierung nötig. Ich ermittelte zuerst die hexadezimalen Werte
      ' mit "Hex(AscW(Zeichen)" und dann mit der Val-Funktion die dezimalen Werte.
      On Error Resume Next
        LngCode = Val("&H" & Hex(AscW(Rng2.Text)) & "&")
      If Err.Number > 0 Then
        LngCode = 0
        Err.Clear
      End If
      On Error GoTo Zeile_Fehler:
      ' Ein Code über 65300 wird in Word unter Windows nie direkt gefunden,
      ' da dann Surrogate verwendet werden:
      If LngCode > 55295 And LngCode < 57344 Then
      '  Zeichen höherer Unicode-Ebene (= Surrogat), nach Aufsatz 115 (Makro "Zeichenanalyse"):
        Rng2.Select
        Rng3.SetRange Start:=Rng2.Start + 1, _
                      End:=Rng2.End + 1
        ' Zeichen höherer Ebene. Suche nach den Surrogaten:
        If LngCode > 55295 And LngCode < 56320 Then
          LngSur1 = LngCode
          LngSur2 = Val("&H" & Hex(AscW(Rng3.Text)) & "&")
          If LngSur2 > 56319 And LngSur2 < 57344 Then
            ' Wenn ja, liegt echtes Zeichen höherer Unicode-Ebene vor:
            ' Berechnung des Unicodes höherer Ebenen nach ISO 10646:2003, Anhang C:
            " Y(0) = ((Y(j) - 55296) * 1024) + (Y(j + 1) - 56320) + 65536
            '  z(0) ist der Unicode des gesuchten Zeichen höherer Unicode-Ebene.
            ' Muster: Der Unicode des Zeichens höherer Ebene ist: 120820,
            ' (der Unicode des Surrogat 1 ist: 55349,
            '  der Unicode des Surrogat 2 ist: 57332),
            LngCode = ((LngSur1 - 55296) * 1024) + (LngSur2 - 56320) + 65536
            If LngCode >= 119808 And LngCode <= 120831 Then
              Selection.Range.HighlightColorIndex = wdWhite
              "MsgBox LngCode & " ist hier der Unicode höherer Ebene (nicht direkt prüfbar)!"
              jsur = jsur + 1
            Else
              Selection.Range.HighlightColorIndex = wdRed
              jF = jF + 1
            End If
            i = i + 1
          Else
            MsgBox "unvollständiges Surrogat " & LngCode
            Selection.Range.HighlightColorIndex = wdRed
           " Selection.Range.Font.ColorIndex = wdDarkBlue
            jF = jF + 1
```

```
          End If
        Else
           MsgBox "unvollständiges Surrogat " & LngCode
           Selection.Range.HighlightColorIndex = wdRed
           "Selection.Range.Font.ColorIndex = wdDarkBlue
           jF = jF + 1
        End If
      '  ******************
      "ElseIf LngCode >= 32 Then
      "ElseIf LngCode > 255 Then
      ElseIf LngCode > 880 Then
      '  Normales Zeichen der Ebene 0,
      '  von dez. 32 bis dez. 880 gibt es keine Zeichen, die bei Cambria fehlen:
        strFund = Data(LngCode, 6)
        If strFund <> "----" Then
          " MsgBox "Zeichen '" & Mid(aRange, i, 1) & "' ist
          " in Cambria bzw. Cambria_Math darstellbar."
        Else
          Rng2.Select
          "If Selection.Range.HighlightColorIndex <> wdPink Then
          jF = jF + 1
          Selection.Range.HighlightColorIndex = wdRed
          "End If
        End If
      End If
    End If
  Next i
' ******************************
If strWarnung_Bidi = "" Then
  strWarnung_Bidi = Fkt_Bidirektionale_Steuerzeichen_markieren(aRange)
Else
  Debug.Print Fkt_Bidirektionale_Steuerzeichen_markieren(aRange)
End If
' ******************************
 ' "Courier New" gelb markieren:
 aRange.Find.ClearFormatting
 aRange.Find.Replacement.ClearFormatting
 With aRange.Find
   .Text = ""
   .Replacement.Text = ""
   .Font.Name = "Courier New"
   .Replacement.Highlight = wdYellow
   .Forward = True
   .Wrap = wdFindContinue
   .Format = True
   .MatchCase = False
   .MatchWholeWord = False
   .MatchWildcards = False
   .MatchSoundsLike = False
   .MatchAllWordForms = False
 End With
 aRange.Find.Execute Replace:=wdReplaceAll
' ***********************************
 ' Überall Cambria zuordnen, soweit nicht Farbmarkierung vorliegt:
 aRange.Find.ClearFormatting
 aRange.Find.Replacement.ClearFormatting
 With aRange.Find
   .Text = ""
   .Highlight = False
   .Replacement.Text = ""
   .Replacement.Font.Name = "Cambria"
   .Forward = True
```

```
        .Wrap = wdFindContinue
        .Format = True
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = False
        .MatchSoundsLike = False
        .MatchAllWordForms = False
      End With
      aRange.Find.Execute Replace:=wdReplaceAll
      '''''''''''''''''''''''''
    End If
    '''''''''''''''''''''''''''
  Next kstory
' ************************************
' Alles wieder einschalten:
  StatusBar = ""
  Options.Pagination = True
  Application.ScreenUpdating = True
  Application.DisplayStatusBar = True
' *******************
' Scrollbalken wieder einschalten:
  With ActiveWindow
    .DisplayHorizontalScrollBar = True
    .DisplayVerticalScrollBar = True
  End With
' In Hauptkomponente gehen:
  ActiveDocument.StoryRanges(1).Select
  Selection.Collapse
' zur ersten Fundstelle gehen:
  Selection.Find.ClearFormatting
  Selection.Find.Highlight = True
  With Selection.Find
    .Text = ""
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindStop   'wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute
' ********************************************************************
' Auswertung:
' ********************************************************************
' Zeitmessen für die Makro-Dauer, Variante mit 1 Nachkommastelle.
  Zeit2 = Timer
  Zeitdauer = Format((Zeit2 - Zeit1), "##,##0.0")
  ''Zeitdauer = Round(Zeitdauer, 1)
' ******************
  If jF > 0 Or jS > 0 Then
    MsgBox "Anzahl mit Cambria/Cambria Math usw. nicht darstellbarer Zeichen (rot) = " & jF _
    & vbCrLf & vbCrLf _
    & "Anzahl der Zeichen mit unzulässigen Fonts Symbol oder Wingdings usw. (dunkelrosa) = " _
    & jS _
    & strWarnung_Bidi _
    & vbCrLf & vbCrLf _
    & "Die Rechenzeit war " & Zeitdauer & " Sekunden."
  Else
    MsgBox "Das Dokument ist vollständig mit Cambria und Cambria Math darstellbar. " _
```

```
      & vbCrLf _
      & "Ggf. ist das ganze Dokument zu markieren und ihm diese Schriftart zuzuweisen." _
      & strWarnung_Bidi _
      & vbCrLf & vbCrLf _
      & "Die Rechenzeit war " & Zeitdauer & " Sekunden."
   End If
'  ******************************************************************
Exit Sub
'  ********************
Zeile_Fehler:
   Application.Visible = True
   WordBasic.DisableAutoMacros 0
   On Error Resume Next
   ' Dok2.Close (wdDoNotSaveChanges)
   Dok1.Activate
   MsgBox "Fehler in Sub Fehlerbehandlung" & vbCrLf & "Fehlernummer: " & Err.Number & _
        vbCrLf & "Fehlerbeschreibung: " & Err.Description, vbCritical
   '  Für die Fehlersuche hier testweise die Fehlerstelle wieder aufsuchen:
  ''Resume
End Sub
' -----------------------------------------------------------------------------------------------



Private Function GetGLYPHSET(ByVal F As Font) As GLYPHSET
   'http://msdn.microsoft.com/en-us/library/windows/desktop/dd144887%28v=vs.85%29.aspx
   Dim Data() As Byte, i As Long, j As Long
   Dim Result As Long, Count As Long
   Dim GS As GLYPHSET
   Dim hDC As Long
'
  'Some of the propertys might not exists in the Font object
   On Error Resume Next
   Dim fSize As Integer, fBold As Integer
   Dim FName As String
   Dim fItalic As Boolean, fUnderline As Boolean, fStrikethrough As Boolean
   Dim hFont As Long, hOldFont As Long
   fSize = -MulDiv(F.Size, GetDeviceCaps(hDC, LOGPIXELSY), 72)
   FName = F.Name
   fBold = IIf(F.Bold, FW_BOLD, FW_NORMAL)
   fItalic = F.Italic
   fUnderline = F.Underline '<> xlUnderlineStyleNone   ' ging so nicht in Word 2000, 13.11.2015!
   ' Die Korrektur funktioniert aber auch in Word 2010 unter Windows 7.
   fStrikethrough = F.StrikeThrough
   On Error GoTo 0
'
  'Create a device context compatible to the screen
   hDC = CreateCompatibleDC(GetDC(0&))
   If hDC = 0 Then Exit Function
   'Create font object
   hFont = CreateFont(fSize, 0, 0, 0, fBold, fItalic, fUnderline, fStrikethrough, _
     DEFAULT_CHARSET, OUT_DEFAULT_PRECIS, CLIP_DEFAULT_PRECIS, _
     DEFAULT_QUALITY, DEFAULT_PITCH, FName)
   If hFont = 0 Then Exit Function
   'Load the font into the device context, storing the original font object
   hOldFont = SelectObject(hDC, hFont)
'
  'Get the size of the GLYPHSET structure required to store the information
   Result = GetFontUnicodeRanges(hDC, ByVal 0&)
   If Result = 0 Then Exit Function
   ReDim Data(0 To (Result - 1)) As Byte
   'Get the data and the number of bytes written
```

```
  Result = GetFontUnicodeRanges(hDC, Data(LBound(Data)))
  If Result = 0 Then Exit Function
'
  'Coyp the first 4 LONG into the GLYPHSET structure
  i = LBound(Data)
  Count = 4 * 4
  CopyMemory GS, Data(i), Count
  i = i + Count
  With GS
    'Did the font support unicode?
    If .cRanges > 0 Then
      'Create the array of ranges and copy the data into there
      ReDim .aRange(0 To (.cRanges - 1)) As WCRANGE
      j = LBound(.aRange)
      Count = .cRanges
      Do While Count > 0
        CopyMemory .aRange(j), Data(i), Len(.aRange(j))
        i = i + Len(.aRange(j))
        j = j + 1
        Count = Count - 1
      Loop
    End If
  End With
  'Destroy the created objects and return results
  DeleteObject hFont
  DeleteDC hDC
  GetGLYPHSET = GS
End Function
' ---------------------------------------------------------------------------------------------------------


Private Function HuntGLYPHSET(ByVal CharCode As Long, ByRef GS As GLYPHSET) _
      As Boolean
  'Hunt through the GLYPHSET, True if the Unicode character is supported
  Dim i As Long, Low As Long
  With GS
    CharCode = CharCode + 1
    For i = 0 To .cRanges - 1
      With .aRange(i)
        If .wcLow < 0 Then
          Low = .wcLow And &HFFFF&
        Else
          Low = .wcLow
        End If
        If CharCode > Low Then
          If CharCode <= Low + .cGlyphs Then
            HuntGLYPHSET = True
            Exit Function
          End If
        End If
      End With
    Next
  End With
End Function
' ---------------------------------------------------------------------------------------------------------



Private Function Fkt_Windows_Sonderschrift(Rng1 As Range)
' Dipl.-Ing. Hanna-Chris Gast, 3. bis 23. April 2014, Stand 24. November 2015.
' Neuer Ansatz übernommen aus altem Makro zur Ersetzung von Sonderzeichen
' von 2009 bis 2011.
```

```
' Funktioniert von Word 2000 bis Word 2010, in Windows XP und in Windows 7.
' Die gefundenen Sonderschrift-Zeichen werden farbig markiert.
' Findet auch Windows-Sonderzeichen, bei denen alle bisherigen Makros versagten.
'
'  Problem: Es gibt Fälle, in denen mit "Suchen" nach Zeichen im "PUA"-Bereich die Sonderschriftarten
'  nicht gefunden werden.
'  Einzige (etwas zeitraubende) Lösung:
'  Jedes Zeichen des ganzen Dokuments wird mittels "Einfügen-Symbol" danach abgefragt,
'  ob es ein Windows-Sonderzeichen ist:'
Dim chara As Variant
"Dim i As Integer
Dim k As Integer: k = 0
Dim fs As String
Dim ys As Long
Dim Dateiende As Boolean
'  ******************************************************************************************
'  Suche nach Windows-Sonderzeichen:
"   i = 0
  For Each chara In Rng1.Characters
    If chara.Text = "(" Or chara.Text Like "[" & ChrW(61472) & "-" & ChrW(61695) & "]" Then
      ' (hier kann eine Sonderschriftart vorliegen)
      chara.Select
      With Dialogs(wdDialogInsertSymbol)
        fs = .Font
        ys = .CharNum
      End With
      ' Schriftart (Font) bestimmen:
      If fs <> "(normaler Text)" And fs <> "" _
      Or ys < 0 And chara.Text = "(" Then
        ' (Windows-Sonderzeichen)
        ' Dez. Code = (ys + 4096 + 61440)
        Selection.Range.HighlightColorIndex = wdPink
        k = k + 1
      End If
    End If
  Next chara
'  **************************
'  kombinierte Zeichen suchen und rot markieren, wenn es Windows-Sonderschriftarten sind
'  (diese zusätzliche Abfrage ist ab Word 2003 nötig):
'  **************************
  Rng1.Select
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
   .Text = "?[" & ChrW(768) & "-" & ChrW(879) & "]"
   .Replacement.Text = ""
   .Highlight = False
   .Forward = True
   .MatchWildcards = True  ' ist hier unbedingt nötig!
    Do While .Execute      ' (tue, solange etwas gefunden wird)
      If Selection.InRange(Rng1) = False Then Exit Do
      If Dialogs(wdDialogInsertSymbol).Font <> "(normaler Text)" Then
         Selection.Range.HighlightColorIndex = wdPink
         k = k + 1
      End If
    Loop
  End With
'  ****************************************
'  und nun Windows-Sonderzeichen mit 2 diakritischen Zeichen (für Word 2003 aufwärts):
  Rng1.Select
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
```

```
  With Selection.Find
    .Text = "?[" & ChrW(768) & "-" & ChrW(879) & "][" & ChrW(768) & "-" & ChrW(879) & "]"
    .Replacement.Text = ""
    .Highlight = False
    .Forward = True
    .MatchWildcards = True  ' ist hier unbedingt nötig!
     Do While .Execute      ' (tue, solange etwas gefunden wird)
       If Selection.InRange(Rng1) = False Then Exit Do
       If Dialogs(wdDialogInsertSymbol).Font <> "(normaler Text)" Then
          Selection.Range.HighlightColorIndex = wdPink
          k = k + 1
       End If
     Loop
  End With
'  ******************
  Fkt_Windows_Sonderschrift = k
End Function
' ---------------------------------------------------------------------------------------------------------------------
```

```
Private Function Fkt_Bidirektionale_Steuerzeichen_markieren(aRange As Range) As String
' H. Chris Gast, 4.bis 8. März 2012, 9. Dezember 2015, 12. + 13. Januar 2017.
' Prozedur zum Violet-Färben aller Bidirektionalen Steuerzeichen.
' In Tabellen werden zusätzlich die Felder rosa schattiert.
' Für Word 2000 bis Word 2010.
Dim Rng0 As Range
Dim Rng1 As Range
Dim IntFundstelle As Integer
Dim strWarnung_Bidi As String
'
aRange.Select
"Selection.HomeKey Unit:=wdStory
aRange.Collapse
'
' Ab hier Schleife:
Zeile1:
  Selection.Find.ClearFormatting
  With Selection.Find
    ' Eines der folgenden Steuerzeichen finden:
    .Text = "[" & ChrW(8206) & ChrW(8207) & ChrW(8234) _
          & ChrW(8235) & ChrW(8236) & ChrW(8237) & ChrW(8238) & "]"
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindStop
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
  End With
  Selection.Find.Execute          ' Replace:=wdReplaceAll
  If Selection.Find.Found Then
      If Selection.Information(wdWithInTable) = False Then
    Selection.SetRange Start:=Selection.Start - 1, End:=Selection.End + 1
    Selection.Range.HighlightColorIndex = wdViolet
    IntFundstelle = IntFundstelle + 1
        Else
          Set Rng0 = Selection.Range
          Set Rng1 = Selection.Range
          Rng0.SetRange Start:=Selection.Start - 1, End:=Selection.End + 1
```

```
            If InStr(Rng0, ChrW(7)) = 0 Then
              Rng0.Select
              Selection.Range.HighlightColorIndex = wdViolet
              IntFundstelle = IntFundstelle + 1
            Else
              Selection.Range.HighlightColorIndex = wdViolet
              Rng1.Cells(1).Select
              Selection.Cells.Shading.BackgroundPatternColor = wdColorRose
              IntFundstelle = IntFundstelle + 1
            End If
          End If
      Selection.Collapse Direction:=wdCollapseEnd
      GoTo Zeile1
    End If
    Selection.Collapse
    '
    If IntFundstelle > 0 Then
      Fkt_Bidirektionale_Steuerzeichen_markieren = vbCrLf & vbCrLf _
      & "ACHTUNG: Die Datei enthält Bidirektionale Steuerzeichen, " _
      & vbCrLf & "welche die Umwandlung in Pdf behindern " & vbCrLf _
      & "(violett markiert, bzw. das Tabellenfeld rosa schattiert)."
    Else
      Fkt_Bidirektionale_Steuerzeichen_markieren = ""
    End If
  End Function
  '-----------------------------------------------------------------------------------------------------------
```

21 Anzeige aller Zeichenobjekte und Rahmen in der Markierung

Ist der Rahmen in der Word-Datei ein "Shape" oder ein "Frame"? Oder ist das eine Grafik oder ein Zeichenobjekt? Dies kann das folgende Makro beantworten.

Wird nichts markiert, wird die Datei nach Rahmen, Grafiken und Zeichenobjekten durchsucht.

```
Sub alle_Shapes_Frames_InlineShapes_in_der_Markierung_zeigen()
' H. Chris Gast, 8. Oktober 2012, Stand 18. November 2012.
' Zählt und zeigt auf Wunsch alle Rahmen und grafischen Gebilde in der Markierung.
' Wenn nichts markiert ist, wird die ganze Dokumentkomponente  des Dokuments durchsucht.
' Wird "Abbrechen" angeklickt, bleibt der Cursor auf dem gerade gezeigten Gebilde.
' Das Makro kann nicht gleichzeitig ein Shape und darin befindliche InlineShapes erkennen.
' *********
Dim i As Integer ' Shapes ("Formen")
Dim j As Integer ' Frames  ("Rahmen")
Dim k As Integer ' InlineShapes (Bilder und Formeln)
Dim Abfrage As Integer
Dim Rng0 As Range: Set Rng0 = Selection.Range
Dim Rng1 As Range
Dim Ganzes_Dokument As Boolean: Ganzes_Dokument = False
'  **********
  If Selection.Type < 2 Then
    '' MsgBox "Es ist nichts ausgewählt!"
    '' Call a04_Alle_Rahmen_usw_in_Hauptkomponente_aufsuchen_mit_Seitenangabe
    '' Exit Sub
    Set Rng1 = Rng0.Duplicate
    Selection.WholeStory
    Set Rng0 = Selection.Range
    Ganzes_Dokument = True
  End If
'  **********
  ' Shapes zählen:
  If Selection.Range.StoryType = 2 Or Selection.Range.StoryType = 3 _
  Or Selection.Range.StoryType = wdTextFrameStory Then
    ' In Fußnoten (2) und Endnoten (3) sind keine Shapes zulässig.
    ' Oder die Markierung befindet sich innerhalb eines Textfeldes.
    i = 0
  Else
    On Error Resume Next
    '  Wenn das Dokument keinerlei Shapes enthält, gibt es ohne Fehlerbehandlung in Word 2000
    '  bei der folgenden Zeile einen Absturz.
    i = Selection.Range.ShapeRange.Count
    If Err.Number > 0 Then
      i = 0
      ' MsgBox Err.Number
      On Error GoTo 0
    End If
  End If
  ' *******
  ' Frames (Positionsrahmen) zählen:
  j = Selection.Range.Frames.Count
  ' *******
  ' InlineShapes zählen:
  k = Selection.Range.InlineShapes.Count
'  *********
  ' Anzeige der Zählungen in einer Messagebox:
  If i = 0 And j = 0 And k = 0 Then
    If Ganzes_Dokument = True Then
      MsgBox "Es gibt in der Dokumentkomponente keine Shapes, Frames oder InlineShapes."
      Rng1.Select
```

```
    Else
      MsgBox "Es gibt in der Markierung keine Shapes, Frames oder InlineShapes."
    End If
    Exit Sub
  End If
  Abfrage = MsgBox("Es gibt in der Markierung insgesamt " _
      & i & " Shapes, " & j & " Frames und " & k & " InlineShapes." & vbCrLf _
      & vbCrLf & "Wollen sie diese einzeln sehen?", vbYesNoCancel + vbDefaultButton2)
  ' Bei "Ja" kommt "6" heraus, bei "Nein" kommt "7" heraus, bei Abbrechen eine "2".
  If Abfrage <> 6 Then
    Exit Sub
  End If
'  ************************
' Zeigt Shapes (Zeichnungs-Elemente oder Textfelder).
  If i > 0 Then
    For i = 1 To Selection.Range.ShapeRange.Count
     Selection.Range.ShapeRange(i).Select
     ActiveWindow.ScrollIntoView Selection.Range  ' Sonst sieht man nichts!
     If Selection.ShapeRange(1).Type = msoTextBox Then  ' 17
       "MsgBox "Hier ist ein Textfeld (Textbox)."
       Abfrage = MsgBox("Hier ist ein Textfeld (Textbox).", vbOKCancel)
       If Abfrage = 2 Then Exit Sub
     ElseIf Selection.ShapeRange(1).Type = msoPicture Then  ' 13
       "MsgBox "Hier ist ein frei positioniertes Bild (Picture)."
       Abfrage = MsgBox("Hier ist ein frei positioniertes Bild (Picture).", vbOKCancel)
       If Abfrage = 2 Then Exit Sub
     ElseIf Selection.ShapeRange(1).Type = msoLinkedPicture Then   ' 11
       "MsgBox "Hier ist ein frei positioniertes verknüpftes Bild (Linked Picture)."
       Abfrage = MsgBox("Hier ist ein frei positioniertes verknüpftes Bild (Linked Picture)., _
                 vbOKCancel)
       If Abfrage = 2 Then Exit Sub
     ElseIf Selection.ShapeRange(1).Type = msoAutoShape Then   ' 1
       "MsgBox "Hier ist ein ""AutoShape""."
       Abfrage = MsgBox("Hier ist ein ""AutoShape""., vbOKCancel)
       If Abfrage = 2 Then Exit Sub
     Else
       "MsgBox "Hier ist eine AutoForm, Zeichnungsobjekt oder sonstige Grafik (Shape)."
       Abfrage = MsgBox("Hier ist eine AutoForm, Zeichnungsobjekt oder sonstige Grafik " _
                 & (Shape)., vbOKCancel)
       If Abfrage = 2 Then Exit Sub
     End If
     Rng0.Select
    Next i
  End If
'  ***************
' zeigt Frames (Positionsrahmen):
 ''  MsgBox "Es gibt " & ActiveDocument.Frames.Count & " Frames bzw. Positionsrahmen."
  For j = 1 To Selection.Range.Frames.Count
    Selection.Range.Frames(j).Select
    ActiveWindow.ScrollIntoView Selection.Range  ' Sonst sieht man nichts!
    "MsgBox "Hier ist ein Positionsrahmen (Frame)."
    Abfrage = MsgBox("Hier ist ein Positionsrahmen (Frame).", vbOKCancel)
    If Abfrage = 2 Then Exit Sub
    Rng0.Select
  Next j
  ' ***********
   ' zeigt InlineShapes wie Bilder und Formeleditor-Objekte:
   For k = 1 To Selection.Range.InlineShapes.Count
    Selection.Range.InlineShapes(k).Select
    ActiveWindow.ScrollIntoView Selection.Range  ' Sonst sieht man nichts!
    If Selection.Range.InlineShapes(1).Type < 3 Then
       '' MsgBox "Hier ist eine Gleichung eingebettet (InlineShape)."
```

```
        Abfrage = MsgBox("Hier ist eine Gleichung eingebettet (InlineShape).", vbOKCancel)
        If Abfrage = 2 Then Exit Sub
      Else
        '  MsgBox "Hier ist ein Bild in der Zeile (InlineShape)."
        Abfrage = MsgBox("Hier ist ein Bild in der Zeile (InlineShape).", vbOKCancel)
        If Abfrage = 2 Then Exit Sub
      End If
      Rng0.Select
    Next k
      ' ************
  ' Markierung zurückstellen:
  If Ganzes_Dokument = True Then Rng1.Select
End Sub
```

Nicht gesucht und angezeigt werden Tabellen, auch wenn Rahmen sich als Tabelle entpuppen können. Hier könnte ich das Makro vielleicht noch erweitern.

Wenn nichts markiert ist, wird nur die Dokumentkomponente durchsucht, in der der Cursor steht.

22 Makro für Text-Nachbearbeitung nach OCR oder nach Konvertierung in Word

22.1 Text-Nachbearbeitung

Es wird im folgenden Makro nur die markierte Textstelle nachbearbeitet.

Texte, die durch Texterkennung entstanden sind, oder Texte, die aus Outlook stammen oder aus einer Pdf-Datei, sollten nachbearbeitet werden. So findet man statt Blocksatz in jeder Zeile Absatzmarken oder Zeilenumbrüche, und Striche können in Wirklichkeit entweder Silbentrennung, Aufzählungszeichen oder auch normale Bindestriche sein. Dies wird im folgenden Makro behoben.

Hyperlinks und E-Mails werden vom Makro deaktiviert, aber wieder unterstrichen. E-Mails werden dabei grün doppelt unterstrichen. Mit dem Makro in Abschnitt 5.2 kann man aber hinterher alle Links ins Internet aktivieren.

Dies ist zurzeit eines meiner am häufigsten benutzten Makros, deshalb lasse ich den Makro-Namen mit "a1" beginnen.

```
' Makromodul OCR_Nachbearbeitung

Option Explicit

Sub a1_OCR_Nachbearbeitung_in_Markierung()
' Hanna-Chris Gast,  Stand 18. Mai 2018.
'
' Dies Makro beseitigt in (und nur in) der Markierung Absatzmarken und Zeilenwechsel
' weitgehend aus gescanntem Text, wenn es keine echten Absätze sind.
'
' Dies Makro unterscheidet  anhand der Rechtschreibprüfung echte Silbentrennung
' von Festen Bindestrichen.
'
' Ergänzt Leerzeichen hinter Satzzeichen, wenn keine Zahlen folgen,
' und beseitigt falsche Leerzeichen vor Satzzeichen.
' (in französischen Texten gehört vor Satzzeichen allerdings ein Festes Leerzeichen,
' hier nicht berücksichtigt).
'
' Markiert Gänsefüßchen mit Leerzeichen davor und dahinter.
'
' Erkennt teilweise auch Aufzählungen.
'
' Hyperlinks werden deaktiviert, aber wieder unterstrichen.
' E-Mails werden grün doppelt unterstrichen und deren Verknüpfung deaktiviert, damit nicht
' versehentlich der Browser gestartet wird beim Anklicken.
'
' Ersatz von Zeilenumbruch durch Absatzmarken im Format "Standard". Jedoch nicht bei
' anderen Formatvorlagen, das zerstört sonst Anhangstitel.
'
' Version mit unsichtbarer Zwischendatei, in der alle Befehle ablaufen.
' Word 2000 bis Word 2010.
'
' Abfragen bei Makrotext, damit diese nicht versehentlich bearbeitet werden (5. April 2016).
'
' Offen:
```

```
' Die Rechtschreibprüfung für den gesamten Text ist in einem separaten Makro,
' ebenso die Rekonstruktion von Überschriftsebenen (auf Anfrage lieferbar).
' Das Erkennen von E-Mail-Adressen hat leider noch einige Macken.
' ********************************
Dim Rng1 As Range
Dim Rng2 As Range
Dim rngStory As Range  ' Dokumentkomponente
Dim Z2 As Long  ' Ende des Rng1
Dim strTestwort As String
Dim IntAbfrage
Dim Dok1 As Word.Document
Dim Dok2 As Word.Document
Dim B_Tabelle As Boolean: B_Tabelle = False
'  ********************************
'  Prüfen, ob etwas markiert ist:
  If Selection.Type < 2 Then
    MsgBox "Es ist nichts markiert!"
    Exit Sub
  End If
'  ********************************
'  Problem beim Wiedereinfügen, wenn mehrere Zellen einer Tabelle ersetzt werden,
'  aber nicht die ganze Tabelle:
  ' If Selection.Information(wdWithInTable) = True Then
  If Selection.Tables.Count > 0 Then
    If InStr(Selection, ChrW(7)) = 0 Then
      ' Nur eine Zelle ist ok:
      ' Markierung als Range speichern und kopieren:
      Set Rng1 = Selection.Range
      Rng1.ListFormat.ConvertNumbersToText
      Selection.Copy
    ElseIf Selection.Information(wdWithInTable) = True Then
      ' nur Zellen in einer Tabelle:
      If Selection.Cells.Count < Selection.Range.Tables(1).Range.Cells.Count Then
        MsgBox "Entweder nur eine Zelle oder die ganze Tabelle markieren!" & vbCrLf _
            & " Das Makro wird abgebrochen.", vbExclamation
        Exit Sub
      Else
        ' Ganze Tabelle Markieren
        Selection.Tables(1).Select
        B_Tabelle = True
      End If
    Else
      'Mehrere Tabellen oder eine ganze Tabelle, Tabellen vollständig markieren:
      If Selection.Range.Start > Selection.Range.Tables(1).Range.Start Then
        Selection.SetRange Start:=Selection.Range.Tables(1).Range.Start, _
        End:=Selection.Range.End
      End If
      If Selection.Range.End < Selection.Range.Tables(Selection.Tables.Count).Range.End Then
        Selection.SetRange Start:=Selection.Range.Start, _
        End:=Selection.Range.Tables(Selection.Tables.Count).Range.End
      End If
    End If
  End If
'  ********************************
'  Damit Word beim Wieder-Einfügen nicht ein Leerzeichen ergänzt,
'  darf die Markierung nicht mitten in einem Wort beginnen oder enden.
  ' Eine am Anfang alleinstehende Absatzmarke soll nicht erfasst werden:
  If Asc(Left(Selection, 1)) = 13 Then
    Selection.MoveStart Unit:=wdCharacter, Count:=1
  Else
    Selection.MoveStartWhile _
     CSet:="ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÜ" _
```

```
      & "abcdefghijklmnopqrstuvwxyzöäüß-1234567890.", _
      Count:=wdBackward
  End If
  ' Es darf auch nicht die Markierung bei einer Absatzmarke erweitert werden.
  If Right(Selection, 1) Like "[A-ZÄÖÜa-zöäüß1-90.]" = True Then
    Selection.MoveEndWhile _
      CSet:="ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÜ" _
      & "abcdefghijklmnopqrstuvwxyzöäüß-1234567890.", _
      Count:=wdForward
  End If
  Set Rng1 = Selection.Range
' ********************************
' Nummerierungen und Aufzählungen fixieren:
  Rng1.ListFormat.ConvertNumbersToText
' ********************************
' Das Makro nicht für Makro-Texte verwenden:
  If InStr(Rng1.Text, "End Sub") > 0 _
  Or InStr(Rng1.Text, "End Function") > 0 Then
    IntAbfrage = _
    MsgBox("Wollen Sie trotz Makrotext Leerzeichen bei Satzzeichen usw. einfügen?", _
    vbYesNoCancel + vbDefaultButton2)
    If IntAbfrage <> 6 Then Exit Sub   ' ja = 6
  End If
' ********************************
' Lästiges "Flackern" des Bildschirms verhindern:
  StatusBar = "Bitte warten - das Makro läuft."
  Application.ScreenUpdating = False
  Options.Pagination = False
' ********************************************************************
' ********************************************************************
' Markierten Text in einem Zwischen-Dokument bearbeiten,
' damit nicht bei manchen Befehlen der Rest des Dokuments verändert wird
' (Ersetzungen erfolgten leider manchmal auch außerhalb der Markierung):
  Selection.Copy
' ****************
' Zwischendokument erstellen:
  Set Dok1 = ActiveDocument  ' (für die Rückkehr zum richtigen Dokument).
  WordBasic.DisableAutoMacros 1   ' Disables auto macros
  Set Dok2 = Documents.Add(DocumentType:=wdNewBlankDocument, Visible:=False)
  Dok2.Activate
  ' Falls in der Normal-dot ein Briefkopf vorhanden ist, diesen löschen:
  Selection.WholeStory
'  Selection.Delete
' ****************
' Im Zwischendokument beim Einfügen per Errorfunktion prüfen,
' ob die Zwischenablage nicht leer ist und auch Text enthält:
On Error GoTo Zeile_Zwischenablage_leer
  Selection.Paste
On Error GoTo Zeile_Fehler
Selection.WholeStory
' ********************************
' Alle Farbmarkierungen entfernen:
  ' Farbmarkierung beseitigen, die eine "Hervorhebung" ist:
  Selection.Range.HighlightColorIndex = wdNoHighlight
  ' Farbmarkierung beseitigen, die eine "Schattierung" bzw. "Hintergrundsfarbe" ist:
  With Selection.Font.Shading
    .Texture = wdTextureNone
    .ForegroundPatternColor = wdColorAutomatic
    .BackgroundPatternColor = wdColorAutomatic
  End With
  ' Farbmarkierung beseitigen, die auf Absatzeigenschaften beruht:
  With Selection.ParagraphFormat.Shading
```

```
    .Texture = wdTextureNone
    .ForegroundPatternColor = wdColorAutomatic
    .BackgroundPatternColor = wdColorAutomatic
  End With
' ********************************
' Markierung als Range speichern:
  Set Rng2 = Selection.Range
' ********************************
' Zeilenwechsel in Absatzmarken umwandeln:
  Rng2.Find.ClearFormatting
  Rng2.Find.Replacement.ClearFormatting
  "rng2.Find.Style = ActiveDocument.Styles(wdStyleNormal)
  With Rng2.Find
    .Text = "^l"
    .Replacement.Text = "^p"
    .Forward = True
    '.Wrap = wdFindAsk
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Rng2.Find.Execute Replace:=wdReplaceAll
' ********************************
' Schleife bilden, in der alle Absatzmarken, die ziemlich weit rechts liegen,
' durch Leerzeichen ersetzt werden, wenn die Zeile nicht mit einem Satzende endet.
' Die meisten (aber nicht alle) echten Absatzmarken sind weiter links:
' Von hinten nach vorne, bzw. von unten nach oben!  5. April 2016.
  Selection.EndKey Unit:=wdStory
  Selection.Collapse Direction:=wdCollapseEnd
  Selection.Find.ClearFormatting
  With Selection.Find
      .Text = "[A-zäöüß,]^13"
      .Font.Bold = False
      .Replacement.Text = ""
      .Forward = False
      .Wrap = wdFindStop
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchAllWordForms = False
      .MatchSoundsLike = False
      .MatchWildcards = True
      Do While .Execute
        ' Nur ersetzen, wenn sehr weit rechts (außer in Tabellen):
        "If Selection.Information(wdHorizontalPositionRelativeToTextBoundary) > 250 Then
        If Abs(Int(Selection.Range.PageSetup.TextColumns(1).Width _
        - Selection.Information(wdHorizontalPositionRelativeToTextBoundary))) < 50 Then
            Selection = Replace(Selection, ChrW(13), " ")
            Selection.Range.HighlightColorIndex = wdYellow
        End If
        Selection.MoveUp Unit:=wdParagraph
      Loop
  End With
  "Selection.Find.Execute
' ********************************
' Mehrfache Leerzeichen beseitigen:
  Rng2.Find.ClearFormatting
  Rng2.Find.Replacement.ClearFormatting
  With Rng2.Find
```

```
    .Text = " {2;}"
    .Replacement.Text = " "
    .Forward = True
    ' .Wrap = wdFindContinue
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
  End With
  Rng2.Find.Execute Replace:=wdReplaceAll
  '  Markierung wiederherstellen:
  Rng2.Select
' ********************************
' Leerzeichen vor Absatzmarke entfernen,
  ' aber nicht am Ende des Textes:
  Selection.MoveEnd Unit:=wdCharacter, Count:=-1
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
      .Text = " ^p"
      .Replacement.Text = "^p"
      '.Forward = True
      ' .Wrap = wdFindContinue
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
' ********************************
' Markierung wiederherstellen:
  Rng2.Select
' ***********************************************************************
' In verkorksten Manuskripten fast alle waagerechten Striche vereinheitlichen:
' (ausgenommen das nicht-funktionierende Zeichen "dez. 173"):
  Call Striche_vereinheitlichen
  Call Striche_am_Zeilenanfang_unterscheiden
' ***********************************************************************
' Hyperlinks farbig markieren, damit diese im Folgenden ausgenommen bleiben:
  ' Auch in Fußnoten suchen (15.4.2016):
  ' Hauptkomponente, Fußnoten, Endnoten nach Hyperlinks durchsuchen:
  For Each rngStory In ActiveDocument.StoryRanges
    ' Nur Hauptkomponenten, Fußnoten und Endnoten prüfen:
    If rngStory.StoryType > 3 Then Exit For
        rngStory.Select
        ' Die eigentliche Prozedur:
        Call Hyperlinks_in_Markierung_aktivieren_und_faerben
        Call Hyperlinks_entlinken_und_faerben
   Next rngStory
  ' Haupt-Dokumentkomponente wieder markieren:
  ActiveDocument.StoryRanges(1).Select
' ***********************************************************************
' Bereits vorhandene SoftHyphen ChrW(173) aus OCR durch Bedingten Strich ersetzen,
' weil nur das Steuerzeichen ChrW(31) als Bedingter Strich in Word funktioniert:
  Options.DefaultHighlightColorIndex = wdYellow
  Rng2.Find.ClearFormatting
  Rng2.Find.Replacement.ClearFormatting
  With Rng2.Find
```

```
        .Text = ChrW(173)
        .Replacement.Text = ChrW(31)
        .Forward = True
        '' .Wrap = wdFindContinue
        .Format = True
        .Replacement.Highlight = True
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = False
        .MatchSoundsLike = False
        .MatchAllWordForms = False
    End With
    Rng2.Find.Execute Replace:=wdReplaceAll
    Rng2.Select
'  ******
'  Falsches SoftHyphen-Zeichen ChrW(172) (Nichtzeichen) durch bedingten Strich ersetzen,
'  wenn es hinter einem einfachen Buchstaben steht
'  (dieser Fehler entsteht fälschlich durch "Einfügen-nur Text"):
    Options.DefaultHighlightColorIndex = wdYellow
    Selection.Find.ClearFormatting
   ' Rng2.Find.Replacement.ClearFormatting
    With Selection.Find
      .Text = "[a-zäöüß]" & ChrW(172)
      .Replacement.Text = ChrW(31)
      .Forward = True
      '' .Wrap = wdFindContinue
      .Format = True
      .Replacement.Highlight = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = True
      .MatchSoundsLike = False
      .MatchAllWordForms = False
      Do While .Execute
        Selection.Text = Replace(Selection, ChrW(172), ChrW(31))
        Selection.Range.HighlightColorIndex = wdYellow
        Selection.Collapse Direction:=wdCollapseEnd
      Loop
    End With
    ''Rng2.Find.Execute Replace:=wdReplaceAll
    Rng2.Select
'  ************************************
'  Leerzeichen hinter SoftHyphen löschen:
    Options.DefaultHighlightColorIndex = wdYellow
    Rng2.Find.ClearFormatting
    Rng2.Find.Replacement.ClearFormatting
    With Rng2.Find
        .Text = ChrW(31) & " "
        .Replacement.Text = ChrW(31)
        .Forward = True
        '' .Wrap = wdFindContinue
        .Format = True
        .Replacement.Highlight = True
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = False
        .MatchSoundsLike = False
        .MatchAllWordForms = False
    End With
    Rng2.Find.Execute Replace:=wdReplaceAll
    Rng2.Select
'  **************
```

```
' Absatzmarke hinter SoftHyphen löschen:
  Options.DefaultHighlightColorIndex = wdYellow
  Rng2.Find.ClearFormatting
  Rng2.Find.Replacement.ClearFormatting
  With Rng2.Find
      .Text = ChrW(31) & ChrW(13)
      .Replacement.Text = ChrW(31)
      .Forward = True
      '' .Wrap = wdFindContinue
      .Format = True
      .Replacement.Highlight = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
  Rng2.Find.Execute Replace:=wdReplaceAll
  Rng2.Select
' ******************************************************************
' Striche vor Zeilenumbruch und folgendem Kleinbuchstaben unterscheiden:
' a) Silbentrennung mit Bedingtem Trennstrich (Soft Hyphen)
' b) Fester Bindestrich
' c) einfacher Strich
' ******************************
' Fall Strich, Zeilenwechsel, Kleinbuchstabe:
  Rng2.Select
  Selection.Find.ClearFormatting
  With Selection.Find
      .Text = "-^13[a-zäöüß]"
      .Replacement.Text = ""
      .Forward = True
      ' .Wrap = wdFindContinue
      .Font.Underline = wdUnderlineNone
      .Format = True
      .MatchCase = True
      .MatchWholeWord = False
      .MatchWildcards = True
      .MatchSoundsLike = False
      .MatchAllWordForms = False
      Do While .Execute
        ' Die Ersetzung darf nur innerhalb der Markierung erfolgen:
        ''If Selection.End > rng2.End Then Exit Do
        If Selection.InRange(Rng2) = False Then Exit Do
        ' Ganzes Wort markieren zwecks Rechtschreibprüfung:
        Selection.MoveStartUntil CSet:=" ,.-" & ChrW(13) & ChrW(11), Count:=wdBackward
        Selection.MoveEndUntil CSet:=" ,.-" & ChrW(13) & ChrW(11), Count:=wdForward
        strTestwort = Replace(Selection.Text, "-" & ChrW(13), "")
        If InStr(Selection.Text, ChrW(13) & "und") > 0 _
        Or InStr(Selection.Text, ChrW(13) & "oder") > 0 Then
          ' Einfacher Strich + Leerzeichen:
          Selection.Text = Replace(Selection.Text, "-" & ChrW(13), "- ")
        ElseIf Application.CheckSpelling(Word:=strTestwort) = True Then
          ' Wenn Rechtschreibung bei Zusammenschreibung ok, dann Silbentrennungsstrich:
          Selection.Text = Replace(Selection.Text, "-" & ChrW(13), ChrW(31))
        Else
          ' anderenfalls normaler Bindestrich(ohne Leerzeichen):
          Selection.Text = Replace(Selection.Text, "-" & ChrW(13), "-")
        End If
        Selection.Range.HighlightColorIndex = wdYellow
        Selection.Collapse Direction:=wdCollapseEnd
      Loop
```

```
  End With
  Rng2.Select
' ********************************
' Fall Buchstabe, Strich, Zeilenwechsel, Großbuchstabe,
' dann Zeilenwechsel löschen,
' Fester Bindestrich nur, wenn vor Bindestrich nur ein Großbuchstabe:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "[A-ZÖÄÜa-zäöüß]-^13[A-ZÄÖÜ]"
    .Replacement.Text = ""
    .Forward = True
    '.Wrap = wdFindContinue
    .Font.Underline = wdUnderlineNone
    .Format = True
    .MatchCase = True
    .MatchWholeWord = False
    .MatchWildcards = True
    .MatchSoundsLike = False
    .MatchAllWordForms = False
    Do While .Execute
      If Selection.InRange(Rng2) = False Then Exit Do
      ' Ganzes Wort markieren (5.3.2015):
      Selection.MoveStartUntil CSet:=" ,.-" & ChrW(13) & ChrW(11), Count:=wdBackward
      If InStr(Selection, "-") = 2 Then
        ' Fester Bindestrich (bei Wörtern wie "E-Modul" und S-Bahn"):
        Selection.Text = Replace(Selection.Text, "-" & ChrW(13), ChrW(30))
        Selection.Range.HighlightColorIndex = wdYellow
        Selection.Collapse Direction:=wdCollapseEnd
      Else
        ' Normaler Bindestrich:
        Selection.Text = Replace(Selection.Text, "-" & ChrW(13), "-")
        Selection.Range.HighlightColorIndex = wdYellow
        Selection.Collapse Direction:=wdCollapseEnd
      End If
    Loop
  End With
' ********************************
' Fester Bindestrich bei Wörtern wie "E-Modul" usw. (in Fließtext nötig):
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "[ ""][A-ZÖÄÜa-zäöüß]-[A-ZÄÖÜ]"
    .Replacement.Text = ""
    .Forward = True
    '.Wrap = wdFindContinue
    .Font.Underline = wdUnderlineNone
    .Format = True
    .MatchCase = True
    .MatchWholeWord = False
    .MatchWildcards = True
    .MatchSoundsLike = False
    .MatchAllWordForms = False
    Do While .Execute
      If Selection.InRange(Rng2) = False Then Exit Do
      ' Fester Bindestrich (bei Wörtern wie "E-Modul" und S-Bahn"):
      Selection.Text = Replace(Selection.Text, "-", ChrW(30))
      Selection.Range.HighlightColorIndex = wdYellow
      Selection.Collapse Direction:=wdCollapseEnd
    Loop
  End With
' ******************************************************************
```

```
'  Fall: Zeilenwechsel + folgender Kleinbuchstabe
'  (= Vermutlich Fließtext, könnte aber auch Aufzählung mit Aufzählungsbuchstabe sein.)
  Rng2.Select
  Selection.Find.ClearFormatting
  ''Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "[!:." & ChrW(13) & "]^13[a-zäöüß]"
    .Replacement.Text = ""
    .Forward = True
    '.Wrap = wdFindContinue
    .Font.Underline = wdUnderlineNone
    .Format = True
    .MatchCase = True
    .MatchWholeWord = False
    .MatchWildcards = True
    .MatchSoundsLike = False
    .MatchAllWordForms = False
    Do While .Execute
      If Selection.InRange(Rng2) = False Then Exit Do
      ' Zeilenumbruch lassen, wenn Schriftgrößen unterschiedlich!
      If Selection.Range.Font.Size > 99 Then Exit Do
      Selection.MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdExtend
      If Selection.Text Like "*)" = False Then
        ' keine Aufzählung
        ' Absatzmarke durch Leerzeichen ersetzen:
        Selection.Text = Replace(Selection.Text, ChrW(13), ChrW(32))
        Selection.Range.HighlightColorIndex = wdYellow
      End If
      Selection.Collapse Direction:=wdCollapseEnd
    Loop
  End With
  Rng2.Select
'  ********************************
'''  Fall: Zeilenwechsel (außer bei Satzende) + folgender Großbuchstabe
''   ' eventuell Fließtext?, Oder Ende einer Überschrift? 18. April 2016.
''   Rng2.Select
''   Selection.Find.ClearFormatting
''   ''Selection.Find.Replacement.ClearFormatting
''   With Selection.Find
''     .Text = "[!:." & ChrW(13) & "]^13[A-ZÄÖÜ]"
''     .Replacement.Text = ""
''     .Forward = True
''     '.Wrap = wdFindContinue
''     .Font.Underline = wdUnderlineNone
''     .Format = True
''     .MatchCase = True
''     .MatchWholeWord = False
''     .MatchWildcards = True
''     .MatchSoundsLike = False
''     .MatchAllWordForms = False
''     Do While .Execute
''       ' Prüfen, ob Schrift in beiden Zeilen gleichgroß (keine Überschrift), und
''       ' prüfen, ob keine Aufzählung mit Buchstaben (nach Klammer-zu suchen):
''       ''''''''''''''''''''''
''       ' 13. Mai 2016: E-Mail Header und -Signaturen ausnehmen:
''       Selection.MoveRight Unit:=wdWord, Count:=1, Extend:=wdExtend
''       If InStr(Selection.Text, "Von") > 0 _
''       Or InStr(Selection.Text, "Gesendet") > 0 _
''       Or InStr(Selection.Text, "Datum") > 0 _
''       Or InStr(Selection.Text, "Betreff") > 0 _
''       Or InStr(Selection.Text, "An") > 0 _
''       Or InStr(Selection.Text, "Tel") > 0 _
```

```
"        Or Mid(Selection, 2, 1) Like "[0-9]" = True > 0 _
"        Or Right(Selection, 3) = "str" Then
"          ' nichts tun!
"        ''''''''''''''''''''
"        ElseIf Selection.Font.Size < 99 _
"        And Selection.Font.Bold < 99 _
"        And Selection.Text Like "*)" = False Then
"          ' einheitliche Schriftgröße, einheitlich fett oder nicht-fett
"          ' und keine Aufzählung; daraus folgt:
"          ' Absatzmarke durch Leerzeichen ersetzen:
"          Selection.Text = Replace(Selection.Text, ChrW(13), ChrW(32))
"          Selection.Range.HighlightColorIndex = wdYellow
"        End If
"        Selection.Collapse Direction:=wdCollapseEnd
"      Loop
"    End With
"    Rng2.Select
'  ************************************************
'  Silbentrennung wiederherstellen in Fließtext, sofern zutreffend.
'  Ersetze Kleinbuchstabe-Bindestrich-Kleinbuchstabe, wenn
'  die Rechtschreibprüfung das Zusammengeschriebene Wort für ok hält:
   Selection.Find.ClearFormatting
   With Selection.Find
       .Text = "[a-zäöü]-[a-zäöü]"
       .Replacement.Text = ""
       .Forward = True
       '.Wrap = wdFindContinue
       .Font.Underline = wdUnderlineNone
       .Format = True
       .MatchCase = True
       .MatchWholeWord = False
       .MatchWildcards = True
       .MatchSoundsLike = False
       .MatchAllWordForms = False
       Do While .Execute
         "If Selection.End > rng2.End Then Exit Do
         If Selection.InRange(Rng2) = False Then Exit Do
         ' Ganzes Wort markieren zwecks Rechtschreibprüfung:
         Selection.MoveStartUntil CSet:=" ,.-" & ChrW(13) & ChrW(11), Count:=wdBackward
         Selection.MoveEndUntil CSet:=" ,.-" & ChrW(13) & ChrW(11), Count:=wdForward
         strTestwort = Replace(Selection.Text, "-", "")
         "MsgBox strTestwort
         If InStr(Selection, "t-online") > 0 Or InStr(Selection, "hanna-chris") > 0 Then
           ' (Bei E-Mails-Bestandteilen nichts tun)
         ElseIf Selection.Range.Underline <> 0 Then
             ' (bei Hyperlinks nicht ändern!)
         ElseIf Application.CheckSpelling(Word:=strTestwort) Then
           ' Wenn Rechtschreibung bei Zusammenschreibung ok, dann Bedingten Trennstrich:
           Selection.Text = Replace(Selection.Text, "-", ChrW(31))
           Selection.Range.HighlightColorIndex = wdYellow
         Else
           ' anderenfalls normaler Bindestrich:
           'Selection.Text = Replace(Selection.Text, "-", ChrW(30))
           'Selection.Range.HighlightColorIndex = wdYellow
         End If
         Selection.Collapse Direction:=wdCollapseEnd
       Loop
   End With
' ***************
' Markierung wiederherstellen
  Rng2.Select
'  ********************************************
```

```
' 11. + 12. August 2015:
' Für englische Texte Änderung bei "E-Mail" rückgängig machen:
  Rng2.Find.ClearFormatting
  Rng2.Find.Replacement.ClearFormatting
  With Rng2.Find
    .Text = "e^-mail"
    .Replacement.Text = "e-mail"
    .Forward = True
    .Wrap = wdFindContinue
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Rng2.Find.Execute Replace:=wdReplaceAll
' ********************************
' häufigen Schreibfehler beseitigen:
  Rng2.Select
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  Selection.Find.Replacement.Highlight = True
  With Selection.Find
    .Text = "E-mail"
    .Replacement.Text = "E-Mail"
    .Forward = True
    '.Wrap = wdFindAsk
    .Format = True
    .MatchCase = True
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
  Rng2.Select
' ********************************
' Gänsefüßchen mit Leerzeichen davor und dahinter dunkelrosa markieren:
  Options.DefaultHighlightColorIndex = wdPink
  Rng2.Find.ClearFormatting
  Rng2.Find.Replacement.ClearFormatting
  Rng2.Find.Replacement.Highlight = True
  With Rng2.Find
    .Text = " "" "
    .Replacement.Text = " "" "
    .Forward = True
    '.Wrap = wdFindContinue
    '' .Font.Underline = wdUnderlineNone ' (gilt auch für Hyperlinks)
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Rng2.Find.Execute Replace:=wdReplaceAll
  Options.DefaultHighlightColorIndex = wdYellow
' Markierung wiederherstellen:
  Rng2.Select
' ************************************
' Leerzeichen bei Satzzeichen prüfen:
```

```
  Call Satzzeichen_Leerzeichen
' ******************************
' Tabellen beseitigen: 22. April 2015:
  Dim atable As Table
  If Selection.Tables.Count > 0 Then
    IntAbfrage = MsgBox("Wollen Sie die Tabellen in der Markierung beseitigen?", vbYesNoCancel)
    If IntAbfrage = 6 Then   ' ja = 6
      For Each atable In Selection.Tables
        atable.Rows.ConvertToText Separator:=wdSeparateByTabs, _
        NestedTables:=True
      Next atable
    End If
  End If
  Rng2.Select
' ******************************
' Shapes beseitigen:
  Call Shapes_alle_umwandeln
' Spaltensatz und Abschnittswechsel (Sections) löschen:
  Call Abschnittswechsel_loeschen
' ******************************
' Auf Wunsch in der gesamten markierten Stelle die Rechtschreibung prüfen:
  '  IntAbfrage = MsgBox("Wollen Sie die Rechtschreibung prüfen?", vbYesNoCancel)
  '  If IntAbfrage = 6 Then   ' ja = 6
  '     Call Rechtschreibfehler_in_Markierung
  '  End If
' ******************************
' Markierung wiederherstellen:
  Rng2.Select
' ********************************************************************
' Alles in der Zwischendatei markieren und in die Zieldatei kopieren
' (außer Absatzmarke am Schluss):
  Selection.WholeStory
  Selection.MoveEnd Unit:=wdCharacter, Count:=-1
  Selection.Copy
' Zwischendatei schließen:
  Dok2.Close (wdDoNotSaveChanges)
' ****************************
'  0,1 Sekunden warten (ergab in Word 2010 sonst Fehlermeldung):
   Dim tt As Single
   tt = Timer
   Do While Timer < tt + 0.1  'Sekunden
      DoEvents
   Loop
' ****************************
' Im ursprünglichen Ziel-Dokument bearbeiteten Text einfügen:
  On Error Resume Next  ' (Es muss auch bei Fehlern unbedingt das "Aufräumen" folgen)
  Dok1.Activate
  If B_Tabelle = True Then
    Selection.Tables(1).Delete
  Else
    Selection.Delete
  End If
  ' Einfügen:
  Selection.Paste
  Selection.SetRange Start:=Rng1.Start, End:=Selection.End
  ' ********
  If InStr(Selection, "<!ehemaliger Abschnittswechsel!>") > 0 Then
    Call Text_einspaltig
  End If
' ************************************************************
' Leerzeichen im folgenden Fall wieder beseitigen:
  Selection.Find.ClearFormatting
```

```
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "Dipl.-Ing."
    .Replacement.Text = "Dipl.-Ing."
    .Forward = True
    .Wrap = wdFindContinue
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
  '
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "Dipl.-Ing."
    .Replacement.Text = "Dipl.-Ing."
    .Forward = True
    .Wrap = wdFindContinue
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
' ************************************************
' Ggf. Inhaltsverzeichnisse zurückstellen:
Dim aTableCont As TableOfContents
  For Each aTableCont In ActiveDocument.TablesOfContents
    aTableCont.Update
  Next aTableCont
  For Each aTableCont In ActiveDocument.TablesOfFigures
    aTableCont.Update
  Next aTableCont
' ************************************************
' Aufräumen:
  Dok1.Activate
  ActiveWindow.ScrollIntoView Selection.Range
  WordBasic.DisableAutoMacros 0   ' Automakros wieder zulassen.
' ************************************************
' Im Suchen/Ersetzen-Menü alles zurückstellen
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
      .Text = ""
      .Replacement.Text = ""
      .Forward = True
      .Wrap = wdFindContinue
      .Format = False
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
' Bildschirm wieder einschalten:
  Application.Visible = True
```

```
  Options.Pagination = True
  StatusBar = False
  Application.ScreenUpdating = True
  Application.DisplayStatusBar = True
'  ********************************************
Exit Sub
' ********************************************************************
' ********************************************************************
Zeile_Zwischenablage_leer:
  Application.Visible = True
  WordBasic.DisableAutoMacros 0
  Dok2.Close (wdDoNotSaveChanges)
  Dok1.Activate
  MsgBox "Die Zwischenablage (Clipboard) ist leer!", vbExclamation
  ''   Set Dok1 = Nothing
  ''   Set Dok2 = Nothing
Exit Sub
'  ************************************
Zeile_Fehler:
  Application.Visible = True
  WordBasic.DisableAutoMacros 0
  ''   ' Für Testzwecke die Fehlerstelle wieder aufsuchen:
  ''   IntAbfrage = MsgBox("Fehler:" & vbCrLf & "Fehlernummer: " & Err.Number & _
  ''      vbCrLf & "Fehlerbeschreibung: " & Err.Description _
  ''      & vbCrLf & vbCrLf & "Wollen Sie testweise an die Fehlerstelle zurück(= OK)?", _
  ''      vbCritical + vbOKCancel, "Hyperlinks entlinken")
  ''   If IntAbfrage <> 2 Then
  ''     Resume
  ''   End If
  MsgBox "Fehler:" & vbCrLf & "Fehlernummer: " & Err.Number & _
       vbCrLf & "Fehlerbeschreibung: " & Err.Description, vbCritical
  Dok2.Close (wdDoNotSaveChanges)
  Dok1.Activate
End Sub
' ---------------------------------------------------------------------------------------------------------


Private Sub Satzzeichen_Leerzeichen()
'  H. Chris Gast, 9. bis 12. Februar 2015, 1. Februar 2017.
'  Leerzeichen hinter Satzzeichen prüfen und ggf. ergänzen.
'  Nicht bei Hyperlinks und E-Mail-Adressen, auch nicht in Programmier-Zeilen!
'  Ausgenommen ist auch, wenn hinter dem Satzzeichen eine Zahl ist.
'  Achtung bei Tabellenzellen-Zeichen.
'  ************************************
' Markierung als Range speichern:
Dim Rng1 As Range
  Set Rng1 = Selection.Range
'  ************************************
  '  Nicht für Makro-Texte verwenden:
  ''   If InStr(Rng1.Text, "End Sub") > 0 _
  ''   Or InStr(Rng1.Text, "End Function") > 0 Then
  ''      Exit Sub
  ''   End If
  '  (diese Abfrage wird in der Hauptprozedur erledigt)
'  ************************************
'  Hinter Satzzeichen in der Markierung stets Leerzeichen, außer bei Gänsefüßchen.
'  Sicherheitshalber sind auch unterstrichene Textstellen ausgenommen (Hyperlinks).
'  Ausgenommen ist auch, wenn hinter dem Satzzeichen eine Zahl ist (1.Februar 2017):
  Options.DefaultHighlightColorIndex = wdGreen
  Selection.Find.ClearFormatting
  With Selection.Find
    .Text = "[,;.:\!\?][! 0-9" & ChrW(34) & ChrW(13) & "]"
```

```
    .Replacement.Text = ""
    ".Forward = True
    ".Wrap = wdFindAsk
    .Format = True
    .Font.Underline = wdUnderlineNone
    .MatchCase = False
    .MatchWholeWord = False
    .Replacement.Highlight = True
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
    'Selection.Find.Execute
    Do While .Execute
      If Selection.InRange(Rng1) = False Then Exit Do
      ' "..." ausnehmen, aber keine ganze Tabellenzelle markieren:
      "If Selection.Information(wdWithInTable) Then
        " Selection.MoveRight Unit:=wdCell
      Selection.MoveEnd Unit:=wdCharacter, Count:=3
      ' nicht für Tabellen, Hyperlinks und E-Mails:
      If Len(Selection) > 6 _
      Or InStr(Selection.Range.Paragraphs(1), "@") > 0 Then
        Selection.Collapse Direction:=wdCollapseEnd
      Else
        If InStr(Selection, "...") = 0 Then
          ' Leerzeichen hinter den Satzzeichen einfügen:
          Selection.Collapse
          Selection.MoveRight Unit:=wdCharacter, Count:=1
          Selection.TypeText Text:=" "
        End If
      End If
      Selection.MoveRight Unit:=wdWord
    Loop
  End With
' Markierung wiederherstellen:
  Rng1.Select
' ******************************
' Vor Satzzeichen kein Leerzeichen (außer bei Französisch,
' aber die Spracherkennung der Textstelle ist sehr unzuverlässig):
  Selection.Find.ClearFormatting
  With Selection.Find
    .Text = "[ ][,;.:\!\?]"
    .Replacement.Text = ""
    .Forward = True
    ".Wrap = wdFindAsk
    .Font.Underline = wdUnderlineNone
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
    'Selection.Find.Execute
    Do While .Execute
      If Selection.InRange(Rng1) = False Then Exit Do
      Selection.Collapse
      Selection.MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdExtend
      If Selection.LanguageID <> wdFrench Then
        Selection.Delete
        Selection.MoveRight ' 24.1.2017
      Else
       Selection.MoveRight
      End If
```

```
    Loop
  End With
'  Markierung wiederherstellen:
  Rng1.Select
End Sub
' ----------------------------------------------------------------------------------------------------------


Private Sub Striche_vereinheitlichen()
'  H. Chris Gast, 9. Februar 2015.
'  alle waagerechten Striche vereinheitlichen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "[-" & ChrW(30) & ChrW(31) & ChrW(8211) & ChrW(8212) & ChrW(8722) & "]"
    .Replacement.Text = "-"
   ' .Forward = True
   ' .Wrap = wdFindContinue
   ' .Font.Underline = wdUnderlineNone '((auch bei Hyperlinks anwenden))
    .Format = True
    .Replacement.Font.Name = ActiveDocument.Styles(wdStyleNormal).Font.Name
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
' ************************************
'  Font "Symbol" bei Strichen beseitigen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = ChrW(61485)
    .Replacement.Text = "-"
   ' .Forward = True
   ' .Wrap = wdFindContinue
   "  .Font.Underline = wdUnderlineNone
    .Font.Name = "Symbol"
    .Format = True
    .Replacement.Font.Name = ActiveDocument.Styles(wdStyleNormal).Font.Name
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
End Sub
' ----------------------------------------------------------------------------------------------------------


Private Sub Striche_am_Zeilenanfang_unterscheiden()
' H. Chris Gast, 10. + 11. Februar 2015.
' Es wird geprüft, ob ein Strich am Zeilenanfang ein Aufzählungszeichen ist.
Dim Rng1 As Range
'  ********************************
  Set Rng1 = Selection.Range
  Rng1.Find.ClearFormatting
  With Rng1.Find
     .Text = "^13- " & vbTab
     .Replacement.Text = "^13" & ChrW(8212) & vbTab
```

```
        .Forward = True
    '   .Wrap = wdFindContinue
        .Format = True
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = True
        .MatchSoundsLike = False
        .MatchAllWordForms = False
    End With
    Rng1.Find.Execute Replace:=wdReplaceAll
    '
    Rng1.Find.ClearFormatting
    With Rng1.Find
        .Text = "^13- "
        .Replacement.Text = "^13" & ChrW(8212) & vbTab
        .Forward = True
        ' .Wrap = wdFindContinue
        .Format = True
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = True
        .MatchSoundsLike = False
        .MatchAllWordForms = False
    End With
    Rng1.Find.Execute Replace:=wdReplaceAll
    '
    Rng1.Find.ClearFormatting
    With Rng1.Find
        .Text = "^13-" & vbTab
        .Replacement.Text = "^13" & ChrW(8212) & vbTab
        .Forward = True
        ".Wrap = wdFindContinue
        .Format = True
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = True
        .MatchSoundsLike = False
        .MatchAllWordForms = False
    End With
    Rng1.Find.Execute Replace:=wdReplaceAll
'  ********************************
    ' Für Aufzählungen Tab setzen und hängend:
    '   Selection.ParagraphFormat.TabStops.Clear
    '   Selection.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(1), _
    '      Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
    '   Selection.ParagraphFormat.TabHangingIndent 1
    '''''''''''''''
    ' Das folgende geht so nicht:
    '  .Replacement.ParagraphFormat.LeftIndent = CentimetersToPoints(1)
    '  .Replacement.ParagraphFormat.FirstLineIndent = CentimetersToPoints(-1)
'  ********************************
'  Sonderfall: "und/oder" Zeilenwechsel, Strich, Kleinbuchstabe:
'  (Voraussetzung: Aufzählungsstriche sind hier bereits ausgeschlossen)
Rng1.Select
    Selection.Find.ClearFormatting
    With Selection.Find
        .Text = "[dr]^13-[a-zäöü]"
        .Replacement.Text = ""
        .Forward = True
        .Wrap = wdFindContinue
        .Format = True
        .MatchCase = False
```

```
      .MatchWholeWord = False
      .MatchWildcards = True
      .MatchSoundsLike = False
      .MatchAllWordForms = False
      Do While .Execute
        If Selection.InRange(Rng1) = False Then Exit Do
        ' Ganzes Wort markieren zwecks Rechtschreibprüfung:
        Selection.MoveStartUntil CSet:=" ,.-" & ChrW(13) & ChrW(11), Count:=wdBackward
        Selection.MoveEndUntil CSet:=" ,.-" & ChrW(13) & ChrW(11), Count:=wdForward
        If InStr(Selection.Text, "und" & ChrW(13)) > 0 _
        Or InStr(Selection.Text, "oder" & ChrW(13)) > 0 Then
          Selection.Text = Replace(Selection.Text, ChrW(13), " ")
        End If
        Selection.Range.HighlightColorIndex = wdYellow
        Selection.Collapse Direction:=wdCollapseEnd
      Loop
    End With
    Rng1.Select
End Sub
' ---------------------------------------------------------------------------------------------------------------


Private Sub Shapes_alle_umwandeln()
' H. Chris Gast, September 2012, Januar 2013,
' 18. Januar 2015, 11. August 2015, 7. + 14. Oktober 2015, 7. April 2016.
' Makro beseitigt störende Shapes und wandelt sie ggf. um in Frames und InlineShapes um.
' Kann nicht alle Shapes-Typen direkt umwandeln, etwa wenn eine Pdf-Datei mittels Adobe X
' in Word umgewandelt wurde.
' Deshalb bei einigen der Umweg durch Aufruf einer Unterprozedur,
' welche bei hartnäckigen Shapes die Word-Datei vorübergehend in eine html-Datei umwandelt.
'
' Erfasst nur Hauptkomponente, keine Fußnoten.
' *****************
Dim i As Integer
Dim j As Integer: j = 0 ' konvertierte Shapes
Dim k As Integer: k = 0 ' Ersatzlos gelöschte Shapes
Dim IntType As Integer
'  ****************
  For i = ActiveDocument.Shapes.Count To 1 Step -1
    If ActiveDocument.Shapes(i).Type = msoTextBox Then  ' 17
      ' Textfelder umwandlen in Textrahmen (Textbox in Frame)
      IntType = ActiveDocument.Shapes(i).Type
      Debug.Print "Shape-Typ: " & IntType
      ActiveDocument.Shapes(i).ConvertToFrame
      j = j + 1
    ElseIf ActiveDocument.Shapes(i).Type = msoPicture Then  ' 13
      ' frei positioniertes Bild (Picture) in normale Bildposition.
      IntType = ActiveDocument.Shapes(i).Type
      Debug.Print "Shape-Typ: " & IntType
      ActiveDocument.Shapes(i).ConvertToInlineShape
      j = j + 1
    ElseIf ActiveDocument.Shapes(i).Type = msoLinkedPicture Then  ' 11
      '  frei positioniertes verknüpftes Bild (Linked Picture).
      IntType = ActiveDocument.Shapes(i).Type
      Debug.Print "Shape-Typ: " & IntType
      On Error Resume Next
      ActiveDocument.Shapes(i).ConvertToInlineShape
      If Err.Number > 0 Then
        ActiveWindow.ScrollIntoView Selection.Range  ' Sonst sieht man nichts!
        ActiveDocument.Shapes(i).Select
        MsgBox "Abbruch: Diese Grafik lässt sich nicht umwandeln! Bitte Makro prüfen.", vbCritical
        ''Selection.Copy
```

```
            Call Einzelshape_knacken_mit_html
            Err.Clear
            "Exit Sub
          End If
          On Error GoTo 0
          j = j + 1
        Else
          ' AutoForm, Zeichnungsobjekt oder sonstige Grafikobjekte löschen:
          IntType = ActiveDocument.Shapes(i).Type
          Debug.Print "Shape-Typ: " & IntType
          ActiveDocument.Shapes(i).Select
          ' ********
          ' Prozedur aufrufen, welche eine weitere Zwischendatei erzeugt und in html umwandelt:
          Call Einzelshape_knacken_mit_html
          ' ********
          k = k + 1
        End If
      Next i
'  ************************************
'  Im 2. Schritt:
Call alle_Frames_durch_Text_ersetzen
  If k + j > 0 Then
    MsgBox "Es wurden " & j & " Shapes direkt in Bilder oder Text umgewandelt und " _
          & k & " Shapes über eine html-Datei in Grafiken umgewandelt."
  End If
End Sub
' ---------------------------------------------------------------------------------------------------------------


Private Sub alle_Frames_durch_Text_ersetzen()
Dim fF As Frame
Dim i As Integer
' Findet nur Rahmen in der Hauptkomponente!
For Each fF In ActiveDocument.Frames
   i = i + 1
   fF.Select
   Selection.Style = ActiveDocument.Styles("Standard")
Next fF
"MsgBox i & " Positionsrahmen wurden gelöscht"
End Sub
' ---------------------------------------------------------------------------------------------------------------


Private Sub Einzelshape_knacken_mit_html()
' H. Chris Gast, 7. + 20. April 2016.
' Manche Shapes lassen sich nur dadurch umwandeln, dass man die Datei
' (bzw. die Stelle in der Datei) in eine einfache (!) html-Datei umwandelt!
' ***********************
Dim doc2 As Document
Dim doc3 As Document
Dim strPath As String
Dim IntViewType As Integer
  IntViewType = ActiveWindow.View.Type
Dim Rng0 As Range
'  **********************
  Set doc2 = ActiveDocument
''''''''''
Selection.Cut
' ***********************
' Zwischendokument 3:
  Set doc3 = Documents.Add(Template:="Normal", NewTemplate:=False, DocumentType:=0)
  Selection.Paste
```

```
' ***
' Speichern als einfaches (!) html, um Shapes durch Grafiken oder Text zu ersetzen:
  ActiveDocument.SaveAs2 FileName:="zzzTemp.htm", FileFormat:= _
    wdFormatFilteredHTML, LockComments:=False, Password:="", AddToRecentFiles _
    :=True, WritePassword:="", ReadOnlyRecommended:=False, EmbedTrueTypeFonts _
    :=False, SaveNativePictureFormat:=False, SaveFormsData:=False, _
    SaveAsAOCELetter:=False, CompatibilityMode:=0
  "ActiveWindow.View.Type = wdWebView
'  ***
'  Speichern und schließen:
  ActiveDocument.Close SaveChanges:=True
'  *********
'  Wieder öffnen:
  Set doc3 = Documents.Open(FileName:="zzzTemp.htm", ConfirmConversions:=False, _
    ReadOnly:=False, AddToRecentFiles:=False, PasswordDocument:="", _
    PasswordTemplate:="", Revert:=False, WritePasswordDocument:="", _
    WritePasswordTemplate:="", Format:=wdOpenFormatAuto, XMLTransform:="")
'  ********
  '" Bildverlinkungen beseitigen:
  "ActiveDocument.Fields.Unlink
  '" Ergebnis markieren:
  "Selection.WholeStory
  '" Ende der Markierung nach vorne verschieben (letzte Absatzmarke gehört nicht dazu):
  "Selection.MoveEnd Unit:=wdCharacter, Count:=-1
  ' ********
'  Ggf. neu entstandene Tabelle ignorieren
'  und nur die neue Grafik kopieren:
  ActiveDocument.InlineShapes(1).Select
  Selection.Fields.Unlink
'  ********
  Selection.Copy
  strPath = doc3.Path
  doc3.Close SaveChanges:=False
'  **********************
'  Ausgangsdokument:
  doc2.Activate
  "Selection.Cut
  Selection.Paste
  Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=True
  Selection.Fields.Unlink
'  **********************
'  Aufräumen:
  ' Temporäre Datei löschen:
  Kill strPath & "\zzzTemp.htm"
  On Error Resume Next
  Kill strPath & "\zzzTemp-Dateien\*.*"
  RmDir strPath & "\zzzTemp-Dateien"
  On Error GoTo 0
  ActiveWindow.View.Type = IntViewType
  "WordBasic.DisableAutoMacros 0   ' Enable auto macros
End Sub
' ----------------------------------------------------------------------------------------------------


Private Sub Abschnittswechsel_loeschen()
' H. Chris Gast, 2012, 2013, Januar 2015, 2.+ 7. + 8. + 20. Juli  2015, 7. Oktober 2015, 20.4.2016.
' Beseitigt im ganzen Dokument Abschnittswechsel ("Sections"), Spaltenwechsel und Seitenwechsel.
Dim k As Integer
Dim i As Integer
'  *********************
  Options.DefaultHighlightColorIndex = wdTurquoise
'  *********************
```

```
  ''  ' Alle Seiten auf Hochkant setzen:
  ''    With ActiveDocument.Range.PageSetup
  ''      .Orientation = wdOrientPortrait
  ''    End With
' ****************************************
' Zeilenabstand auf 0 stellen und Einrückungen beseitigen:
  With ActiveDocument.Range.ParagraphFormat
    .LeftIndent = CentimetersToPoints(0)
    .RightIndent = CentimetersToPoints(0)
    .SpaceBefore = 0
    .SpaceBeforeAuto = False
    .SpaceAfter = 0
    .SpaceAfterAuto = False
    .LineSpacingRule = wdLineSpaceSingle
    ''.Alignment = wdAlignParagraphCenter
   '' .WidowControl = False
    .KeepWithNext = False
    .KeepTogether = False
    .PageBreakBefore = False
    .NoLineNumber = False
    .Hyphenation = True
    .FirstLineIndent = CentimetersToPoints(0)
    .OutlineLevel = wdOutlineLevelBodyText
    .CharacterUnitLeftIndent = 0
    .CharacterUnitRightIndent = 0
    .CharacterUnitFirstLineIndent = 0
    .LineUnitBefore = 0
    .LineUnitAfter = 0
   ' .MirrorIndents = False
   ' .TextboxTightWrap = wdTightNone
  End With
' ***********************************************
' Spaltenumbrüche beseitigen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  Selection.Find.Replacement.Highlight = True
  With Selection.Find
    .Text = "^n"
    .Replacement.Font.Name = "Arial"
    .Replacement.Font.Size = 8
    .Replacement.Font.Color = wdColorGreen
    .Replacement.Text = "<!ehemaliger Spaltenumbruch!> ^p"
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
' ******************************************************************
' Abschnittswechsel ("Sections") beseitigen
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  Selection.Find.Replacement.Highlight = True
  With Selection.Find
    .Text = "^b"
    .Replacement.Font.Name = "Arial"
    .Replacement.Font.Size = 8
    .Replacement.Font.Color = wdColorGreen
```

```
    .Replacement.Text = "<!ehemaliger Abschnittswechsel!> ^p"
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
' **********************
' Seitenumbrüche beseitigen (20.April 2016):
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  Selection.Find.Replacement.Highlight = True
  With Selection.Find
    .Text = "^12"
    .Replacement.Font.Name = "Arial"
    .Replacement.Font.Size = 8
    .Replacement.Font.Color = wdColorGreen
    .Replacement.Text = "<!ehemaliger Seitenumbruch oder Abschnittswechsel!> ^p"
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
' **********************
  Options.DefaultHighlightColorIndex = wdYellow
End Sub
' ----------------------------------------------------------------------------------------------------


Private Sub Text_einspaltig()
' Hanna-Chris Gast, 11. Oktober 2015.
On Error Resume Next
  With Selection.PageSetup.TextColumns
    .SetCount NumColumns:=1
    .EvenlySpaced = True
    .LineBetween = False
    '' .Width = CentimetersToPoints(16) '(ergibt Fehler bei einspaltig)
  End With
' ****************
If Err.Number > 0 Then _
  MsgBox "Fehler in Sub Fehlerbehandlung" & vbCrLf & "Fehlernummer: " & Err.Number & _
      vbCrLf & "Fehlerbeschreibung: " & Err.Description, vbCritical
End Sub
' ----------------------------------------------------------------------------------------------------



Private Sub Hyperlinks_in_Markierung_aktivieren_und_faerben()
' Dipl.-Ing. Hanna-Chris Gast, 2011, 2012, 2014, 2015, 18. + 19. Mai 2018, 10. Juni 2018..
' Dieses Makro färbt Hyperlinks und E-Mails blau und unterstreicht sie.
' Besonderheit:
' Es wird auch nach nicht-aktiven Internet- und E-Mail-Adressen gesucht,
```

```
'  und diese werden aktiviert.
'  ******************************
Dim afield As Field
'  Markierung speichern:
Dim Rng1 As Range
  Set Rng1 = Selection.Range
Dim RngI As Range  ' Inhaltsverzeichnis, das nicht durchsucht werden darf.
'  ********************************************************************
'  Inaktive Hyperlinks und inaktive E-Mail-Adressen aktivieren:
'  ********************************************************************
  Rng1.Select
'  ***************
  ' Suche nach "www":
  Selection.Find.ClearFormatting
  With Selection.Find
    .Text = "www.*."
    .Replacement.Text = ""
    .Forward = True
    ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
    Do While .Execute
      'If Selection.InRange(rng1) = False Then Exit Do
      If Selection.InRange(Rng1) = True Then
        ' Ganzen Ausdruck markieren (bis Leerzeichen oder Satzzeichen, auch Tabulator):
        Selection.MoveStartUntil CSet:=" ,!?;" & ChrW(13) & ChrW(160) _
          & vbTab, Count:=wdBackward
        Selection.MoveEndUntil CSet:=" ,!?;" & ChrW(13) & ChrW(160) & vbTab, Count:=wdForward
        '
        ' Wenn Absatzmarke in der Markierung, ist es kein Hyperlink:
        If InStr(Selection.Text, ChrW(13)) > 0 Then
          ' Absatzmarke in Markierung.
          Selection.Collapse Direction:=wdCollapseEnd
          ' Nur wenn im Hyperlink keine Sternchen stehen, Hyperlink erzeugen:
        ElseIf InStr(Selection.Text, "*") = 0 Then
          On Error Resume Next:
          ActiveDocument.Hyperlinks.Add _
              Anchor:=Selection.Range, _
              Address:=Selection.Text, _
              SubAddress:="", ScreenTip:="", TextToDisplay:=Selection.Text
          Selection.Collapse Direction:=wdCollapseEnd
          If Err.Number > 0 Then
            ActiveDocument.Undo 1
            Selection.Range.HighlightColorIndex = wdPink
            Selection.Collapse Direction:=wdCollapseEnd
          End If
          On Error GoTo 0
        Else
          ' Enthält Sternchen!
          Selection.Range.HighlightColorIndex = wdRed
          Selection.Collapse Direction:=wdCollapseEnd
        End If
      Else
        Selection.Collapse Direction:=wdCollapseEnd
      End If
    Loop
  End With
  ' ********************************
```

```
' Suche nach "http://"
Rng1.Select
Selection.Find.ClearFormatting
With Selection.Find
  ".Text = "http://[!w]*."
   .Text = "http://*."
  .Replacement.Text = ""
  .Forward = True
  ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
  .Format = True
  .MatchCase = False
  .MatchWholeWord = False
  .MatchAllWordForms = False
  .MatchSoundsLike = False
  .MatchWildcards = True
  Do While .Execute
     'If Selection.InRange(rng1) = False Then Exit Do
     If Selection.InRange(Rng1) = True Then
       ' Ganzen Ausdruck markieren:
       Selection.MoveEndUntil CSet:=" ,!;" & ChrW(13), Count:=wdForward
       '
       ' Wenn Absatzmarke in der Markierung, ist es kein Hyperlink:
       If InStr(Selection.Text, ChrW(13)) > 0 Then
         Selection.Collapse Direction:=wdCollapseEnd
       ' In einem Hyperlink darf kein Sternchen stehen:
       ElseIf InStr(Selection.Text, "*") = 0 Then
         ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
             Address:=Selection.Text, _
             SubAddress:="", ScreenTip:="", TextToDisplay:=Selection.Text
         Selection.Collapse Direction:=wdCollapseEnd
       Else
         Selection.Range.HighlightColorIndex = wdRed
         Selection.Collapse Direction:=wdCollapseEnd
       End If
    Else
       Selection.Collapse Direction:=wdCollapseEnd
    End If
  Loop
End With
' ***********************************
' Suche nach "https://"
Rng1.Select
Selection.HomeKey Unit:=wdStory
Selection.Find.ClearFormatting
With Selection.Find
  .Text = "https://[!w]*."
  .Replacement.Text = ""
  .Forward = True
  ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
  .Format = True
  .MatchCase = False
  .MatchWholeWord = False
  .MatchAllWordForms = False
  .MatchSoundsLike = False
  .MatchWildcards = True
  Do While .Execute
     If Selection.InRange(Rng1) = True Then
       ' Ganzen Ausdruck markieren:
       Selection.MoveEndUntil CSet:=" ,!;" & ChrW(13), Count:=wdForward
       '
       ' Wenn Absatzmarke in der Markierung, ist es kein Hyperlink:
       If InStr(Selection.Text, ChrW(13)) > 0 Then
```

```
            Selection.Collapse Direction:=wdCollapseEnd
          ' In einem Hyperlink darf kein Sternchen stehen:
          ElseIf InStr(Selection.Text, "*") = 0 Then
            ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
              Address:=Selection.Text, _
              SubAddress:="", ScreenTip:="", TextToDisplay:=Selection.Text
            Selection.Collapse Direction:=wdCollapseEnd
          Else
            Selection.Range.HighlightColorIndex = wdRed
            Selection.Collapse Direction:=wdCollapseEnd
          End If
        Else
         Selection.Collapse Direction:=wdCollapseEnd
        End If
      Loop
    End With
    ' ************************************
    ' Suche nach "ftp://"
    Rng1.Select
    Selection.Find.ClearFormatting
    With Selection.Find
      .Text = "ftp://[!w]*."
      .Replacement.Text = ""
      .Forward = True
      ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchAllWordForms = False
      .MatchSoundsLike = False
      .MatchWildcards = True
      Do While .Execute
        If Selection.InRange(Rng1) = False Then Exit Do
        ' Ganzen Ausdruck markieren:
        "Selection.MoveStartUntil Cset:=" ,!?;" & ChrW(13), Count:=wdBackward
        Selection.MoveEndUntil CSet:=" ,!;" & ChrW(13), Count:=wdForward
        '
        ' Wenn Absatzmarke in der Markierung, ist es kein Hyperlink:
        If InStr(Selection.Text, ChrW(13)) > 0 Then
          Selection.Collapse Direction:=wdCollapseEnd
        ' In einem Hyperlink darf kein Sternchen stehen:
        ElseIf InStr(Selection.Text, "*") = 0 Then
          ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
            Address:=Selection.Text, _
            SubAddress:="", ScreenTip:="", TextToDisplay:=Selection.Text
          Selection.Collapse Direction:=wdCollapseEnd
        Else
          Selection.Range.HighlightColorIndex = wdRed
          Selection.Collapse Direction:=wdCollapseEnd
        End If
      Loop
    End With
  ' ***************************************
  ' Suche nach E-Mails bzw. nach dem Zeichen "@":
    Rng1.Select
    Selection.Find.ClearFormatting
    With Selection.Find
      .Text = "[0-9A-Za-z]\@*.[A-Za-z][A-za-z]"
      .Replacement.Text = ""
      .Forward = True
      ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
      .Format = True
```

```
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = True
    Do While .Execute
      If Selection.InRange(Rng1) = False Then Exit Do
      ' Ganzen Ausdruck markieren:
      'Selection.MoveStartUntil Cset:=" ,!?;" & ChrW(13), Count:=wdBackward
      ' (gab Problem am Anfang der Dokumentkomponente)
      Selection.MoveStartWhile CSet:="0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÜ" _
        & "._-abcdefghijklmnopqrstuvwxyzäöüß", Count:=wdBackward
      'Selection.MoveEndUntil CSet:=" ,!;" & ChrW(13), Count:=wdForward
      Selection.MoveEndWhile CSet:="0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÜ" _
        & "._-abcdefghijklmnopqrstuvwxyzäöüß", Count:=wdForward
      ' In einem Hyperlink darf kein Sternchen stehen:
      If InStr(Selection.Text, "*") = 0 Then
        ' Für E-Mails Tabstopp bei 2 Zentimeter:
        Selection.ParagraphFormat.TabStops.ClearAll
        Selection.ParagraphFormat.TabStops.Add _
          Position:=CentimetersToPoints(2), _
          Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
        With Selection.Font
          .Color = wdColorBlue
          ' Hier war der Unterstrich nicht vom Leerzeichen zu unterscheiden.
          ' Deshalb jetzt doppelte Unterstreichung:
          .Underline = wdUnderlineDouble
          .UnderlineColor = wdColorGreen "wdColorBlue
        End With
        ' 11. August 2015.
        If Selection.Range.Hyperlinks.Count = 0 Then
          ActiveDocument.Hyperlinks.Add _
            Anchor:=Selection.Range, Address:=Selection.Text, _
            SubAddress:="", ScreenTip:="", TextToDisplay:=Selection.Text
        End If
        Selection.Collapse Direction:=wdCollapseEnd
      Else
        ' Sternchen in E-Mails sind unzulässig!
        Selection.Range.HighlightColorIndex = wdRed
        Selection.Collapse Direction:=wdCollapseEnd
      End If
    Loop
  End With
' ****
' Markierung wiederherstellen:
  Rng1.Select
' ************************************
' aktive Hyperlinks (inklusive E-Mail-Adressen) blau färben
' und unterstreichen,
' ausgenommen Inhaltsverzeichnis:
  If ActiveDocument.TablesOfContents.Count > 0 Then
    Set RngI = ActiveDocument.TablesOfContents(1).Range
  Else
    Selection.HomeKey Unit:=wdStory
    Set RngI = Selection.Range
  End If
' **************************
  For Each afield In Rng1.Fields
    If afield.Type = wdFieldHyperlink _
    And afield.Result.InRange(RngI) = False Then
      With afield.Result
        .Font.Color = wdColorBlue
```

```
        .Font.Underline = True
        .HighlightColorIndex = wdNoHighlight
      End With
    End If
  Next afield
  Rng1.Select
End Sub
' ----------------------------------------------------------------------------------------------------


Private Sub Hyperlinks_entlinken_und_faerben()
'  H. Chris Gast, 2011, 2012, 2014, 2015, 21. April 2016, 5. April 2017.
'  Alle Hyperlinks werden in der Markierung deaktiviert, aber die Unterstreichung
'  wird wieder hergestellt. E-Mails werden doppelt unterstrichen.
'
'  Da manchmal bei Hyperlinks Feldcode  und Feldergebnis voneinander abweichen,
'  ist es sinnvoll, beim Entlinken den Feldcode-Text dazuzusetzen:
'  Weicht der Link vom angezeigten Text ab, wird beides angezeigt,
'  sowohl Feldcode als auch ehemaliges Feldergebnis:
'  Muster: "http://www.din."  ((ursprüngliche Anzeige = "DIN-Webseite"))
'
'  Extras:
'  Leerzeichen in Hyperlink werden grün markiert
'  (eigentlich sind ja Leerzeichen in Internetadressen zu vermeiden).
'  "%20" ist der UTF8-Code für Leerzeichen, der im Makro decodiert wird.
'  ************
'  Markierung speichern:
Dim Rng1 As Range
  Set Rng1 = Selection.Range
Dim IntAbfrage As Integer
Dim Feldcode As String
Dim Feldergebnis As String
Dim Ergebnis As String
Dim afield As Field
'  ***********
'  Prozedur nur anwenden, wenn Hyperlinks vorhanden:
  If Rng1.Hyperlinks.Count = 0 Then Exit Sub
'  ************************************************
'  Hyperlinks aufheben, aber blaufärben und unterstreichen:
'  Bei Abweichung zwischen Feldergebnis und Feldcode wird beides angezeigt.
'  Jedoch werden die Wort "Hyperlink" und "Mailto:" entfernt,
'  ebenso ggf. der überflüssig gewordene Zusatz "\o" mit der Anzeigeoption.
'  ************************************************
'  Feldcode-Ansicht wählen, um das Wort "Hyperlink" zu suchen:
  ActiveWindow.View.ShowFieldCodes = True
  ' **************
  ' Inhaltsverzeichnis ausnehmen:
  Dim RngI As Range
  If ActiveDocument.TablesOfContents.Count > 0 Then
    Set RngI = ActiveDocument.TablesOfContents(1).Range
  Else
    Selection.HomeKey Unit:=wdStory
    Set RngI = Selection.Range
  End If
'  ********************************************************
  For Each afield In Rng1.Fields
    If InStr(afield.Code, "HYPERLINK") <> 0 _
    And afield.Result.InRange(RngI) = False Then
      ' Wenn Feldcode vom Feld-Ergebnis abweicht, beides anzeigen:
      Feldcode = afield.Code
      Feldcode = Replace(Feldcode, "HYPERLINK ", "")
      Feldcode = Trim(Replace(Feldcode, """", ""))
```

```
      Feldcode = Replace(Feldcode, "mailto:%20", "")
      Feldcode = Replace(Feldcode, "mailto:", "")
      Feldcode = Replace(Feldcode, "mailto: ", "")
      If InStr(Feldcode, "\o") > 4 Then _
         Feldcode = Left(Feldcode, InStr(Feldcode, "\o") - 1)
      Feldcode = Trim(Feldcode)
      ' 21.4.2016:
      Feldcode = Replace(Feldcode, "%20", " ")
      ' ********
      Feldergebnis = Trim(afield.Result)
      Ergebnis = "  ((ursprüngliche Anzeige = """ & Feldergebnis & """))"
      ' Verknüpfung löschen und Textstelle wieder blaufärben:
      afield.Result.Select
      afield.Unlink
      Selection.Font.Reset
      Selection.Text = Feldcode
      If InStr(Selection, "@") = 0 Then
        With Selection
          .Font.Color = wdColorBlue
          .Font.Underline = True
          '  .HighlightColorIndex = wdNoHighlight
        End With
      Else
        With Selection
          .Font.Color = wdColorBlue
          .Font.Underline = wdUnderlineDouble
          .Font.UnderlineColor = wdColorGreen
          '  .HighlightColorIndex = wdNoHighlight
        End With
      End If
      Selection.Collapse Direction:=wdCollapseEnd
      Selection.Font.Reset
      Selection.TypeText Text:="  "
      ' Ursprünglichen Anzeigetext, wenn er abwich, dahintersetzen
      ' (das Wort "Mailto" sollte dabei nicht erscheinen. 9. Juli 2015):
      ' If Feldcode <> Feldergebnis Then
      If Feldcode <> Replace(Feldergebnis, "mailto:", "") Then
        'Selection.InsertAfter Ergebnis
        Selection.Collapse Direction:=wdCollapseEnd
        With Selection
          .Text = Ergebnis
          .Font.ColorIndex = wdGreen
          .Font.Italic = True
          .Font.Name = "Arial"
          .Font.Size = 9
        End With
      End If
    End If
  Next afield
  ' **************
  ActiveWindow.View.ShowFieldCodes = False
' ****************************************
' Unterstrichene Unterstriche markieren:
  Rng1.Select
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
      .Text = "_"
      .Font.Underline = wdUnderlineDouble ' wdUnderlineSingle
      .Replacement.Text = "_"
      .Forward = True
      .Format = False
```

```
        ' .Replacement.Font.Underline = wdUnderlineSingle "wdUnderlineNone
        .Replacement.Font.UnderlineColor = wdColorRed
        .Replacement.Font.Bold = True
        '.Replacement.Highlight = True
        .MatchWholeWord = False
        .MatchWildcards = False
        .MatchSoundsLike = False
        .MatchAllWordForms = False
    End With
    Selection.Find.Execute Replace:=wdReplaceAll
    ' Bei doppelter Unterstreichung:
    Rng1.Select
    Selection.Find.ClearFormatting
    Selection.Find.Replacement.ClearFormatting
    With Selection.Find
        .Text = "_"
        .Font.Underline = wdUnderlineSingle
        .Replacement.Text = "_"
        .Forward = True
        .Format = False
        .Replacement.Font.Underline = wdUnderlineDouble
        ".Replacement.Font.UnderlineColor = wdColorRed
        ".Replacement.Font.Color = wdColorRed
        .Replacement.Font.Bold = True
        '.Replacement.Highlight = True
        .MatchWholeWord = False
        .MatchWildcards = False
        .MatchSoundsLike = False
        .MatchAllWordForms = False
    End With
    Selection.Find.Execute Replace:=wdReplaceAll
'  ****************
'  Unterstrichene Leerzeichen markieren:
    Rng1.Select
    Selection.Find.ClearFormatting
    Selection.Find.Replacement.ClearFormatting
    With Selection.Find
        .Text = " "
        .Font.Underline = wdUnderlineSingle
        .Replacement.Text = " "
        .Forward = True
        .Format = False
        " .Replacement.Font.Underline = wdUnderlineDouble ' wdUnderlineNone
        " .Replacement.Font.UnderlineColor = wdColorRed
        .Replacement.Highlight = True
        .MatchWholeWord = False
        .MatchWildcards = False
        .MatchSoundsLike = False
        .MatchAllWordForms = False
    End With
    Selection.Find.Execute Replace:=wdReplaceAll
'  Doppelt unterstrichene Leerzeichen:
    Rng1.Select
    Selection.Find.ClearFormatting
    Selection.Find.Replacement.ClearFormatting
    With Selection.Find
        .Text = " "
        .Font.Underline = wdUnderlineDouble
        .Replacement.Text = " "
        .Forward = True
        .Format = False
        " .Replacement.Font.Underline = wdUnderlineDouble ' wdUnderlineNone
```

```
      '' .Replacement.Font.UnderlineColor = wdColorRed
      .Replacement.Highlight = True
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
'  ************************************
  Rng1.Select
End Sub
' ----------------------------------------------------------------------------------------------------
```

22.2 Überschriftformate zuordnen je nach Schriftgröße im Absatz

```
' Modul "OCR_Gliederung"

Option Explicit

Private strFontMax
Private j2max
Private strMsgStatistik


Sub OCR_Gliederung_einführen()
' H. Chris Gast, Stand 22. Januar 2017.
' Das Makro prüft, ob es eine Abschnittsgliederung mit Formatvorlagen gibt.
' Wenn nein, wie in mit Texterkennung (OCR) erzeugten Dateien oder bei
' rückkonvertierten Pdf-Dateien, wird eine neue Datei erzeugt, in der allen Absätzen
' mit größerer Schriftgröße jeweils Überschrifts-Formatvorlagen zugewiesen werden.
'
' Dazu wird zuerst in einer Unterprozedur geprüft, welches die häufigste Schriftgröße
' und welches die häufigste Schriftart ist, und dies wird der Formatvorlage "Standard"
' zugewiesen, ohne aber den zugehörigen Text zu verändern.
'
' Dann werden allen Absätze, deren Schriftgrößer größer als Standard ist, je nach
' Schriftgröße die Überschrifts-Formatvorlagen "Überschrift1" (rote Farbe) bzw. "Überschrift2"
' (blaue Farbe) zugewiesen. Absätzen in Standard-Schriftgröße, die mit "Zahl-Punkt-Zahl"
' anfangen und deren Zeichen fett sind, erhalten die Formatvorlage "Überschrift3"
' (lila Farbe).
' Fängt ein Absatz der Überschrift1 mit dem Wort "Anhang" an, wird er dunkelrot gefärbt.
' Hier sind ggf. von Hand Absatzmarken durch Zeilenumbrüche zu ersetzen.
'
' In einer Unterprozedur werden Absätzen, die mit dem Wort "ANMERKUNG" oder "Anmerkung",
' anfangen, die Formatvorlage "Note" zugewiesen, die ggf. vorher neu erzeugt wird.
'
' Noch zu verbessern:
' Es bleibt die Frage, ob man Text in Tabellen ausnehmen sollte.
' Bei Absätzen mit Schriftgröße größer als 18 Punkt wäre vielleicht die Formatvorlage "Titel"
' sinnvoller.
' ******
' Die hier aufgerufene Funktion entspricht weitgehend der Funktion im Makro
' für Schlüsselwörtersuche, allerdings wird hier schon ab Seite 1 geprüft, und es
' werden hier auch größere Schriftgrößen berücksichtigt als bei Normen.
' Ferner wird dabei die Formatvorlage "Standard" an die häufigste Schriftgröße angepasst.
' ************************************************
  Dim i As Integer
  i = Fkt_UEberschriften
  If i = 1 Then
    ' Datei bereits mit Überschrifts-Ebenen:
    MsgBox "Die Datei hat bereits Überschrifts-Ebenen, das Makro wird abgebrochen."
  ElseIf i = 2 Then
    ' Neue Datei mit neuen Überschrifts-Ebenen:
    MsgBox strMsgStatistik & vbCrLf & vbCrLf _
    & "Es wurden in der Datei Überschrifts-Ebenen eingeführt." & vbCrLf _
    & "Zu Prüfzwecken wurden die Überschriftstexte eingefärbt."
  Else
    ' Datei ohne Überschrifts-Ebenen:
    MsgBox strMsgStatistik & vbCrLf & vbCrLf _
    & "Es konnten keine Überschrifts-Ebenen ergänzt werden."
  End If
End Sub
' ---------------------------------------------------------------------------------------------------
```

```
Private Function Fkt_UEberschriften() As Integer
' H. Chris Gast, 2014, Juni, September, Oktober 2015, 22. Januar 2017, 15. Januar 2018.
' Zuweisung von Überschrifts-Formatvorlagen (Styles) in Dokumenten,
' die aus Pdf zurückkonvertiert worden sind.
'
' Häufigste Schriftgröße = Standard (wird mit Unterprozedur bestimmt).
' Zuweisung von Überschrift1 bei Größe "Standard + 2" bis Größe 36 Punkt;
' dazu rote Schrift. Wenn das Wort "Anhang" vorkommt, dunkelrote Schrift.
' Überschrift2 bei "Standard + 1 Punkt", blaue Schrift.
' Überschrift3 bei fettem Absatz + "Zahl-Punkt-Zahl" am Absatz-Anfang, violette Schrift.
' Folgen auf einen Anhangstitel mehrere Zeilen mit dergleichen Schriftgröße, werden diese
' ebenfalls dunkelrot gefärbt, erhalten aber keine Überschrifts-Formatvorlage.
'
' Damit sich die Seitenumbrüche und Seitenzahlen nicht verändern,
' müssen bei neu zugewiesenen Überschrifts-Formatvorlagen die Zeilenabstände auf Null
' gesetzt werden. Ferner bleibt die Schriftgröße erhalten. Die hier zugewiesenen Schriftfarben
' kann der Anwender ja leicht nach einer kurzen Blick in den Text wieder beseitigen.
'
' Es sollen hier bei der Font-Größe auch nicht-ganze Zahlen erfasst werden,
' aber nur ",0" und ",5" als Nachkommastellen, so dass mit einem Trick
' die Zählung mit Integer-Zahlen möglich ist:
' Integer-Werte gehen von -32 768 bis 32 767 und haben 2 Byte.
' Damit sich das Makro in geschachtelten Tabellen nicht verfängt, muss bei der Suche stets
' nach der Formatvorlage "wdStyleNormal" gesucht werden (15. Januar 2018).
'
' In einer Unterprozedur wird allen Abschnitten, die  mit dem Wort "ANMERKUNG..." beginnen,
' die Formatvorlage "Note" zugewiesen.
' ***********************************************
Dim i As Integer
Dim Abfrage As Integer
Dim MsgBoxAnzeige As String
Fkt_UEberschriften = 0  ' Dieser Wert bleibt Null, wenn keine Überschriften gefunden wurden.
' ***********************************************
' Prüfen, ob eine der Überschriftsebenen 1 bis 3 vorhanden ist:
  Selection.Find.ClearFormatting
  Selection.Find.Style = ActiveDocument.Styles("Überschrift 1")
  With Selection.Find
    .Text = ""
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .Highlight = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute
  If Selection.Find.Found = True Then
    Fkt_UEberschriften = 1
    Exit Function
  End If
  '
  Selection.Find.ClearFormatting
  Selection.Find.Style = ActiveDocument.Styles("Überschrift 2")
  With Selection.Find
    .Text = ""
    .Replacement.Text = ""
```

```
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .Highlight = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute
  If Selection.Find.Found = True Then
    Fkt_UEberschriften = 1
    Exit Function
  End If
  ' ***********
  Selection.Find.ClearFormatting
  Selection.Find.Style = ActiveDocument.Styles("Überschrift 3")
  With Selection.Find
    .Text = ""
    .Replacement.Text = ""
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .Highlight = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute
  If Selection.Find.Found = True Then
    Fkt_UEberschriften = 1
    Exit Function
  End If
' ***************************************************************
' ***************************************************************
' Abfrage, ob Überschrifts-Formatvorlagen eingeführt werden sollen:
  MsgBoxAnzeige = "Dieses Dokument enthält keine Überschrifts-Formate. " & vbCrLf _
     & "Sollen diese eingeführt werden? " & vbCrLf _
     & "Dann erfolgt Speicherung als neue Datei."
  Abfrage = MsgBox(MsgBoxAnzeige, vbYesNoCancel + vbDefaultButton2, "Abschnittsgliederung")
    ' Bei "Ja" ist Abfrage = 6, bei "Nein" = 7, bei Abbrechen =2.
  If Abfrage = 7 Then
    Exit Function
  ElseIf Abfrage = 2 Then
    End
  End If
' ***************************************************************
' Datei speichern und als neue Word-Datei weiterführen:
  If Left(ActiveDocument.Name, 5) <> "!mit_Überschriftsformaten!" Then
    ActiveDocument.SaveAs "!mit_Überschriftsformaten!_" & ActiveDocument.Name
  Else
    ActiveDocument.Save
  End If
' ***************************************************************
' Schriftart- und Schriftgrößenstatistik aufrufen:
  Call Schriftstatistik_anhand_Absaetzen
  ' Ergibt strFontmax = Name der häufigsten Schriftart
  ' und j2max = doppelte Standardschriftgröße.
  ' Achtung: Standardformat erst ganz zum Schluss ändern!
```

```
' *************************************************************
' Überschrifts-Formatvorlagen einfügen in neuer Datei
' bei allen Absätzen mit größeren Zeichen:
' *************************************************************
' Anhangstitel bzw. große Überschriften mit Formatvorlage "Überschrift 1" versehen:
  For i = (j2max + 4) To 72 ' von Standard + 2 bis 36 Punkt in Halb-Schritten
    Selection.HomeKey Unit:=wdStory
    Selection.Find.ClearFormatting
    Selection.Find.Font.Size = i / 2
    Selection.Find.Style = "Standard"
    With Selection.Find
      .Text = "^$^$^?"  ' Soll Text enthalten, nicht nach Absatzmarken suchen!
      .Replacement.Text = ""
      .Forward = True
      ' .Wrap = wdFindContinue
      .Style = ActiveDocument.Styles(wdStyleNormal)
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False ' True
      .MatchSoundsLike = False
      .MatchAllWordForms = False
      Do While .Execute
        Selection.Paragraphs(1).Range.Select
        Selection.Style = "Überschrift 1"
        "If Left(Selection, 6) = "Anhang" _
        "Or Left(Selection, 6) = "ANHANG" _
        "Or Left(Selection, 5) = "ANNEX" Then
        ' Es gibt auch "Nationaler Anhang" usw.
        If InStr(Selection, "Anhang") > 0 _
        Or InStr(Selection, "ANHANG") > 0 _
        Or InStr(Selection, "ANNEX") > 0 Then
         ' Anhangstitel, hier Überschrift 1, dunkelrot gefärbt:
          Selection.Font.ColorIndex = wdDarkRed 'wdGreen
          Selection.Font.Name = strFontMax
          Selection.Font.Size = i / 2
          Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
          With Selection.ParagraphFormat
            .SpaceBefore = 0
            .SpaceAfter = 0
          End With
          ' Ein Anhangstitel hat meist mehrere Zeilen mit dergleichen Schriftgröße,
          ' die hier nicht als separate Überschriften "1" erfasst werden sollen:
          ' sondern wie der Anhangstitel ebenfalls dunkelrot gefärbt werden:
          If Selection.Next.Paragraphs(1).Range.Font.Size = i / 2 Then
            Selection.Next.Paragraphs(1).Range.Font.ColorIndex = wdDarkRed
            Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
          End If
          Selection.Collapse Direction:=wdCollapseEnd
          Selection.MoveDown Unit:=wdParagraph, Count:=1
          If Selection.Next.Paragraphs(1).Range.Font.Size = i / 2 Then
            Selection.Next.Paragraphs(1).Range.Font.ColorIndex = wdDarkRed
            Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
          End If
          Selection.Collapse Direction:=wdCollapseEnd
          Selection.MoveDown Unit:=wdParagraph, Count:=1
          If Selection.Next.Paragraphs(1).Range.Font.Size = i / 2 Then
            Selection.Next.Paragraphs(1).Range.Font.ColorIndex = wdDarkRed
            Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
          End If
          Selection.Collapse Direction:=wdCollapseEnd
          Selection.MoveDown Unit:=wdParagraph, Count:=1
```

```
            If Selection.Next.Paragraphs(1).Range.Font.Size = i / 2 Then
              Selection.Next.Paragraphs(1).Range.Font.ColorIndex = wdDarkRed
              Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
            End If
            Selection.Collapse Direction:=wdCollapseEnd
            Selection.MoveDown Unit:=wdParagraph, Count:=1
            ' *********
          Else
            ' Normale Überschrift 1, hier rot gefärbt:
            Selection.Font.ColorIndex = wdRed  ' rot
            Selection.Font.Name = strFontMax
            Selection.Font.Size = i / 2
            Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter
          End If
          ' Damit der Seitenumbruch sich nicht ändert:
          With Selection.ParagraphFormat
            .SpaceBefore = 0
            .SpaceAfter = 0
          End With
          ' Zum Weitersuchen:
          Selection.Collapse Direction:=wdCollapseEnd
        Loop
      End With
    Next i
'  ************************************
'  Formatvorlage "Überschrift2" für Überschriften mit der Größe (Standard + 1)
'  bis (Standard + 1,5):
    For i = (j2max + 2) To (j2max + 3)
      Selection.HomeKey Unit:=wdStory
      Selection.Find.ClearFormatting
      Selection.Find.Font.Size = i / 2
      With Selection.Find
        ".Text = "[A-z0-9]"    ' Keine Absatzmarken suchen!
        .Text = "^$^$^?"  ' Soll Text enthalten, nicht nach Absatzmarken suchen!
        .Replacement.Text = ""
        .Forward = True
        ' .Wrap = wdFindContinue
        .Style = ActiveDocument.Styles(wdStyleNormal)
        .Format = True
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = False ' True
        .MatchSoundsLike = False
        .MatchAllWordForms = False
        Do While .Execute
          Selection.Paragraphs(1).Range.Select
          "If Selection.Font.Bold = True Then
          Selection.Style = "Überschrift 2"
          Selection.Font.ColorIndex = wdBlue 'blau
          Selection.Font.Name = strFontMax
          Selection.Font.Size = i / 2
          With Selection.ParagraphFormat
            .SpaceBefore = 0
            .SpaceAfter = 0
          End With
          "End If
          Selection.Collapse Direction:=wdCollapseEnd
        Loop
      End With
    Next i
'  ****************************************
'  Überschrift 3 für fette Absätze, die mit "Zahl-Punkt-Zahl" beginnen (violett):
```

```
  For i = j2max To j2max + 1   ' Standard oder (Standard + 0,5 Punkt)
    Selection.HomeKey Unit:=wdStory
    Selection.Find.ClearFormatting
    ' Selection.Find.Style = "Standard"
    Selection.Find.Font.Bold = True
    Selection.Find.Font.Size = i / 2 ' Überschrift 3
    With Selection.Find
      .Text = "[0-9].[0-9]"  ' Suche nur nach "Zahl-Punkt-Zahl" wie etwa "12.3.4"
      .Replacement.Text = ""
      .Forward = True
      ' .Wrap = wdFindContinue
      .Style = ActiveDocument.Styles(wdStyleNormal)
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = True
      .MatchSoundsLike = False
      .MatchAllWordForms = False
      Do While .Execute
        Selection.Paragraphs(1).Range.Select
        ' Der Absatz muss aus mehr Zeichen bestehen als nur der Zahl:
        If Len(Selection) > 3 Then
          ' Die ersten beiden Ziffer dürfen auch Buchstaben sein.
          If Val(Selection) > 0 _
          Or Val(Mid(Selection, 2)) > 0 _
          Or Val(Mid(Selection, 3)) > 0 Then
            Selection.Style = "Überschrift 3"
            Selection.Range.ListFormat.RemoveNumbers _
                NumberType:=wdNumberParagraph
            Selection.Font.ColorIndex = wdViolet  ' violett
            "Selection.Range.HighlightColorIndex = wdDarkYellow 'dunkelgelb
            Selection.Font.Name = strFontMax
            Selection.Font.Size = i / 2
            With Selection.ParagraphFormat
              .SpaceBefore = 0
              .SpaceAfter = 0
            End With
          End If
        End If
        Selection.Collapse Direction:=wdCollapseEnd
        If Selection.End = ActiveDocument.Range.End - 1 Then Exit Do
      Loop
    End With
  Next i
' ****************************************************
' Zum Schluss der Formatvorlage (Style) "Standard" die häufigste Größe und die
' häufigste Schriftart zuweisen, ohne im Text etwas zu ändern:
  With ActiveDocument.Styles(wdStyleNormal).Font
    .Name = strFontMax
    .Size = j2max / 2
  End With
' ****************************************************
' Formatvorlage "Note" hinzufügen
' und allen Anmerkungen den Style "Note" zuweisen:
  Call Anmerkungen_Formatzuweisung
' ********************************************************************
' Erfolgskontrolle:
' ********************************************************************
' Prüfen, ob jetzt eine Überschriftsformatvorlage 1 vorhanden ist:
  Selection.Find.ClearFormatting
  Selection.Find.Style = ActiveDocument.Styles("Überschrift 1")
  With Selection.Find
```

```
      .Text = ""
      .Replacement.Text = ""
      .Forward = True
      .Wrap = wdFindContinue
      .Format = True
      .Highlight = False
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
    End With
    Selection.Find.Execute
    If Selection.Find.Found = True Then
      Fkt_UEberschriften = 2
      Exit Function
    End If
  ' *******
  ' Prüfen, ob jetzt eine Überschriftsformatvorlage 2 vorhanden ist:
    Selection.Find.ClearFormatting
    Selection.Find.Style = ActiveDocument.Styles("Überschrift 2")
    With Selection.Find
      .Text = ""
      .Replacement.Text = ""
      .Forward = True
      .Wrap = wdFindContinue
      .Format = True
      .Highlight = False
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
    End With
    Selection.Find.Execute
    If Selection.Find.Found = True Then
      Fkt_UEberschriften = 2
      Exit Function
    End If
  ' ********************
  ' Prüfen, ob jetzt eine Überschriftsformatvorlage 3 vorhanden ist:
    Selection.Find.ClearFormatting
    Selection.Find.Style = ActiveDocument.Styles("Überschrift 1")
    With Selection.Find
      .Text = ""
      .Replacement.Text = ""
      .Forward = True
      .Wrap = wdFindContinue
      .Format = True
      .Highlight = False
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
    End With
    Selection.Find.Execute
    If Selection.Find.Found = True Then
      Fkt_UEberschriften = 2
    End If
  End Function
  '-----------------------------------------------------------------------------------------------------------
```

```
Private Sub Schriftstatistik_anhand_Absaetzen()
' H. Chris Gast, 25. Juni 2015 + 19. Oktober 2015, 13. April 2016, 20. Januar 2017.
' Absätze nach Größe sortiert zählen.
' Es müssen bei der Größe auch nicht-ganze Zahlen erfasst werden,
' aber nur ",0" und ",5" als Nachkommastellen, so dass mit einem Trick
' die Zählung mit Integer-zahlen möglich ist:
' Integer geht nur von -32 768 bis 32 767, (2 Byte).
' Wenn das nicht reicht, ist "Long" zuverwenden (4 Byte).
'
'  Es werden für die Auswertung die Schriftgrößen von 7 bis 36 Punkt erfasst.
'  ************************************************************
'  Bildschirmaktualisierung abstellen.
  StatusBar = "Bitte warten - das Makro läuft."     '  Geht nicht immer.
  Application.DisplayStatusBar = False
'  Problem:  Die StatusBar-Anzeige wird sehr schnell wieder überschrieben.
  Application.ScreenUpdating = False
  Options.Pagination = False
'  Scrollbars ausblenden:
  With ActiveWindow
    .DisplayHorizontalScrollBar = False
    .DisplayVerticalScrollBar = False
  End With
'  ************************************************************
Dim i As Integer ' oder Long?
Dim j As Integer
Dim iMax As Integer: iMax = 0    ' oder Long?
"Dim j2max As Integer
"Dim iArial As Integer: iArial = 0
"Dim iCambria As Integer: iCambria = 0
"Dim iTNR As Integer: iTNR = 0
"Dim strFontMax As String
Dim strMessage As String: strMessage = _
   "Es wurden folgende Absätze der Größen 7 bis 36 Punkt gefunden:"
Dim strMessageFont As String
Dim AxF() As String  ' Array
Dim strFontliste As String
'  **********
  For j = 14 To 72 '40
    ' Absätze mit Größe j/2 zählen:
    i = 0
    Selection.HomeKey Unit:=wdStory
    Selection.Find.ClearFormatting
    With Selection.Find
      .Text = "^?^p"   ' "^?^?^p"
      .Font.Size = j / 2
      .Replacement.Text = ""
      .Forward = True
      ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchAllWordForms = False
      .MatchSoundsLike = False
      .MatchWildcards = False
      Do While .Execute
        i = i + 1
      Loop
    End With
    If i > 0 Then
      strMessage = strMessage _
```

```
      & vbCrLf & "    " & i & " Absatz/Absätze der Größe " & j / 2 & "."
      If i > iMax Then
         iMax = i  ' Anzahl der Absätze mit der häufigsten Schriftgröße
         j2max = j  ' (j/2= häufigste Schriftgröße)
      End If
    End If
  Next j
  ' ***
  ' Abbruchbedingungen:
  If j2max = 0 Then
  '  Nichts gefunden:
    strMessage = "Es wurden keine Absätze mit Größen zwischen 7 und 36 Punkt gefunden!"
    MsgBox strMessage, vbCritical
    End ' Exit Sub
  End If
  If iMax < 10 Then
  '  Zuwenig Fundstellen:
    strMessage = _
    "Es wurden nur " & iMax & " Absätze mit der häufigsten Schriftgröße gefunden. " _
    & "Das ist nicht ausreichend, um eindeutig die Formatvorlage 'Standard' zu bestimmen. " _
    & "Das Makro wird abgebrochen, ohne die Formatvorlage 'Note' zu erzeugen."
    MsgBox strMessage, vbCritical
    End 'Exit Sub
  End If
  ' ***
  ' Zwischenergebnis:
  strMessage = strMessage _
  & vbCrLf & "Die häufigste Schriftgröße ist " & j2max / 2 & " Punkt."
'  ***************************************
  '  Fonts-Häufigkeit, 19.10.2015:
  Selection.HomeKey Unit:=wdStory
  Selection.Find.ClearFormatting
  With Selection.Find
    .Text = "^?^p"
    .Font.Size = j2max / 2
    .Replacement.Text = ""
    .Forward = True
    ' .Wrap = wdFindContinue   (führt zu Endlosschleife!)
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchAllWordForms = False
    .MatchSoundsLike = False
    .MatchWildcards = False
    Do While .Execute
      If InStr(strFontliste, Selection.Font.NameAscii) = 0 Then
        strFontliste = strFontliste & ", 1ä" & Selection.Font.NameAscii
      Else
        strFontliste = Replace(strFontliste, "ä" & Selection.Font.NameAscii, _
          "1ä" & Selection.Font.NameAscii)
      End If
    Loop
  End With
  ' ***
'  Fonts nach Häufigkeit sortieren mittels Array:
  '  Array erzeugen:
  'Dim strÜ
  AxF = Split(strFontliste, ",")
  ' Sortierbefehl geht nicht bei zuvielen einsen.
    For i = 0 To UBound(AxF)
      If AxF(i) <> "" Then _
      AxF(i) = CStr(Format(InStr(AxF(i), "ä"), "00000")) & Mid(AxF(i), InStr(AxF(i), "ä"))
```

```
    If i > 5 Then Exit For
  Next
  ' ***
  ' Sortieren:
  WordBasic.SortArray AxF(), 1
  ' ***
  ' String zurücksetzen:
  strFontliste = ""
  '  Ausgabe der 5 häufigsten Ergebnisse
  '  (aber nur, wenn kein Pluszeichen enthalten ist wie in "+Überschriften"):
  For i = 0 To UBound(AxF)
    If InStr(AxF(i), "+") = 0 And AxF(i) <> "" Then _
    strFontliste = strFontliste & Mid(AxF(i), InStr(AxF(i), "ä") + 1) & ", "
    If i > 5 Then Exit For
  Next
  ' Die Einsen nach der Sortierung wieder entfernen:
 '' strFontliste = Replace(strFontliste, "1", "")
  If strFontliste = "" Then
    MsgBox "Die Schriftart konnte nicht bestimmt werden!" _
    & vbCrLf & """" & Replace(AxF(1), "1", "") & """" _
    & vbCrLf & "Es wird hilfsweise 'Arial' zugeordnet.", vbCritical
    '' Exit Sub
    strFontliste = "Arial"
  End If
  strFontliste = Left(strFontliste, Len(strFontliste) - 2)
  ' Die erste Schriftart ist die häufigste:
  If InStr(strFontliste, ",") > 0 Then
    strFontMax = Left(strFontliste, InStr(strFontliste, ",") - 1)
    ' Zwischenergebnis:
     strMessageFont = _
     "Die häufigsten Schriftarten der häufigsten Schriftgröße sind: " & vbCrLf _
     & "   " & strFontliste & "." & vbCrLf _
     & "Die häufigste Schriftart ist: " & strFontMax & "." & vbCrLf
  Else
    ' Es wurde nur eine brauchbare Schriftart gefunden:
    strFontMax = Trim(strFontliste)
    ' Zwischenergebnis:
    strMessageFont = _
      "Die häufigste Schriftart ist: " & strFontMax & "." & vbCrLf
  End If
'  ************************************
  '''  kurze Warteschleife  für Word 2010:
  ''Dim tt As Single
  ''   tt = Timer
  ''   Do While Timer < tt + 0.1  'Sekunden
  ''      DoEvents
  ''   Loop
'  ***********************************************************************
'  ursprüngliche Bildschirmansicht wiederherstellen
  StatusBar = ""
  Application.DisplayStatusBar = True
  Options.Pagination = True
  Application.ScreenUpdating = True
  Application.DisplayStatusBar = True
'  Scrollbalken wieder einschalten:
  With ActiveWindow
    .DisplayHorizontalScrollBar = True
    .DisplayVerticalScrollBar = True
  End With
'  ****************************
'  Ergebnis ausgeben:
  strMsgStatistik = strMessage & vbCrLf _
```

```
    & vbCrLf & strMessageFont _
    & vbCrLf & "Der Formatvorlage 'Standard' wurde jetzt die Schriftart '" _
    & strFontMax & "'" & vbCrLf & "und die Schriftgröße " & j2max / 2 & " Punkt zugewiesen."
End Sub
' ----------------------------------------------------------------------------------------------------


Private Sub Anmerkungen_Formatzuweisung()
' H. Chris Gast, September, Oktober 2015, 11. Januar 2016, 20. Januar 2017.
' Formatvorlage (Style) "Note" erzeugen, wenn noch nicht vorhanden.
' Dann allen Absätzen, die mit dem Wort "ANMERKUNG" (in Großbuchstaben)
' oder mit "Anmerkung:" (mit Doppelpunkt dahinter) beginnen,
' diese Formatvorlage "Note" zuweisen.
' Die Suche nach einer Schriftgröße kleiner als "Standard" hat sich nicht bewährt.
' Für englische Texte müsste nach "NOTE" gesucht werden (hier nicht ausgeführt).
' ******************************
On Error Resume Next
  Application.DisplayAlerts = wdAlertsNone
  ActiveDocument.Styles.Add _
    Name:="Note", _
    Type:=wdStyleTypeParagraph
  ' ***
  With ActiveDocument.Styles("Note")
    .AutomaticallyUpdate = False
    .NextParagraphStyle = "Standard"
    '' .BaseStyle = "Standard"
    .Font.Size = ActiveDocument.Styles(wdStyleNormal).Font.Size - 1
    .Font.Name = ActiveDocument.Styles(wdStyleNormal).Font.Name
  End With
  Application.DisplayAlerts = wdAlertsAll
  ' ***
  With ActiveDocument.Styles("Note").ParagraphFormat
    .LeftIndent = CentimetersToPoints(0)
    .RightIndent = CentimetersToPoints(0)
    .SpaceBefore = 0
    .SpaceBeforeAuto = False
    .SpaceAfter = 6
    .SpaceAfterAuto = False
    .LineSpacingRule = wdLineSpaceSingle
    .Alignment = wdAlignParagraphLeft
    .WidowControl = False
    .KeepWithNext = False
    .KeepTogether = False
    .PageBreakBefore = False
    .NoLineNumber = False
    .Hyphenation = True
    .FirstLineIndent = CentimetersToPoints(0)
    .OutlineLevel = wdOutlineLevelBodyText
    .CharacterUnitLeftIndent = 0
    .CharacterUnitRightIndent = 0
    .CharacterUnitFirstLineIndent = 0
    .LineUnitBefore = 0
    .LineUnitAfter = 0
    .MirrorIndents = False
    .TextboxTightWrap = wdTightNone
  End With
'  ****************
  On Error GoTo 0
'  *****************
'  0,1 Sekunden warten:
  Dim tt As Single
  tt = Timer
```

```
  Do While Timer < tt + 0.1  ' Zeit in Sekunden
    DoEvents
  Loop
' ****************
  Selection.HomeKey Unit:=wdStory
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "ANMERKUNG"
    '' .Font.Size = j2max/2 - 1
    .Replacement.Text = ""
    .Forward = True
    '.Wrap = wdFindContinue
    .Format = True
    '.Highlight = False
    .MatchCase = True
    .MatchWholeWord = True
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
    Do While .Execute
      If InStr(Selection.Paragraphs(1).Range.Text, "ANMERKUNG") < 2 Then
        Selection.Range.Paragraphs(1).Range.Select
        Selection.Style = ActiveDocument.Styles("Note")
        'Selection.Range.HighlightColorIndex = wdBrightGreen
        Selection.Font.ColorIndex = wdGreen
        Selection.Font.Size = j2max / 2 - 1
      End If
      Selection.Collapse Direction:=wdCollapseEnd
    Loop
  End With
' ************
  Selection.HomeKey Unit:=wdStory
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
    .Text = "Anmerkung:"
    '' .Font.Size = j2max/2 - 1
    .Replacement.Text = ""
    .Forward = True
    '.Wrap = wdFindContinue
    .Format = True
    '.Highlight = False
    .MatchCase = True
    .MatchWholeWord = True
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
    Do While .Execute
      If InStr(Selection.Paragraphs(1).Range.Text, "Anmerkung:") < 2 Then
        Selection.Range.Paragraphs(1).Range.Select
        Selection.Style = ActiveDocument.Styles("Note")
        'Selection.Range.HighlightColorIndex = wdBrightGreen
        Selection.Font.ColorIndex = wdGreen
        Selection.Font.Size = j2max / 2 - 1
      End If
      Selection.Collapse Direction:=wdCollapseEnd
    Loop
  End With
' ***************
  ActiveDocument.Save
End Sub
```

```
' ----------------------------------------------------------------------------------------------------
```

22.3 Rechtschreibprüfung nach OCR

Die Korrekturen durch diese Rechtschreibprüfung sind erst im Ansatz verwirklicht. Korrekturen werden dunkelgelb markiert.

```
Option Explicit


Sub Rechtschreibfehler_in_Markierung_suchen_teils_korrigieren()
' H. Chris Gast, Stand 8. Februar 2017.
' Markiert falsche Wörter (rot).
' Repariert einige Wörter, aber noch nicht ausreichend (dunkelgelb).
'
' Normale Rechtschreibprüfung per VBA funktioniert nur bei Deutscher Sprache!
' Nicht-deutsche Wörter werden durch eine spezielle Unterprozedur geprüft.
Dim rngWort As Range
Dim IntFremdsprache As Long
   IntFremdsprache = 0
Dim IntFehlerfund As Integer
   IntFehlerfund = 0
Dim strTestwort As String
Dim IntKorrektur As Integer
   IntKorrektur = 0
' Markierung als Range speichern:
Dim Rng1 As Range
  Set Rng1 = Selection.Range
Dim strProtokoll As String
'  ********************************
'  Prüfen, ob etwas markiert ist:
  If Selection.Type < 2 Then
     MsgBox "Es ist nichts markiert!"
     Exit Sub
  End If
'  Markierung als Range speichern:
  Set Rng1 = Selection.Range
'''''''
  WordBasic.DisableAutoMacros 1     ' schaltet Automakros (etwa für Briefkopf) aus.
'  ********************************
On Error Resume Next ''Goto Zeile_Fehler
' *********************************
  For Each rngWort In Selection.Range.Words
    rngWort.Select
    If rngWort.Underline <> 0 Then
      ' nichts tun, bei Hyperlinks nicht prüfen!
    ' ***********
    ElseIf Left(Languages(rngWort.LanguageID), 7) <> "Deutsch" Then
      ' bei Fremdsprachen aufwendigeres Prüfverfahren.
      If Err.Number = 0 Then
        ''IntFremdsprache = IntFremdsprache + 1
        Call CheckSpelling_Fremdsprachen
        If Selection.Range.HighlightColorIndex = wdRed Then
          IntFehlerfund = IntFehlerfund + 1
        End If
      Else
        ' Fehler, wenn die Markierung keine eindeutige Sprachzuordnung hat:
        Selection.Range.HighlightColorIndex = wdTurquoise
        IntFremdsprache = IntFremdsprache + 1
      End If
    ' ***********
    ElseIf Application.CheckSpelling(Word:=rngWort.text) = False Then
      rngWort.Select
      ' Silbentrennzeichen macht bei Word 2010 Probleme, muss zur Prüfung gelöscht sein:
```

```
      If Application.CheckSpelling(Word:=Replace(Selection.text, ChrW(31), "")) = False Then
        '' rngWort.HighlightColorIndex = wdRed
        ' ***
        ' Erweiterung der Markierung, falls das Wort hinter einem Bindestrich weitergeht:
        If Right(Selection, 1) = "-" Then
          Selection.MoveEndUntil Cset:=" ,.-" & ChrW(13) & ChrW(11), Count:=wdForward
        End If
        ' Unterprozedur aufrufen:
        Call Einzelwort_Rechtschreibkorrektur
        ' Fehler und Korrekturen separat zählen:
        If Selection.Range.HighlightColorIndex = wdRed Then
          IntFehlerfund = IntFehlerfund + 1
        ElseIf Selection.Range.HighlightColorIndex = wdDarkYellow Then
          IntKorrektur = IntKorrektur + 1
        Else
          ' Wort ist nach Korrekturen ok.
        End If
        ' ***
      End If
    ' ***********
    Else
      ' Das Wort ist richtig geschrieben (keine Markierung).
    End If
    Err.Clear
  Next rngWort
' ******************************
' Auswertung:
  strProtokoll = ""
  If IntFehlerfund > 0 Then
    strProtokoll = strProtokoll _
    & IntFehlerfund & " gefundene(r) Rechtschreibfehler wurde(n) rot markiert." _
    & vbCrLf
  End If
  If IntKorrektur > 0 Then
    strProtokoll = strProtokoll _
    & "Es wurde(n) " & IntKorrektur & " dunkelgelb markierte(n) Stelle(n) korrigiert." _
    & vbCrLf
  End If
  If strProtokoll = "" Then
    strProtokoll = strProtokoll & "Es wurden keine Rechtschreibfehler gefunden." _
    & vbCrLf
  End If
  If IntFremdsprache > 0 Then
    strProtokoll = strProtokoll _
    & "Die " & IntFremdsprache _
    & " hellblau markierten Wörter konnten nicht auf Rechtschreibung geprüft werden. "
  End If
' ******************************
' Markierung wiederherstellen:
  Rng1.Select
  MsgBox strProtokoll
' ***
  WordBasic.DisableAutoMacros 0
  Application.Visible = True
Exit Sub
' ******************************
Zeile_Fehler:
  MsgBox Err.Number & Selection.text
'  Resume ' Next
End Sub
' ----------------------------------------------------------------------------------------------------
```

```
Private Sub Einzelwort_Rechtschreibkorrektur()
' H. Chris Gast, 11. bis 13. Februar 2015, 23. Januar 2017, 8. Februar 2017.
' Rechtschreibung prüfen mit einigen Korrekturversuchen.
' Dabei korrigierte Wörter dunkelgelb, verbliebene Fehler rot markieren:
Dim strTestwort As String
strTestwort = Selection.text
Dim Rng2 As Range
Set Rng2 = Selection.Range
'  Bei Silbentrennung funktioniert in Word 2010 die Rechtschreibprüfung nicht:
  strTestwort = Replace(strTestwort, "-", "")
  strTestwort = Replace(strTestwort, ChrW(31), "")
'  ********************************
  If Application.CheckSpelling(Word:=strTestwort) = True Then
    ' Wenn Rechtschreibung bei Zusammenschreibung ok,
    ' dann ggf. Silbentrennungsstrich statt Bindestrich:
    Rng2.text = Replace(Rng2.text, "-" & ChrW(13), ChrW(31))
    Rng2.HighlightColorIndex = wdDarkYellow
  ElseIf Application.CheckSpelling(Word:=Replace(strTestwort, "m", "rn", 1, 1)) = True Then
    ' Bei eingescannten Texten (OCR)wird oft "rn" für "m" gehalten:
    Rng2.text = Replace(strTestwort, "m", "rn", 1, 1)
    Rng2.HighlightColorIndex = wdDarkYellow
  ElseIf Application.CheckSpelling(Word:=Replace(strTestwort, "rn", "m", 1, 1)) = True Then
    ' Bei eingescannten Texten (OCR) wird oft "m" für "rn" gehalten:
    Rng2.text = Replace(strTestwort, "rn", "m", 1, 1)
    Rng2.HighlightColorIndex = wdDarkYellow
  ElseIf Application.CheckSpelling(Word:=Replace(strTestwort, "1", "l")) = True Then
    ' "1", "l", "I" und "!" werden bei OCR manchmal verwechselt!
    Rng2.text = Replace(strTestwort, "1", "I")
    Rng2.HighlightColorIndex = wdDarkYellow
  ElseIf Application.CheckSpelling(Word:=Replace(strTestwort, "1", "l")) = True Then
    ' Ziffer "1" durch kleinen Buchstaben "l" ersetzen:
    Rng2.text = Replace(strTestwort, "1", "l")
    Rng2.HighlightColorIndex = wdDarkYellow
  ElseIf Application.CheckSpelling(Word:=Replace(strTestwort, "ß", "ss")) = True Then
    ' Bei alter Rechtschreibung "ß" durch "ss" ersetzen:
    Rng2.text = Replace(strTestwort, "ß", "ss")
    Rng2.HighlightColorIndex = wdDarkYellow
  ElseIf Selection.text = "usw" Then
    ' Bei "usw." gibt es Fehlermeldung, wenn der Punkt nicht mitmarkiert ist:
    Selection.MoveEnd Unit:=wdCharacter, Count:=1
    If Application.CheckSpelling(Word:=Selection.text) = False Then _
    Rng2.HighlightColorIndex = wdRed
  Else
    ' unkorrigierter Fehler, rot markieren:
    Rng2.HighlightColorIndex = wdRed
  End If
  Rng2.Select
End Sub
' ---------------------------------------------------------------------------------------------------------------


Private Sub CheckSpelling_Fremdsprachen()
' H. Chris Gast, 3. Februar 2017.
' https://www.experts-exchange.com/questions/20934350/VBA-Function-checkSpelling-finds-only-
' words-in-default-Language-but-not-in-others.html (3. Februar 2017)
' aber stark modifiziert.
Dim strText As String
 strText = Selection.text
Dim oDoc As Document
 Set oDoc = Application.Documents.Add(Visible:=False)
Dim lid As WdLanguageID
```

```
Dim CheckSpelling As Boolean
'  ******
'  Sprache des jeweiligen Wortes ermitteln und neues, unsichtbares Dokument anlegen,
'  dem als Ganzes diese Sprache zugewiesen wird:
   lid = Selection.LanguageID
   oDoc.Range.LanguageID = lid
   oDoc.Range.InsertAfter strText
'  ******
'  Rechtschreibfehler zählen (das funktioniert auch bei Fremdsprachen!).
   If oDoc.SpellingErrors.Count > 0 Then
      CheckSpelling = False
   Else
      CheckSpelling = True
   End If
'  ******
'  Unsichtbares Dokument schließen:
   oDoc.Close wdDoNotSaveChanges
   Set oDoc = Nothing
'  ******
'  Im zu untersuchenden Dokument im Fehlerfall das Wort rot markieren:
   If CheckSpelling = False Then
      Selection.Range.HighlightColorIndex = wdRed
   End If
 End Sub
'-----------------------------------------------------------------------------------------------------------
```

23 Zwei Word-Dateien auf dem Bildschirm nebeneinander anordnen

Das folgende Makro fand ich im Internet. Ich passte es für mein Word 2010 an.

Problem: Es müssen die Dateien in **derselben** Word-Instanz geöffnet sein (also nacheinander geöffnet, nicht gleichzeitig).

Es gab Probleme. Da tauchte plötzlich der Navigationsbereich auf und nahm Platz weg, oder der Platz für den Navigationsbereich wurde trotzdem ausgespart und von Mal zu Mal größer. Helfende Hinweise sind willkommen.

```
Sub Dokumente_nebeneinander_anordnen()
' H. Chris Gast 14. - 22. März 2019.
' aus http://www.chf-online.de/vba/wordfenster.htm, Word 2000,
' für Word 2010 modifiziert (Windows 7).
' Es können mit diesem Makro zwei oder drei Dateien nebeneinander auf dem Bildschirm
' angeordnet werden.
' Probleme:
' Die Dateien müssen alle in derselben Word-Instanz geöffnet sein.
' Beim wiederholten Anwenden wird links der Rand immer größer.
' *****************
Dim wnd As Word.Window
Dim appHeight As Integer
Dim appWidth As Integer
Dim lWindows As Integer
Dim appLeft As Integer
Dim appTop As Integer
Dim tt As Single
Dim IntGröße As Integer
'  ****************
  Application.ScreenUpdating = False
'  ****************
  'Fenster Info holen
  Set wnd = Application.ActiveWindow
  lWindows = Application.Windows.Count
  ' Prüfen, wieviel Fenster geöffnet sind
  If (lWindows = 0) Then
      '(Es ist keine Word-Datei geöffnet?)
      Exit Sub
  ElseIf (lWindows = 1) Then
      MsgBox "Es ist nur eine Datei geöffnet, " _
      & " oder die Dateien sind unterschiedlichen Word-Instanzen geöffnet!", vbCritical
      Exit Sub
  ElseIf lWindows = 2 Then
      ' zwei Dateien, 80 = optimiert für meinen Bildschirm:
      IntGröße = 80
  ElseIf lWindows = 3 Then
      ' bei drei Word-Dateien:
      IntGröße = 60
  Else
      MsgBox "Es sind " & lWindows & " Worddateien geöffnet, " _
      & "eine gleichzeitige Ansicht wäre nicht mehr lesbar." _
      & "Das Makro wird abgebrochen.", vbCritical
      Exit Sub
  End If
'  ****************
```

```
  'Bildschirmbreite, geteilt durch Anzahl Fenster
  appWidth = Application.UsableWidth / lWindows
  'Bildschirmhöhe
  appHeight = Application.UsableHeight
  'Position für Rand oben
  appTop = 0
  'Anfangsposition für linken Rand
  appLeft = 0
'  ****************
'  Eventuell noch vorhandene Ansicht mit 2 Fenstern beseitigen:
  Windows.BreakSideBySide
'  ****************
'  Schleife über alle Word-Fenster:
  For Each wnd In Word.Application.Windows
    'Fenstergröße "wiederherstellen"
    wnd.WindowState = wdWindowStateMaximize ' ergab Fehler bei "wnd.Width = appWidth!"
    ' nur mit dem folgenden Befehl funktioniert bei mir die Einstellung von "wnd.Widt":
    DoEvents
    wnd.WindowState = wdWindowStateNormal
    ActiveWindow.ActivePane.View.Zoom.Percentage = lntGröße
    ' ***********
    ' Fenstergröße für jedes Fenster "wnd" zuordnen:
    ' Fensterbreite
    wnd.Width = appWidth
    'Fensterhöhe
    wnd.Height = appHeight
    'Position links setzen
    wnd.Left = appLeft
    'Pos. oben ist bei Anordnung nebeneinander konstant:
    wnd.Top = 0   ' = appTop
    ' ***********
    ' Der folgende Befehl funktioniert unzuverlässig:
    ActiveWindow.ActivePane.View.Zoom.PageFit = wdPageFitBestFit
    If ActiveWindow.ActivePane.View.Zoom.Percentage > lntGröße + 5 Then
      ' Korrektur:
      ActiveWindow.ActivePane.View.Zoom.Percentage = lntGröße
    End If
    ' Etwas warten:
    Do While Timer < tt + 0.5  'Sekunden
      DoEvents
    Loop
    'linke Position für das nächste Fenster berechnen:
    '(letzte linke Position + Fensterbreite)
    appLeft = appLeft + appWidth
  Next wnd
  DoEvents
  Application.ScreenUpdating = True
End Sub
```

Neuer Versuch:

Mit dem Makro-Rekorder fand ich einen passenden Makro-Befehl:

```
…
  Windows.CompareSideBySideWith "Testdok1.doc [Kompatibilitätsmodus]"
…
```

Damit schrieb ich zwei neue Makro-Versionen, die beide zufriedenstellend funktionieren.

Die folgenden beiden Makros funktionieren bei mir gut (Word 2010)

```
Sub Zwei_Dokumente_nebeneinader_mittels_Aufruf_des_zweiten_Dokuments()
'  H. Chris Gast, 20. + 22. März 2019
'  Zwei Word-Dokumente nebeneinander auf dem Bildschirm anordnen,
'  die 2. Datei muss über den Menü-Dialog geöffnet werden.
'  Für Word 2010.
   Dim Dok1 As Word.Document
   Dim Dok2 As Word.Document
'  ****************
   Set Dok1 = ActiveDocument
   MsgBox "Bitte im folgenden Dialog die 2. Datei öffnen."
   With Application.Dialogs(wdDialogFileOpen)
      .Name = "*.doc*"
      ".Display = "Bitte 2. Datei öffnen."
      ".Show
      If .Show <> -1 Then
         MsgBox "Hallo, es ist nichts ausgewählt!"
         Exit Sub
      End If
   End With
   Set Dok2 = ActiveDocument
   If Dok1 = Dok2 Then
      MsgBox "Die Dateien sind identisch!", vbCritical
      Exit Sub
   End If
'  ****************
'  Eventuell noch vorhandene Ansicht mit 2 Fenstern beseitigen:
   Windows.BreakSideBySide
'  ****************
'  Windows-Befehl für nebeneinander aufrufen:
   Dok2.Activate
   Windows.CompareSideBySideWith Dok1.Name
'  Dokumente passend zoomen:
   ActiveWindow.ActivePane.View.Zoom.PageFit = wdPageFitBestFit
   Dok1.Activate
   ActiveWindow.ActivePane.View.Zoom.PageFit = wdPageFitBestFit
End Sub
' ---------------------------------------------------------------------------------------------------------------


Sub Zwei_offene_Worddokumente_nebeneinander_anordnen()
' H. Chris Gast, 20. bis 29. März 2019
' Zwei bereits geöffnete Dokumente nebeneinander auf dem Bildschirm anordnen.
' Es müssen genau zwei Word-Dateien in einer Instanz geöffnet sein.
' Die Dateien müssen in derselben Instanz geöffnet sein.
' Word 2010.
' Aus Platzgründen werden ggf. die Navigationsbereiche abgeschaltet.
' Funktionierte bei einer Kollegin trotz gleichem Computer überhaupt nicht.
' *****************
Dim Dok1 As Word.Document
Dim Dok2 As Word.Document
Dim IntWindows As Integer
'  ****************
   IntWindows = Application.Windows.Count
'  Prüfen, wieviel Word-Dateien geöffnet sind:
   If (IntWindows = 1) Then
      MsgBox "Es ist nur eine Datei geöffnet, " _
      & " oder die Dateien sind unterschiedlichen Word-Instanzen geöffnet!", vbCritical
      Exit Sub
   ElseIf IntWindows = 2 Then
```

```
    ' Ok, es sind 2 Word-Dateien geöffnet.
  Else
    MsgBox "Es sind " & IntWindows & " Worddateien geöffnet, " _
    & "es sind bei diesem Makro nur 2 geöffnete Word-Dateien zulässig." _
    & "Das Makro wird abgebrochen.", vbCritical
    Exit Sub
  End If
'  ****************
'  Ggf. alte bereits vorhandene Ansicht mit 2 Fenstern beseitigen:
  Windows.BreakSideBySide
'  ****************
'  Die beiden Dateien intern benennen:
  Set Dok1 = ActiveDocument
  ' Aus Platzmangel ggf. Navigationsfenster schließen:
   CommandBars("Navigation").Visible = False
   ActiveWindow.DocumentMap = False
  On Error Resume Next
  ActiveWindow.Next.Activate
  If Err.Number > 0 Then
    ActiveWindow.Previous.Activate
    Err.Clear
  End If
  Set Dok2 = ActiveDocument
   ' Aus Platzmangel auch hier Navigationsfenster schließen:
   CommandBars("Navigation").Visible = False
   ActiveWindow.DocumentMap = False
'  ****************
  '' Windows(Dok2.Name & " [Kompatibilitätsmodus]").Activate
  Dok2.Activate
  ' Nebeneinander anordnen:
  Windows.CompareSideBySideWith Dok1.Name
  ' Größe anpassen:
  Dok2.Activate
  ActiveWindow.ActivePane.View.Zoom.PageFit = wdPageFitBestFit
  If ActiveWindow.ActivePane.View.Zoom.Percentage > 80 Then
    ' notfalls Korrektur:
    ActiveWindow.ActivePane.View.Zoom.Percentage = 75
  End If
  ' Größe des ersten Dokuments anpassen:
  Dok1.Activate
  ActiveWindow.ActivePane.View.Zoom.PageFit = wdPageFitBestFit
  If ActiveWindow.ActivePane.View.Zoom.Percentage > 80 Then
    ' notfalls Korrektur:
    ActiveWindow.ActivePane.View.Zoom.Percentage = 75
  End If
End Sub
'-----------------------------------------------------------------
```

24 Anwendung von SendKeys-Befehlen für ein Printscreen-Makro

Per Makro kann man die Printscreen-Taste simulieren, und den Inhalt der Zwischenablage wahlweise in Photoeditor, Paint oder irgendein anderes Bildbearbeitungsprogramm einfügen. Allerdings funktionieren diese Makros noch nicht optimal (für Verbesserungsvorschläge wäre ich dankbar).

24.1 Printscreen mit Photoeditor (mittels Word-Makro, VBA)

Für die Printscreen-Funktion gibt es nur 2 Möglichkeiten, sie per Makro auszuführen:

a) per SendKeys-Befehl mit dem alten WordBasic;
b) per API-Funktion.

Will man nicht nur Word-Dateien mittels Printscreen fotografieren, muss Word dafür extra gestartet werden. Hierfür entwickelte ich ein Makro (VBA) in einer separaten Word-Datei, welches mittels der Prozedur "Sub AutoOpen()" die "PrintScreen"-Funktion ausführt, dann das Programm "Microsoft-Paint" (mspaint.exe) bzw. das Programm "Photoeditor" aufruft und dort die Zwischenablage einfügt. Gestartet wird das Makro über einen Link der Word-Datei zum Desktop, wo ich diesem Link ein Tastenkürzel zuordnen kann14

Gelöstes Problem: Befehle an fremde Programme wie Paint bzw. Photoeditor lassen sich nur mit "**SendKeys**" schicken. Eigentlich sind SendKeys-Befehle zu vermeiden, weil sie unter anderem die NumLock-Taste deaktivieren können. Wenn man sie also benutzt, dann nur mit Vorsicht! Und man muss oft hinterher die NumLock-Taste wieder einschalten. Ferner muss eine Pause von 1 Sekunde eingefügt werden, wenn ein Makro, dass SendKeys-Befehle enthält, über ein Tastenkürzel gestartet wird. **Auch zwischen den SendKeys-Befehlen sind Pausen einzufügen.** Diese hängen vom jeweiligen Computer ab. Anfangs gelang es mir zum Beispiel nicht, mittels des Makros mit SendKeys-Befehlen den Desktop zu fotografieren. Erst als ich die Warteschleifen länger machte, funktionierte dies.15 Ein Screenshot durch Anklicken der Datei direkt im Windows-Explorer braucht offenbar weniger Zeit, als der Start über einen Link vom Desktop und über ein Tastenkürzel zum Link auf dem Desktop.

Weiteres gelöstes Problem: In Windows XP wird (fehlerhaft) durch den SendKeys-Befehl nur das "Aktive Fenster" fotografiert, auch wenn ich keine Alt-Taste im Befehl enthalten habe.

```
WordBasic.SendKeys "{druck}"   ' (nur für deutsche Version)
```

Den Durchbruch erreichte ich hier durch die Verwendung der Variante mit API-Funktion. In Windows 7 wird im folgenden Makro mit einer API-Funktion auch die NumLock-Taste wieder eingeschaltet, wenn sie durch die SendKeys-Befehle (fehlerhaft) deaktiviert worden ist.

Installation:
In einer neu erzeugten Word-Datei wird im Makro-Explorer (aufrufen mit Alt + F11) im Makromodul "ThisDocument" das folgende Makro untergebracht (siehe Bild auf der nächsten Seite).

Achtung: Das Makro schließt am Ende die Word-Datei. Will man die fertige Datei mit dem Makro einsehen, ohne dass das Makro startet und die Datei sofort schließt, öffne man die Datei mit gedrückter "Shift-Taste", wodurch alle Auto-Makros unterdrückt werden.

---

14 Zuordnung von Tastenkürzeln siehe www.siebener-kurier.de/chris-aufsaetze/Word-Tastaturbefehle.pdf, dort den Abschnitt über Windows-Explorer.

15 Im Zweifel zuerst alle Warteschleifen recht lang machen und dann verkürzen, bis das Makro nicht mehr funktioniert. Die Wartezeiten für SendKeys sind bei jedem Computer anders.

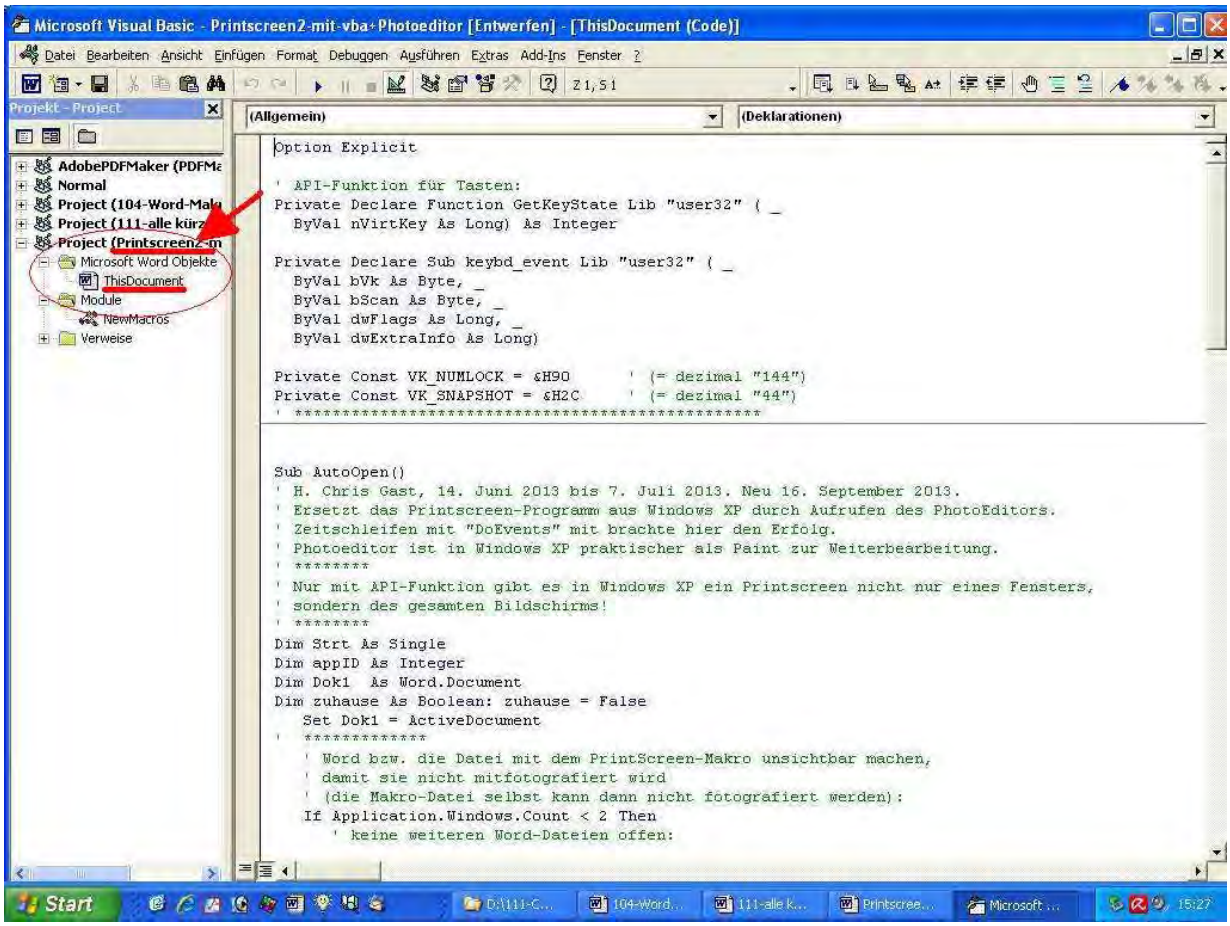


Bild 24.1:Screenshot des Makromoduls "ThisDocument" mit der Prozedur "Sub AutoOpen()"

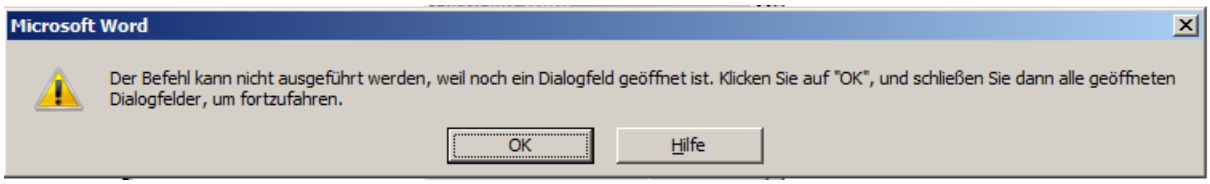
Das fertige Makro im Makro-Modul "ThisDocument" einer Word-Datei

```
Option Explicit

' API-Funktion für Tasten:
Private Declare Function GetKeyState Lib "user32" ( _
  ByVal nVirtKey As Long) As Integer

Private Declare Sub keybd_event Lib "user32" ( _
  ByVal bVk As Byte, _
  ByVal bScan As Byte, _
  ByVal dwFlags As Long, _
  ByVal dwExtraInfo As Long)
' ******************************************************


Sub AutoOpen()
' H. Chris Gast, Stand 16. September 2013.
' Printscreen-Programm mit Aufrufen des PhotoEditors.
' Zeitschleifen mit "DoEvents" mit brachte hier den Erfolg.
' Photoeditor ist in Windows XP praktischer als Paint zur Weiterbearbeitung.
' In Windows 7 dagegen wäre Paint praktischer.
' ********
Dim Strt As Single
Dim appID As Integer
Dim Dok1  As Word.Document
   Set Dok1 = ActiveDocument
'  *************
```

```
  ' Word bzw. die Datei mit dem PrintScreen-Makro unsichtbar machen,
  ' damit sie nicht mitfotografiert wird:
  ' (die Makro-Datei selbst kann dann nicht fotografiert werden):
  If Application.Windows.Count < 2 Then
    ' keine weiteren Word-Dateien offen:
    Application.Visible = False
    ' Zeitschleife 1 Sekunden, damit sich Word in Ruhe öffnet:
    Strt = Timer
    Do While Timer < Strt + 1#
      DoEvents
    Loop
  Else
    ' andere Word-Dateien offen (nur Makro-Datei unsichtbar machen):
    '' ActiveDocument.Visible = False ' geht nicht, gibt Fehlermeldung!
    '' Application.WindowState = wdMinimized  ' (ging nicht so gut)
    Application.ActiveWindow.Visible = False
    Strt = Timer
    Do While Timer < Strt + 1#
      DoEvents
    Loop
  End If
'  ****************
  ' Printscreen-Funktion per API
   keybd_event 44, 0, 0, 0
      ' PrintScreenTaste per VBA drücken:
      '' SendKeys "{PRTSC}"       ' Funktionierte nicht!
      '' WordBasic.SendKeys "{druck}"   ' (nur für deutsche Version, erfasst nur das aktuelle Fenster)
   Strt = Timer
   Do While Timer < Strt + 0.1
      DoEvents
   Loop
'  ****************************************
'  PhotoEditor aufrufen (mit Pfadangabe)
'  (Die Pfadangabe ist vom Computer abhängig. Bei Paint kann der Pfad entfallen),
  appID = Shell("C:\Programme\Gemeinsame Dateien\Microsoft Shared\PhotoEd\PHOTOED.exe", _
              vbNormalFocus)
  ' Zeitschleife:
  Strt = Timer
  Do While Timer < Strt + 1#  ' 0.5 ' (in Sekunden)
    DoEvents
  Loop
'  ********
'  Sicherstellen, dass hier wirklich "Photoeditor" den Fokus hat:
  AppActivate appID
'  ********
  Strt = Timer
  Do While Timer < Strt + 0.7
    DoEvents
  Loop
'  *******************
  ' SendKeys aufrufen:
    '' SendKeys "^v"  ' geht nicht bei PhotoEditor (aus der Zwischenablage einfügen = Strg + v)
   SendKeys "(%b)"
    Strt = Timer
   Do While Timer < Strt + 0.1
     DoEvents
   Loop
   SendKeys "n"
'  *******************
'  Falls SendKeys-Befehle die NumLock-Taste deaktiviert haben, sie wieder einschalten:
' NUM-Lock aktivieren (falls deaktiviert):
  '  vbKeyNumlock = 144
```

```
  If GetKeyState(vbKeyNumlock) <> 1 Then
    keybd_event vbKeyNumlock, 1, 0, 0
    keybd_event vbKeyNumlock, 1, 2, 0
  End If
'  ***************
'  Word, bzw. nur diese Datei wieder schließen:
  If Application.Windows.Count < 2 Then
     'Application.Visible = True
     Application.Quit SaveChanges:=False
  Else
    "Application.ActiveWindow.Visible = True
    Dok1.Close SaveChanges:=False
  End If
End Sub
```

Problem für Printscreen einer MsgBox: Printscreen mittels VBA-Makro funktioniert nur, solange nicht in Word eine MsgBox oder InputBox oder Userform offen ist. Beim Versuch des Startes gibt es dann eine Fehlermeldung. Denn solange nicht "OK" geklickt ist, lässt sich keinerlei Word-Makro starten oder neues Word-Dokument öffnen (siehe Bild):



"Der Befehl kann nicht ausgeführt werden, weil noch ein Dialogfeld geöffnet ist. Klicken Sie auf "OK", und schließen Sie dann alle geöffneten Dialogfelder, um fortzufahren."

Bild 24.2: Screenshots einer Fehlermeldung

Lösung: Durch den Aufruf über ein VBScript (VBS) wird Word in einer **neuen Instanz** geöffnet. Damit kann man auch eine Msgbox oder eine UserForm fotografieren.

VB-Scripts schreibt man am besten mit dem Editor (notfalls mit Word), speichert es als Textdatei und ändert die Datei-Endung ".txt" in ".vbs".

Ich verknüpfte das VB-Script mit dem Desktop und wählte dort ein Tastenkürzel mit AltGr .

Aufruf der Worddatei mit dem VBA-Makro in einer neuen Instanz mittels VB-Script (VBS):

Printscreen2_mit_Photoeditor_über_Wordmakro-Aufruf.vbs

```
'<Printscreen_mit_Paint_über_Wordmakro-Aufruf.vbs>
' H. Chris Gast, 22. September 2013.
' Dieses VB-Script ruft eine Word-Datei mit Makro in separater Instanz auf,
' damit auch eine MsgBox aus Word fotografiert werden kann.
'
Set oShell = CreateObject("WScript.Shell")
Set oword = CreateObject ("Word.Application")
'
'  Der folgende Pfad erwies sich bei mir als nötig. Er ist in jedem Computer anders.
Set Textobjekt = oword.Documents.Open _
      (""""D:\111-Chris\113-Wordzubehör\Makros für Word\VBS-Makros\Printscreen2-mit-
          vba+Photoeditor.doc""")
' **************
```

```
' Word wieder verlassen und aufräumen:
On Error Resume Next
oword.Quit   ' wenn das VBA-Makro abgestürzt war, bleibt sonst eine offene Word-Instanz.
Set oword = Nothing
Set oShell = Nothing
```

Auswertung:

Dieses Makro kann bei Start über VBS auch eine MsgBox von Word erfassen, siehe Bild:

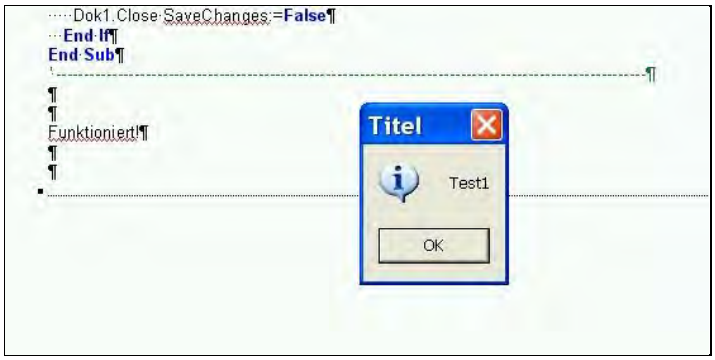


Bild 24.3: Screenshot einer MsgBox mitsamt Hintergrund in Word 2000 und Windows XP mittels VBS und VBA mit API

Offenes Probleme:

Das Makro stürzt ab, wenn der Photoeditor bereits geöffnet ist. Dieses Problem entfällt, wenn man statt dem "Photoeditor" das Programm "Paint" verwendet.

In Windows 7 wird die Kopfzeile einer fotografierten MsgBox farblos, sobald der Fokus nicht mehr darauf liegt, etwa weil das Makro gestartet ist.

24.2 Printscreen mit Paint, ganz ohne Word, nur mittels VBS

Am einfachsten ist natürlich jetzt, auf Word und VBA ganz zu verzichten. In VBS kann man aber keine API-Funktionen benutzen. Und in Windows XP wird bei PrintScreen mittels SendKeys-Befehl leider stets nur der aktuelle Bildschirm erfasst, also von einer Msgbox nicht der Hintergrund dargestellt (Fehler nur in meiner Version von Windows XP?). Deshalb ist die folgende Lösung bei mir nur für Windows 7 geeignet, wenn ich mehrere Fenster gleichzeitig erfassen will. Und in Windows 7 ist Paint günstiger als der Photoeditor.

vbs-Printscreen_nur_mit_paint.vbs

```
' Printscreen nur mit Paint, 2. September 2013.
' Funktioniert mit Ergänzung für NumLock-Taste auch in Windows 7.
Dim id, opaint
' ***********
  set oShell = WScript.CreateObject("WScript.Shell")
  '
  with CreateObject("Word.Basic")
     .SendKeys "{1068}"      ' (sprachunabhängige Version)
  end with
  '
  WScript.Sleep 1000 '500 ' 1500
' ************************************
'
' Paint öffnen und Fenster aktivieren:
 Set oPaint = oShell.Exec("mspaint")
 id = oPaint.ProcessID
```

```
 WScript.Sleep 1000
'
 oShell.AppActivate(id)
 WScript.Sleep 500  '200
'
'' WScript.Quit
 '
 oShell.sendkeys "%{ }w"
 WScript.Sleep 200
' ***********************************
if not oshell.AppActivate(id) Then
   MsgBox "Konnte Fenster nicht aktivieren! " & id
   WScript.Quit
Else
  ' Msgbox "Fenster " & id & " ist aktiv."
End If
' **********************************
' Fenster maximieren:
WScript.Sleep 500 ' 1500
 oShell.SendKeys "% x"
' *********************************
' Zwischenablage einfügen mit Strg + V:
 WScript.Sleep 100 '500
 oShell.sendkeys"^(v)"
'
' Damit die Auswahl gleich neu gewählt werden kann:
 WScript.Sleep 100 '500
 oShell.SendKeys "{ESC}"
'*********************************************
' in Windows 7 folgendes ergänzen:
' NumLock-Taste wieder einschalten:
 WScript.Sleep 100
 oShell.Sendkeys "{NumLock}"
' *********************************************
```

Vorteil dieser Makro-Version mit Paint gegenüber der Version mit Photoeditor:
Kann auch Screenshots bei bereits geöffnetem Paint-Programm machen.

Nachteil der Version ohne Word:
Erfasst wegen des SendKeys-Befehl in Windows XP nur das aktuelle Fenster.

Offenes Problem bei Windows 7:
In Windows 7 wird die Kopfzeile einer fotografierten MsgBox farblos, sobald der Fokus nicht mehr darauf liegt, etwa weil das Makro gestartet ist.

24.3 Weitere Tipps zu VB-Script (VBS)

Man kann auch einzelne VBS-Befehle in VBA einfügen (jedoch nicht umgekehrt VBA-Befehle in VBS einfügen):

a) Befehl für MsgBox-ähnliches "Popup" mit beschränkter Anzeigezeit:

```
oShell.Popup "Nachricht", 1 , "Titel" , vbinformation
```

Die Zahl "1" ist hierbei die Zeit in Sekunden.

```
Sub Popup_für_VBA()
'  Die Dauer ist als Ganze Zahl in Sekunden einzugeben:
   Dim oShell As Object
   Set oShell = CreateObject("WScript.Shell")
   oShell.Popup "Nachricht", 1, "Titel", vbInformation
   Set oShell = Nothing
End Sub
```

b) Bildschirmtastatur starten:

In Windows 7 funktionierte der VBA-Befehl mit "Shell" für die Bildschirmtastatur nicht mehr.

~~Shell("C:\Windows\system32\osk.exe", vbNormalFocus)~~

~~Abhilfe: Mittels VBS-Befehl:~~

```
Sub Bildschirmtastatur_für_Windows_7()
' aus: http://www.office-loesung.de/ftopic390970_0_0_asc.php
   Dim sh As Object
   Set sh = CreateObject("WScript.Shell")
   sh.Run Environ("windir") & "\System32\osk.exe"
   Set sh = Nothing
End Sub
```

~~Beide Beispiele funktionieren unter Windows XP und unter Windows XP.~~

Unter Windows 10 funktioniert bei mir keine dieser beiden Möglichkeiten mehr.
Neue Lösung (November 2020) siehe Abschnitt 25.

25 Bildschirm-Tastatur per VBA ein- und auschalten

Für das Öffnen der Bildschirmtastatur gibt es das Tastenkürzel Strg + Win + o. Man kann dies aber auch per VBA-Makro erledigen. Die Bildschirmtastatur brauche ich für die Eingabe von griechischen oder russischen Buchstaben. Für das Umstellen der Tastatur usw. muss natürlich erst die Spracheinstellungen in Windows installiert werden.

Unter Windows 10 funktionierte bei mir das bisherige VBA-Makro für Bildschirmtastaturen nicht mehr. Im Internet fand ich nach **längerer** Suche schließlich eine **funktionierende** Lösung. Dafür sind im Makro-Modul vor dem allerersten Makro zwei Funktionen zu deklarieren.

```
Option Explicit
'
' Für Bildschirmtastatur in Windows 10 mit 64 Bit,  Download 2. Dezember 2020:
'  aus: https://stackoverflow.com/questions/49826995/calling-on-screen-keyboard-from-excel-vba
Private Declare Function ShellExecute Lib "shell32.dll" Alias "ShellExecuteA" _
        (ByVal hwnd As Long, ByVal lpOperation As String, _
       ByVal lpFile As String, ByVal lpParameters As String, ByVal lpDirectory As String, _
       ByVal nShowCmd As Long) As Long


Private Declare Function Wow64EnableWow64FsRedirection Lib "kernel32.dll" _
         (ByVal Enable As Boolean) As Boolean
'  ***********************************************************************


Sub Bildschirmtastatur_starten()
'   Download 2. Dezember 2020.
'   Für Windows 10 und 64 Bit wird eine neue Methode für das Makro benötigt.
'   Das folgende VBA-Makro funktioniert bei mir:
'   https://stackoverflow.com/questions/49826995/calling-on-screen-keyboard-from-excel-vba
   "Sub RunOsk_on64Bit()
Const SW_SHOWNORMAL = 1
   On Error Resume Next
   Wow64EnableWow64FsRedirection False
   ShellExecute 0, "open", "osk.exe", "", "C:\windows\system32\osk.exe", SW_SHOWNORMAL
   Wow64EnableWow64FsRedirection True
End Sub
' ---------------------------------------------------------------------------------------------


Sub Bildschirmtastatur_schließen()
' H. Chris Gast, Stand 12. Januar 2016.
' Funktioniert in Word 2000 bis Word 2010, und in Windows XP bis  Windows 10.
'  *************
' Bildschirmtastatur per Makro schließen:
' aus: http://www.office-loesung.de/ftopic146962_0_0_asc.php, verfasst: 21. Mai 2007.
Dim objWMIService
Dim colProcessList
Dim objProcess
  Set objWMIService = _
         GetObject("winmgmts:{impersonationLevel=impersonate}!\\.\root\cimv2")
  Set colProcessList = objWMIService.ExecQuery _
                  ("Select * from Win32_Process Where Name = 'osk.exe'")
  For Each objProcess In colProcessList
    objProcess.Terminate
  Next
End Sub
' ---------------------------------------------------------------------------------------------
```

26 "Bibliotheken" (Verweise/References) des VBA-Editors auflisten

Manchmal funktionieren Makro-Befehle in einem anderen Computer nicht, obwohl sie dort in der betreffenden Word-Version vorhanden sein sollten. Meistens ist einfach die Sicherheit zu hoch eingestellt.
Aber es kann auch vorkommen, dass ein einfacher Makrobefehl wie MsgBox "Hallo Welt" funktioniert, aber nicht die gewünschten Makrozeilen, und es eine Fehlermeldung gibt wie im folgenden Bild:

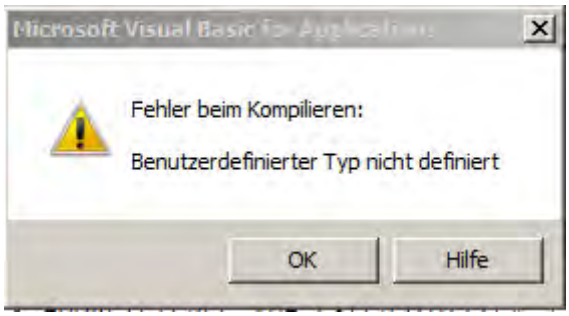
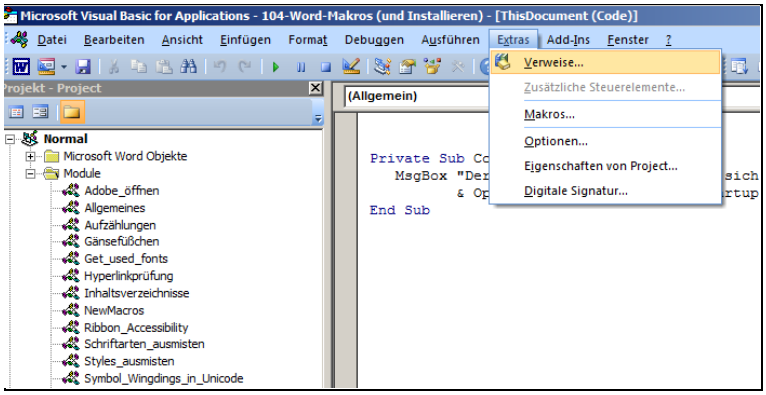


Bild 26.1: Fehlermeldung bei fehlender Bibliotheks-Verknüpfung

Ursache kann sein, dass im anderen Computer einige "Bibliotheken" nicht eingebunden sind. Abhilfe: Mit Alt + F11 in den Makroeditor gehen, dann "Extras"↱→↱"Verweise . . .", und nun nachsehen, bei welchen "Verfügbaren Verweisen" ("Bibliotheken") Häkchen gesetzt sind.



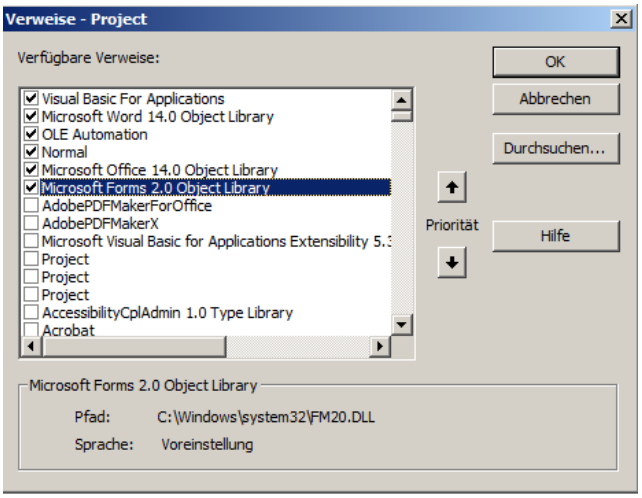


Bild 26.2: Bibliotheks-Verknüpfung ("Verweise. . .") nachprüfen

Dies kann man sich per Screenshot speichern, oder (umständlicher) per Makro ausgeben:

```
Sub Bibliotheken_Verweise_auflisten()
' Verschiedene Quellen, kombiniert, erweitert und modifiziert.
Dim Verweis  As Object      ' Dim Verweis As Reference
   ' ("Reference" setzt Verweis auf "Microsoft VBA-Extensibility 5.3" voraus).
Dim i As Integer: i = 1
' ****
On Error Resume Next
  For Each Verweis In Application.VBE.ActiveVBProject.References
    Debug.Print i & vbLf _
        & "Bezeichnung: " & Verweis.Description & vbLf _
        & "Speicherort: " & Verweis.FullPath & vbLf _
        & "Referenznummer (GUID): " & Verweis.GUID & ", " _
        & "Nr-Major: " & Verweis.Major & ",   Nr-Minor: " & Verweis.Minor & vbLf _
        & "Name: " & Verweis.Name & vbLf _
        & vbLf
    i = i + 1
  Next Verweis
End Sub
' ---------------------------------------------------------------------------------------------------------
```

Dies ergibt zum Beispiel:

```
1
Bezeichnung: Visual Basic For Applications
Speicherort: C:\PROGRA~1\COMMON~1\MICROS~1\VBA\VBA7\VBE7.DLL
Referenznummer (GUID): {000204EF-0000-0000-C000-000000000046}, Nr-Major: 4,   Nr-Minor: 1
Name: VBA
2
Bezeichnung: Microsoft Word 14.0 Object Library
Speicherort: C:\Program Files\Microsoft Office\Office14\MSWORD.OLB
Referenznummer (GUID): {00020905-0000-0000-C000-000000000046}, Nr-Major: 8,   Nr-Minor: 5
Name: Word
3
Bezeichnung: OLE Automation
Speicherort: C:\Windows\system32\stdole2.tlb
Referenznummer (GUID): {00020430-0000-0000-C000-000000000046}, Nr-Major: 2,   Nr-Minor: 0
Name: stdole
4
Bezeichnung: Microsoft Office 14.0 Object Library
Speicherort: C:\Program Files\Common Files\Microsoft Shared\OFFICE14\MSO.DLL
Referenznummer (GUID): {2DF8D04C-5BFA-101B-BDE5-00AA0044DE52}, Nr-Major: 2,   Nr-Minor: 5
Name: Office
5
Bezeichnung:
Speicherort: C:\Users\gst\111-transfer\Normal
Referenznummer (GUID): , Nr-Major: 0,   Nr-Minor: 0
Name: Normal
```

Man kann die fehlenden Bibliotheken auch per Makro einfügen:

```
  Application.VBE.ActiveVBProject.References.AddFromFile _
      "C:\Program Files\Common Files\Microsoft Shared\VBA\VBA6\VBE6EXT.OLB"
```

oder:

```
  Application.VBE.ActiveVBProject.References.AddFromGuid _
      "{0002E157-0000-0000-C000-000000000046}", 1, 0
```

Am einfachsten fügt man sie von Hand ein, da dies nur einmal pro Computer nötig ist.

Weitere Literatur

[1] Hanna-Chris Gast: "**Tastenkombinationen (Tastenkürzel) für Word, Windows, Internet-Explorer und Adobe-Pdf**"; Berlin, Stand 2019, als Download unter: http://www.siebener-kurier.de/chris-aufsaetze/Word-Tastaturbefehle.pdf

[2] Hanna-Chris Gast: "**Schrift- und Sonderzeichen in Word**", Broschüre im Selbstverlag, Berlin, Stand 2019; Download unter: http://www.siebener-kurier.de/chris-aufsaetze/Word-Schriftzeichen.pdf.

[3] Hanna-Chris Gast: **"Einige Probleme mit Word und deren Lösung";** Stand 2018, Download: http://www.siebener-kurier.de/chris-aufsaetze/Word-Probleme-Anleitungen.pdf.

[4] Hanna-Chris Gast: "**Formblatt mit Makros zur Übertragung von Tastenkürzel und AutoKorrekturen für Word**"; Berlin 2011, Stand 2015; Als Download: http://www.siebener-kurier.de/chris-aufsaetze/Formblatt-Tastenkuerzel+Autokorrekturen_uebertragen.doc

[5] Hanna-Chris Gast: "**Word-Makros für Unicode**, Analyse von Schrift- und Sonderzeichen, Ersatz von Symbol- und Wingdingszeichen durch Unicode sowie ein verbessertes Inhalt-Einfügen"; Selbstverlag; Berlin 2010/2011, Neu-Auflage, 2015, Stand 2018; http://www.siebener-kurier.de/chris-aufsaetze/Unicode_Word-Makros.pdf;
Die Makros aus diesem Aufsatz sind in einer fertigen Dokumentvorlage downloadbar:
http://www.siebener-kurier.de/chris-aufsaetze/Unicode-Makros.zip

[6] Hanna-Chris Gast: "Ergonomisch eingerichteter Computer-Arbeitsplatz", Stand 2017, www.siebener-kurier.de/chris-aufsaetze/Ergonomie_fuer_Bildschirmarbeitsplatz.pdf

[7] Toni Jollans, http://www.wordarticles.com/Articles/Colours/2007.php Abruf März 2017.

[8] Wikipedia, u. a. https://de.wikipedia.org/wiki/CMYK-Farbmodell,
https://en.wikipedia.org/wiki/CMYK_color_model;
https://fr.wikipedia.org/wiki/Quadrichromie;
https://ru.wikipedia.org/wiki/Обсуждение:CMYK (Abruf zuletzt März 2017).

Impressum

Hanna-Chris Gast: "**Einige Makros für Word (und deren Installation)**";
Download: http://www.siebener-kurier.de/chris-aufsaetze/Word-Makros.docx
Selbstverlag Berlin 2013, Stand: 25. April 2023

Dipl.-Ing. Hanna- Chris Gast
Bergstr. 27
12169 Berlin

Im Internet: www.siebener-kurier.de/chris
bzw. www.siebener-kurier.de/chris-aufsaetze
E-Mail: hcgast "ät" siebener-kurier.de *(dabei "ät" durch "@" ersetzen).*