# The Macros

**Sub CommonCode()**

' Paul Beverley - Version 16.11.22

' Not a macro - just useful computer code

' Stop the screen update

myScreenOff = True

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

Application.ScreenUpdating = False

On Error GoTo ReportIt

Application.ScreenUpdating = True

Exit Sub

' Switch the screen back on if there's an error

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

' Create a range, rounding off the selection to whole words

' - or select the whole paragraph, if nothing was selected

If Selection.Start = Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

Else

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

rng.Select

End If

' Create a new document, copying the old

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

' Or for pure text, use:

' rng.Text = rngOld.Text

' Set up a range numerically

Set rng = ActiveDocument.Range(Start:=0, End:=10)

' Bog-standard F&R using a range

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13{2,}"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Turn off "smart" (ha!) cut and paste

mySmart = Options.SmartCutPaste

Options.SmartCutPaste = False

' turn it back on if the user wants it

Options.SmartCutPaste = mySmart

' Dummy Select Case command

Select Case nowChar

Case 13: Selection.TypeText "-"

Case 32: Selection.TypeText "elephant"

Case Else: Selection.TypeText myDash1

End Select

' Standard response from user

myResponse = MsgBox("Scan the whole document?!", \_

vbQuestion + vbYesNo, "SerialCommaAlyse")

If myResponse <> vbYes Then Exit Sub

' Input a number from the user

Do

myText = InputBox("1: Copy this macro" & vbCr & \_

"2: Restore keystroke", "MacroUpdater")

myNumber = Val(myText)

If myNumber = 0 Then Beep: Exit Sub

Loop Until myNumber = 1 Or myNumber = 2

' Set up a range to the end of the document

Set rng = ActiveDocument.Range(Selection.End, ActiveDocument.Content.End)

'Extend selection to nearest para

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

Selection.End = rng.End

' Prepare to use speech

Set speech = New SpVoice

' When you need to say something, use

speech.Speak myWords, SVSFPurgeBeforeSpeak

' Set whether to track or not

doTrack = True

' Remember whether tracking was on

myTrack = ActiveDocument.TrackRevisions

' Switch tracking off if it's not wanted

If doTrack = False Then ActiveDocument.TrackRevisions = False

' Put tracking back on, if it was on before

ActiveDocument.TrackRevisions = myTrack

' Reduce ange/selection to beginning/end

rng.Collapse wdCollapseEnd

Selection.Collapse wdCollapseStart

Selection.Expand wdFieldLink

Selection.Expand wdParagraph

Selection.Expand wdTable

Selection.Expand wdSentence

Selection.Expand wdWord

' Other useful commands

Documents.Open fileName:=myFullName

Selection.Range.Case = wdLowerCase

' Set up for neat, easy newlines

CR = vbCr

CR2 = CR & CR

' then

Selection.TypeText Text:=myParaText & CR2

' How long did it take?

myT = Timer

' Do whatever

myT2 = Timer

Debug.Print myT2 - myT

'Select just the word

Selection.MoveEndWhile cset:=ChrW(8217) & " '", count:=wdBackward

Loop

' Current page number?

pageNum = rng.Information(wdActiveEndAdjustedPageNumber)

inAtable = rng.Information(wdWithInTable)

' Delete the currently active document and don't ask the user

' Be careful what you wish for!

ActiveDocument.Close SaveChanges:=False

' Create a double beep

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

' Run another macro from this one

Application.Run macroName:="AcceptchangesSelected"

' or just

Call AcceptchangesSelected

' Undo the previous editing action

WordBasic.EditUndo

' Choose the colour that F&R searches, then restore the colour

myColour = wdYellow

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

' Do F&R

Options.DefaultHighlightColorIndex = oldColour

' Let the user choose various words/phrases

wds = "cows,pigs,fish"

' Now tidy it up

' Force comma at start; no comma at end

wds = "," & wds & ","

wds = Replace(wds, ",,", ",")

wds = Left(wds, Len(wds) - 1)

myWord = Split(wds, ",")

For i = 1 To UBound(myWord)

Blah = myWord(i)

Next i

' What's the normal font in this file?

nmlFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

' Do things in each para

For Each myPar In ActiveDocument.Paragraphs

myText = myPar.Range.Text

Next myPar

' current para number

paraNum = ActiveDocument.Range(0, Selection.Paragraphs(1).Range.End).Paragraphs.Count

' OR (more understandable)

Set rng = ActiveDocument.Range(0, Selection.End)

paraNum = rng.Paragraphs.Count

' Note how some entities (like .Paragraphs) need to

' be followed by .Range, while others don’t!?

For Each myTable In ActiveDocument.Tables

For Each myCell In myTable.Range.Cells

myText = myCell.Range

...

Next myCell

Next myTable

For Each rv In sourceText.Revisions

Set rng = rv.Range.Duplicate

...

Next rv

For Each wd In rng.Words

If wd.Font.Shadow = True Then wd.Case = wdUpperCase

Next

For Each ch In rng.Characters

If ch.Text = "Blah" Then

ch.Select

...

End If

Next ch

For Each sn In ActiveDocument.Sentences

sn.Select

If Len(sn) > 5 Then

...

End If

Next sn

' SAVE OLD VALUES

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

' Work on all tables or only selected ones.

If Selection.Start = Selection.End Then

Set workRange = ActiveDocument.Tables

doTheLot = True

Else

Set workRange = Selection.Tables

End If

With workRange.Find

' etc

.Text = oldFind

.Forward = True

.Replacement.Text = oldReplace

.Wrap = wdFindContinue

End With

' Get rid of weird colouring

Selection.Shading.Texture = wdTextureNone

Selection.Shading.ForegroundPatternColor = wdColorAutomatic

Selection.Shading.BackgroundPatternColor = wdColorAutomatic

' Selection.Information is a powerful command

hPosition = Selection.Information(wdHorizontalPositionRelativeToPage)

Dim ss(2) As String

ss(0) = "orange"

ss(1) = "apple"

ss(2) = "banana"

WordBasic.SortArray ss()

' To sort up or down

If sortReversed Then

WordBasic.SortArray ss(), 1

Else

WordBasic.SortArray ss(), 0

End If

' Check all characters in a text

For i = 1 To Len(myText)

thisChar = Asc(Mid(myText, i, 1))

Next

' Check through all the open documents

For Each myDoc In Documents

myName = myDoc.Name

allFileNames = allFileNames & "|" & myName

If InStr(allFileNames, myName) = 0 Then

Else

' This file has the same name as anothe open file

' Maybe check full address and name of file

myFullName = myDoc.FullName

End If

Next myDoc

On Error GoTo ReportIt

' Error reporting is complicated

On Error GoTo ReportIt

' At end of macro

Exit Sub

ReportIt:

If Err.Number = 5097 Then

DoEvents

' Do something

Resume Next

Else

On Error GoTo 0

DoEvents

Resume

End If

'End Sub

' Or use

' On Error Resume Next

' End Sub

End Sub

**Sub MacroFetch()**

' Paul Beverley - Version 21.08.21

' Fetches the macro named in the clipboard or at the cursor

Selection.Collapse wdCollapseEnd

Selection.Expand wdWord

If Len(Selection) < 3 Then

Selection.MoveLeft , 3

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

myMacroName = Selection

thisIsMacro = True

ch1 = Left(myMacroName, 1)

If LCase(ch1) = ch1 Then thisIsMacro = False

For i = 1 To Len(myMacroName) - 3

ch = Mid(myMacroName, i, 1)

If LCase(ch) = UCase(ch) Then

thisIsMacro = False

Exit For

End If

DoEvents

Next i

GotMacro = True

ch1 = Left(myMacroName, 1)

If LCase(ch1) = ch1 Then GotMacro = False

For i = 1 To Len(myMacroName) - 3

ch = Mid(myMacroName, i, 1)

If LCase(ch) = UCase(ch) Then

GotMacro = False

Exit For

End If

DoEvents

Next i

If Len(myMacroName) > 3 And GotMacro = True Then

myFolder = Left(myMacroName, 1)

myFullName = "http://www.wordmacrotools.com/macros/" & \_

myFolder & "/" & myMacroName

ActiveDocument.FollowHyperlink Address:=myFullName

End If

If GotMacro = True Then

ShowVisualBasicEditor = True

Else

Beep

MsgBox "Are you sure this is a macro name?"

End If

End Sub

**Sub CommentAddFromFile()**

**'** Paul Beverley - Version 04.07.24

' Adds a comment from a file of ready-made comments

myPartFilename = "hapter"

myMarker = "|"

deleteHeader = True

Selection.Expand wdParagraph

Selection.MoveEnd , -1

If deleteHeader = True Then

colonPos = InStr(Selection, ": ")

If colonPos > 0 Then Selection.MoveStart , colonPos + 1

End If

Selection.Copy

markerPos = InStr(Selection.Text, myMarker)

moveCursor = Len(Selection) - markerPos

gottaDoc = False

For Each myDoc In Documents

myName = myDoc.Name

If InStr(myName, myPartFilename) > 0 Then

gottaDoc = True

Exit For

End If

Next myDoc

If gottaDoc = False Then

Beep

myResponse = MsgBox("Can't find a file with part name:" & \_

vbCr & vbCr & myPartFilename, vbOKOnly, "CommentAddFromFile")

Exit Sub

End If

myDoc.Activate

If Selection.start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.MoveEnd , 1

If rng.Text = "-" Then Selection.MoveEnd wdWord, 2

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.start = startNow

End If

Dim cmt As Comment

Set cmt = Selection.Comments.Add(Range:=Selection.Range)

Selection.Paste

ActiveDocument.ActiveWindow.View.SplitSpecial = wdPaneNone

cmt.Edit

If markerPos > 0 Then

Selection.MoveLeft , moveCursor

Selection.MoveStart , -1

Selection.Delete

End If

End Sub

**Sub** **MacroFetchKeystrokeRead()**

' Paul Beverley - Version 20.11.21

' Reads and restores a macro's keystroke while upgrading

allKeyBits = " Alt Shift Ctrl + - "

startNow = Selection.Start

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

If Len(Selection) < 3 Then

Selection.MoveLeft , 3

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

myWord = Selection

If InStr(allKeyBits, myWord) > 0 Then GoTo keyRestore

If Len(Trim(myWord)) = 1 Then GoTo keyRestore

myMacroName = myWord

' Read current keystroke

myKeyString = ""

For Each kb In KeyBindings

If kb.KeyCategory = 2 Then ' It's a Macro

cmd = kb.Command

gottit = (InStr(cmd, "." & myMacroName) > 0)

If Left(cmd, 6) = "Normal" And gottit And \_

Right(cmd, Len(myMacroName)) = myMacroName Then

myKeyString = kb.KeyString

Exit For

End If

End If

DoEvents

Next kb

If myKeyString = "" Then myKeyString = "Unknown"

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=" " & myKeyString

' Fetch new macro

myFolder = Left(myMacroName, 1)

myFullName = "http://www.wordmacrotools.com/macros/" & \_

myFolder & "/" & myMacroName

ActiveDocument.FollowHyperlink Address:=myFullName

Exit Sub

keyRestore:

Set rng = Selection.Range.Duplicate

rng.Start = 0

numWds = rng.Words.Count

For i = numWds To 1 Step -1

myTest = Trim(rng.Words(i))

Debug.Print myTest

If InStr(allKeyBits, " " & myTest & " ") = 0 Then

' myWd = i

Exit For

End If

DoEvents

Next i

Selection.Start = rng.Words(i).End

Selection.End = startNow

myMacroName = myTest

myKeyString = Selection

Selection.Collapse wdCollapseEnd

ks = myKeyString

kCode = 0

If InStr(ks, "Ctrl+") > 0 Then

kCode = kCode + 512

ks = Replace(ks, "Ctrl+", "")

End If

If InStr(ks, "Alt+") > 0 Then

kCode = kCode + 1024

ks = Replace(ks, "Alt+", "")

End If

If InStr(ks, "Shift+") > 0 Then

kCode = kCode + 256

ks = Replace(ks, "Shift+", "")

End If

' Ordinary capital letters

If ks Like "[A-Z]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' Ordinary numbers

If ks Like "[0-9]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' F keys

If Left(ks, 1) = "F" And Len(ks) > 1 Then

aCode = 111 + Val(Replace(ks, "F", ""))

ks = ""

End If

Select Case ks

Case "!": aCode = wdKey1: kCode = kCode + 256 ' Shifted number 1

Case """": aCode = wdKey2: kCode = kCode + 256 ' Shifted number 2

Case ChrW(163): aCode = wdKey3: kCode = kCode + 256 ' Shifted number 3

Case "$": aCode = wdKey4: kCode = kCode + 256 ' Shifted number 4

Case "%": aCode = wdKey5: kCode = kCode + 256 ' Shifted number 5

Case "^": aCode = wdKey6: kCode = kCode + 256 ' Shifted number 6

Case "&": aCode = wdKey7: kCode = kCode + 256 ' Shifted number 7

Case "\*": aCode = wdKey8: kCode = kCode + 256 ' Shifted number 8

Case "(": aCode = wdKey9: kCode = kCode + 256 ' Shifted number 9

Case ")": aCode = wdKey0: kCode = kCode + 256 ' Shifted number 0

Case "'": aCode = 192

Case "-": aCode = wdKeyHyphen ' The - key

Case "#": aCode = 222

Case ",": aCode = wdKeyComma ' The , key

Case ".": aCode = wdKeyPeriod ' The key

Case "/": aCode = wdKeySlash

Case ":": aCode = wdKeySemiColon: kCode = kCode + 256

Case ";": aCode = wdKeySemiColon ' The ; key

Case "?": aCode = wdKeySlash: kCode = kCode + 256

Case "@": aCode = 192: kCode = kCode + 256

Case "[": aCode = wdKeyOpenSquareBrace ' The ] key

Case "\": aCode = wdKeyBackSlash ' The \ key

Case "]": aCode = wdKeyCloseSquareBrace ' The ] key

Case "\_": aCode = wdKeyHyphen ' Does it need? kCode = kCode + 256

Case "`": aCode = 223

Case "{": aCode = wdKeyOpenSquareBrace: kCode = kCode + 256

Case "}": aCode = wdKeyCloseSquareBrace: kCode = kCode + 256

Case "~": aCode = 222: kCode = kCode + 256

Case "+": aCode = wdKeyEquals ' kCode = kCode + 256

Case "<": aCode = wdKeyComma: kCode = kCode + 256

Case "=": aCode = wdKeyEquals ' The = key

Case ">": aCode = wdKeyPeriod: kCode = kCode + 256

Case "Backspace": aCode = wdKeyBackspace ' The Backspace key

Case "Clear (Num 5)": aCode = wdKeyNumeric5: kCode = kCode + 256

Case "Delete": aCode = wdKeyDelete ' The Delete key

Case "Down": aCode = 40 ' The Down Cursor key

Case "End": aCode = 35 ' The EndD key

Case "Home": aCode = wdKeyHome ' The HOME key

Case "Insert": aCode = wdKeyInsert ' The Insert key

Case "Left": aCode = 37 ' The Left Cursor key

Case "Num -": aCode = wdKeyNumericSubtract ' The - key on the keypad

Case "Num \*": aCode = wdKeyNumericMultiply ' The \* key on the keypad

Case "Num /": aCode = wdKeyNumericDivide ' The / key on the keypad

Case "Num +": aCode = wdKeyNumericAdd ' The + key on the keypad

Case "Num 0": aCode = wdKeyNumeric0 ' The 0 key on the keypad

Case "Num 1": aCode = wdKeyNumeric1 ' The 1 key on the keypad

Case "Num 2": aCode = wdKeyNumeric2 ' The 2 key on the keypad

Case "Num 3": aCode = wdKeyNumeric3 ' The 3 key on the keypad

Case "Num 4": aCode = wdKeyNumeric4 ' The 4 key on the keypad

Case "Num 5": aCode = wdKeyNumeric5 ' The 5 key on the keypad

Case "Num 6": aCode = wdKeyNumeric6 ' The 6 key on the keypad

Case "Num 7": aCode = wdKeyNumeric7 ' The 7 key on the keypad

Case "Num 8": aCode = wdKeyNumeric8 ' The 8 key on the keypad

Case "Num 9": aCode = wdKeyNumeric9 ' The 9 key on the keypad

Case "Page Down": aCode = 34 ' The Page Down key

Case "Page Up": aCode = 33 ' The Page Up key

Case "Return": aCode = wdKeyReturn ' The Enter or Return key

Case "Right": aCode = 39 ' The Right Cursor key

Case "Space": aCode = wdKeySpacebar ' The Spacebar key

Case "Up": aCode = 38 ' The Up Cursor key

End Select

kCode = kCode + aCode

hexCode = Replace(Hex(kCode), "FFFF", "")

KeyBindings.Add KeyCode:=kCode, \_

KeyCategory:=wdKeyCategoryMacro, Command:=myMacroName

Beep

MsgBox "Keystroke restored"

End Sub

**Sub MacroFetchKeystrokeReadMac()**

' Paul Beverley - Version 20.11.21

' Reads and restores a macro's keystroke while upgrading

allKeyBits = " Alt Shift Ctrl Control Command Option + - "

startNow = Selection.Start

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

If Len(Selection) < 3 Then

Selection.MoveLeft , 3

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

myWord = Selection

If InStr(allKeyBits, myWord) > 0 Then GoTo keyRestore

If Len(Trim(myWord)) = 1 Then GoTo keyRestore

myMacroName = myWord

' Read current keystroke

myKeyString = ""

For Each kb In KeyBindings

If kb.KeyCategory = 2 Then ' It's a Macro

cmd = kb.Command

gottit = (InStr(cmd, "." & myMacroName) > 0)

If Left(cmd, 6) = "Normal" And gottit And \_

Right(cmd, Len(myMacroName)) = myMacroName Then

myKeyString = kb.KeyString

Exit For

End If

End If

Do Events

Next kb

If myKeyString = "" Then myKeyString = "No keystroke set"

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=" " & myKeyString

' Fetch new macro

myFolder = Left(myMacroName, 1)

myFullName = "http://www.wordmacrotools.com/macros/" & \_

myFolder & "/" & myMacroName

ActiveDocument.FollowHyperlink Address:=myFullName

Exit Sub

keyRestore:

Set rng = Selection.Range.Duplicate

rng.Start = 0

numWds = rng.Words.Count

For i = numWds To 1 Step -1

myTest = Trim(rng.Words(i))

Debug.Print myTest

If InStr(allKeyBits, " " & myTest & " ") = 0 Then

' myWd = i

Exit For

End If

DoEvents

Next i

Selection.Start = rng.Words(i).End

Selection.End = startNow

myMacroName = myTest

myKeyString = Selection

Selection.Collapse wdCollapseEnd

ks = myKeyString

kCode = 0

If InStr(ks, "Command+") > 0 Then

kCode = kCode + 256

ks = Replace(ks, "Command+", "")

End If

If InStr(ks, "Control+") > 0 Then

kCode = kCode + 4096

ks = Replace(ks, "Control+", "")

End If

If InStr(ks, "Option+") > 0 Then

kCode = kCode + 2048

ks = Replace(ks, "Option+", "")

End If

If InStr(ks, "Shift+") > 0 Then

kCode = kCode + 512

ks = Replace(ks, "Shift+", "")

End If

' Ordinary capital letters

If ks Like "[A-Z]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' Ordinary numbers

If ks Like "[0-9]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' F keys

If Left(ks, 1) = "F" And Len(ks) > 1 Then

aCode = 111 + Val(Replace(ks, "F", ""))

ks = ""

End If

Select Case ks

Case "!": aCode = wdKey1: kCode = kCode + 256 ' Shifted number 1

Case """": aCode = wdKey2: kCode = kCode + 256 ' Shifted number 2

Case ChrW(163): aCode = wdKey3: kCode = kCode + 256 ' Shifted number 3

Case "$": aCode = wdKey4: kCode = kCode + 256 ' Shifted number 4

Case "%": aCode = wdKey5: kCode = kCode + 256 ' Shifted number 5

Case "^": aCode = wdKey6: kCode = kCode + 256 ' Shifted number 6

Case "&": aCode = wdKey7: kCode = kCode + 256 ' Shifted number 7

Case "\*": aCode = wdKey8: kCode = kCode + 256 ' Shifted number 8

Case "(": aCode = wdKey9: kCode = kCode + 256 ' Shifted number 9

Case ")": aCode = wdKey0: kCode = kCode + 256 ' Shifted number 0

Case "'": aCode = 192

Case "-": aCode = wdKeyHyphen ' The - key

Case "#": aCode = 222

Case ",": aCode = wdKeyComma ' The , key

Case ".": aCode = wdKeyPeriod ' The key

Case "/": aCode = wdKeySlash

Case ":": aCode = wdKeySemiColon: kCode = kCode + 256

Case ";": aCode = wdKeySemiColon ' The ; key

Case "?": aCode = wdKeySlash: kCode = kCode + 256

Case "@": aCode = 192: kCode = kCode + 256

Case "[": aCode = wdKeyOpenSquareBrace ' The ] key

Case "\": aCode = wdKeyBackSlash ' The \ key

Case "]": aCode = wdKeyCloseSquareBrace ' The ] key

Case "\_": aCode = wdKeyHyphen ' Does it need? kCode = kCode + 256

Case "`": aCode = 223

Case "{": aCode = wdKeyOpenSquareBrace: kCode = kCode + 256

Case "}": aCode = wdKeyCloseSquareBrace: kCode = kCode + 256

Case "~": aCode = 222: kCode = kCode + 256

Case "+": aCode = wdKeyEquals ' kCode = kCode + 256

Case "<": aCode = wdKeyComma: kCode = kCode + 256

Case "=": aCode = wdKeyEquals ' The = key

Case ">": aCode = wdKeyPeriod: kCode = kCode + 256

Case "Backspace": aCode = wdKeyBackspace ' The Backspace key

Case "Clear (Num 5)": aCode = wdKeyNumeric5: kCode = kCode + 256

Case "Delete": aCode = wdKeyDelete ' The Delete key

Case "Down": aCode = 40 ' The Down Cursor key

Case "End": aCode = 35 ' The EndD key

Case "Home": aCode = wdKeyHome ' The HOME key

Case "Insert": aCode = wdKeyInsert ' The Insert key

Case "Left": aCode = 37 ' The Left Cursor key

Case "Num -": aCode = wdKeyNumericSubtract ' The - key on the keypad

Case "Num \*": aCode = wdKeyNumericMultiply ' The \* key on the keypad

Case "Num /": aCode = wdKeyNumericDivide ' The / key on the keypad

Case "Num +": aCode = wdKeyNumericAdd ' The + key on the keypad

Case "Num 0": aCode = wdKeyNumeric0 ' The 0 key on the keypad

Case "Num 1": aCode = wdKeyNumeric1 ' The 1 key on the keypad

Case "Num 2": aCode = wdKeyNumeric2 ' The 2 key on the keypad

Case "Num 3": aCode = wdKeyNumeric3 ' The 3 key on the keypad

Case "Num 4": aCode = wdKeyNumeric4 ' The 4 key on the keypad

Case "Num 5": aCode = wdKeyNumeric5 ' The 5 key on the keypad

Case "Num 6": aCode = wdKeyNumeric6 ' The 6 key on the keypad

Case "Num 7": aCode = wdKeyNumeric7 ' The 7 key on the keypad

Case "Num 8": aCode = wdKeyNumeric8 ' The 8 key on the keypad

Case "Num 9": aCode = wdKeyNumeric9 ' The 9 key on the keypad

Case "Page Down": aCode = 34 ' The Page Down key

Case "Page Up": aCode = 33 ' The Page Up key

Case "Return": aCode = wdKeyReturn ' The Enter or Return key

Case "Right": aCode = 39 ' The Right Cursor key

Case "Space": aCode = wdKeySpacebar ' The Spacebar key

Case "Up": aCode = 38 ' The Up Cursor key

End Select

kCode = kCode + aCode

hexCode = Replace(Hex(kCode), "FFFF", "")

KeyBindings.Add KeyCode:=kCode, \_

KeyCategory:=wdKeyCategoryMacro, Command:=myMacroName

Beep

MsgBox "Keystroke restored"

End Sub

**Sub MacroFetchUpdate()**

' Paul Beverley - Version 21.08.22

' Updates an existing macro, preserving the keystroke

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

If Len(Selection) < 3 Then

Selection.MoveLeft , 3

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

myMacroName = Selection

Selection.Expand wdParagraph

If InStr(Selection, "+") > 0 Then GoTo restoreKeystroke

' Attempt to read keystroke

myKeyString = ""

For Each kb In KeyBindings

If kb.KeyCategory = 2 Then ' It's a Macro

cmd = kb.Command

gottit = (InStr(cmd, "." & myMacroName) > 0)

If Left(cmd, 6) = "Normal" And gottit And \_

Right(cmd, Len(myMacroName)) = myMacroName Then

myKeyString = kb.KeyString

Exit For

End If

End If

Next kb

Debug.Print myKeyString

If myKeyString > "" Then

Documents.Add

Selection.TypeText Text:=myMacroName & vbTab & myKeyString & vbCr

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Now, you need to..." \_

& vbCr & vbCr & "1) Copy the text of the new version" \_

& vbCr & vbCr & "2) Delete the existing copy of the macro" \_

& vbCr & vbCr & "3) Paste the new version into Visual Basic (VBA)" \_

& vbCr & vbCr & vbCr & " Finally, rerun this macro", \_

vbOKOnly, "MacroFetchUpdate")

Else

MsgBox "Macro " & myMacroName & " has no keystroke assigned to it."

End If

Call MacroFetch

Exit Sub

restoreKeystroke:

dotPos = InStr(Selection, ".")

If dotPos > 0 Then Selection.MoveStart , dotPos

dotPos = InStr(Selection, ".")

If dotPos > 0 Then Selection.MoveStart , dotPos

tabPos = InStr(Selection, vbTab)

Selection.MoveEnd , -1

myKeyString = Mid(Selection, tabPos + 1)

myResponse = MsgBox("Set keystroke for macro: " & myMacroName & \_

vbCr & vbCr & " to: " & myKeyString & "?", \_

vbQuestion + vbYesNoCancel, "MacroFetchUpdate")

If myResponse = vbYes Then

ks = myKeyString

kCode = 0

If InStr(ks, "Ctrl+") > 0 Then

kCode = kCode + 512

ks = Replace(ks, "Ctrl+", "")

End If

If InStr(ks, "Alt+") > 0 Then

kCode = kCode + 1024

ks = Replace(ks, "Alt+", "")

End If

If InStr(ks, "Shift+") > 0 Then

kCode = kCode + 256

ks = Replace(ks, "Shift+", "")

End If

' Ordinary capital letters

If ks Like "[A-Z]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' Ordinary numbers

If ks Like "[0-9]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' F keys

If Left(ks, 1) = "F" And Len(ks) > 1 Then

aCode = 111 + Val(Replace(ks, "F", ""))

ks = ""

End If

Select Case ks

Case "!": aCode = wdKey1: kCode = kCode + 256 ' Shifted number 1

Case """": aCode = wdKey2: kCode = kCode + 256 ' Shifted number 2

Case ChrW(163): aCode = wdKey3: kCode = kCode + 256 ' Shifted number 3

Case "$": aCode = wdKey4: kCode = kCode + 256 ' Shifted number 4

Case "%": aCode = wdKey5: kCode = kCode + 256 ' Shifted number 5

Case "^": aCode = wdKey6: kCode = kCode + 256 ' Shifted number 6

Case "&": aCode = wdKey7: kCode = kCode + 256 ' Shifted number 7

Case "\*": aCode = wdKey8: kCode = kCode + 256 ' Shifted number 8

Case "(": aCode = wdKey9: kCode = kCode + 256 ' Shifted number 9

Case ")": aCode = wdKey0: kCode = kCode + 256 ' Shifted number 0

Case "'": aCode = 192

Case "-": aCode = wdKeyHyphen ' The - key

Case "#": aCode = 222

Case ",": aCode = wdKeyComma ' The , key

Case ".": aCode = wdKeyPeriod ' The key

Case "/": aCode = wdKeySlash

Case ":": aCode = wdKeySemiColon: kCode = kCode + 256

Case ";": aCode = wdKeySemiColon ' The ; key

Case "?": aCode = wdKeySlash: kCode = kCode + 256

Case "@": aCode = 192: kCode = kCode + 256

Case "[": aCode = wdKeyOpenSquareBrace ' The ] key

Case "\": aCode = wdKeyBackSlash ' The \ key

Case "]": aCode = wdKeyCloseSquareBrace ' The ] key

Case "\_": aCode = wdKeyHyphen ' Does it need? kCode = kCode + 256

Case "`": aCode = 223

Case "{": aCode = wdKeyOpenSquareBrace: kCode = kCode + 256

Case "}": aCode = wdKeyCloseSquareBrace: kCode = kCode + 256

Case "~": aCode = 222: kCode = kCode + 256

Case "+": aCode = wdKeyEquals ' kCode = kCode + 256

Case "<": aCode = wdKeyComma: kCode = kCode + 256

Case "=": aCode = wdKeyEquals ' The = key

Case ">": aCode = wdKeyPeriod: kCode = kCode + 256

Case "Backspace": aCode = wdKeyBackspace ' The Backspace key

Case "Clear (Num 5)": aCode = wdKeyNumeric5: kCode = kCode + 256

Case "Delete": aCode = wdKeyDelete ' The Delete key

Case "Down": aCode = 40 ' The Down Cursor key

Case "End": aCode = 35 ' The EndD key

Case "Home": aCode = wdKeyHome ' The HOME key

Case "Insert": aCode = wdKeyInsert ' The Insert key

Case "Left": aCode = 37 ' The Left Cursor key

Case "Num -": aCode = wdKeyNumericSubtract ' The - key on the keypad

Case "Num \*": aCode = wdKeyNumericMultiply ' The \* key on the keypad

Case "Num /": aCode = wdKeyNumericDivide ' The / key on the keypad

Case "Num +": aCode = wdKeyNumericAdd ' The + key on the keypad

Case "Num 0": aCode = wdKeyNumeric0 ' The 0 key on the keypad

Case "Num 1": aCode = wdKeyNumeric1 ' The 1 key on the keypad

Case "Num 2": aCode = wdKeyNumeric2 ' The 2 key on the keypad

Case "Num 3": aCode = wdKeyNumeric3 ' The 3 key on the keypad

Case "Num 4": aCode = wdKeyNumeric4 ' The 4 key on the keypad

Case "Num 5": aCode = wdKeyNumeric5 ' The 5 key on the keypad

Case "Num 6": aCode = wdKeyNumeric6 ' The 6 key on the keypad

Case "Num 7": aCode = wdKeyNumeric7 ' The 7 key on the keypad

Case "Num 8": aCode = wdKeyNumeric8 ' The 8 key on the keypad

Case "Num 9": aCode = wdKeyNumeric9 ' The 9 key on the keypad

Case "Page Down": aCode = 34 ' The Page Down key

Case "Page Up": aCode = 33 ' The Page Up key

Case "Return": aCode = wdKeyReturn ' The Enter or Return key

Case "Right": aCode = 39 ' The Right Cursor key

Case "Space": aCode = wdKeySpacebar ' The Spacebar key

Case "Up": aCode = 38 ' The Up Cursor key

End Select

kCode = kCode + aCode

hexCode = Replace(Hex(kCode), "FFFF", "")

KeyBindings.Add KeyCode:=kCode, \_

KeyCategory:=wdKeyCategoryMacro, Command:=myMacroName

Beep

MsgBox "Keystroke restored"

Else

Beep

End If

End Sub

**Sub MacroNameToLink()**

**'** Paul Beverley - Version 31.06.24

' Makes URL for macro name in clipboard or at cursor and puts in clipboard

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

myMacroName = Selection

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

myFolder = Left(myMacroName, 1)

myName = ""

For i = 1 To Len(myMacroName)

ch = Mid(myMacroName, i, 1)

If AscW(ch) > 128 Then

myName = myName & "%" & Hex(AscW(ch))

myFolder = "ES"

Else

myName = myName & ch

End If

Next i

myFullName = "https://www.wordmacrotools.com/macros/" & myFolder & "/" & myName

Selection.InsertAfter Text:=myFullName

Selection.Range.Copy

WordBasic.EditUndo

End Sub

**Sub AddKeys()**

' Paul Beverley - Version 08.07.20

' Adds keybindings to macros

' Sub CustomKeys = Shift-Alt-K

' Sub AccentAlyse = Shift-Alt-A

' Sub DocAlyse = Shift-Alt-D

' Sub FindBack = Alt-Left

' Sub FindBackCase = Shift-Alt-Left

' Sub FindFwd = Alt-Right

' Sub FindFwdCase = Shift-Alt-Right

' Sub FRedit = Shift-Alt-F

' Sub HyphenAlyse = Shift-Alt-H

' Sub InstantFindDown = Shift-Alt-Down

' Sub InstantFindUp = Shift-Alt-Up

' Sub IStoIZ = Shift-Alt-Z

' Sub IZIScount = Shift-Alt-I

' Sub IZtoIS = Shift-Alt-S

' Sub MultiSwitch = Alt-Q

' Sub CopyTextSimple = Alt-C

' Sub MultiFileText = Alt-T

' Sub ProperNounAlyse = Shift-Alt-P

' Sub SpellingErrorHighlighter = Shift-Alt-E

' Sub SpellingErrorLister = Shift-Alt-L

' Sub UKUScount = Shift-Alt-U

' Sub WhatChar = Shift-Alt-Slash

' Sub WordPairAlyse = Shift-Alt-W

' Sub MacroMenu = Shift-Alt-M

keyCount = 0

For Each myPara In ActiveDocument.Paragraphs

myLine = Replace(myPara.Range.Text, vbCr, "")

myLine = Replace(myLine, " ", "")

If Left(myLine, 6) = "EndSub" Then

If keyCount > 0 Then

MsgBox keyCount & " shortcut keys programmed."

Else

Beep

MsgBox "Please ensure that 'The\_Starter\_Macros' document is" & vbCr \_

& "still open on screen, and that the cursor" & vbCr \_

& "is in that file. Now re-run this macro."

End If

Exit Sub

End If

gotOne = False

If Left(myLine, 4) = "'Sub" Then myLine = Mid(myLine, 5): gotOne = True

eqPos = InStr(myLine, "=")

If gotOne And eqPos > 0 Then

keyCount = keyCount + 1

myMacroName = Left(myLine, eqPos - 1)

myKeyCode = 0

myKeyText = Mid(myLine, eqPos + 1)

i = Len(myKeyText)

myKeyText = Replace(myKeyText, "Ctrl-", "")

If Len(myKeyText) < i Then myKeyCode = myKeyCode + wdKeyControl

i = Len(myKeyText)

myKeyText = Replace(myKeyText, "Shift-", "")

If Len(myKeyText) < i Then myKeyCode = myKeyCode + wdKeyShift

i = Len(myKeyText)

myKeyText = Replace(myKeyText, "Alt-", "")

If Len(myKeyText) < i Then myKeyCode = myKeyCode + wdKeyAlt

If Len(myKeyText) = 1 Then

myKeyCode = myKeyCode + Asc(myKeyText)

Else

If Asc(myKeyText) = Asc("F") Then

aCode = 111 + Val(Replace(myKeyText, "F", ""))

Else

Select Case myKeyText

Case "Up": aCode = vbKeyUp ' The Up Cursor key

Case "Down": aCode = vbKeyDown ' The Down Cursor key

Case "Right": aCode = vbKeyRight ' The Right Cursor key

Case "Left": aCode = vbKeyLeft ' The Left Cursor key

Case "BackSingleQuote": aCode = wdKeyBackSingleQuote ' The ` key

Case "BackSlash": aCode = wdKeyBackSlash ' The \ key

Case "Backspace": aCode = wdKeyBackspace ' The Backspace key

Case "CloseSquareBrace": aCode = wdKeyCloseSquareBrace ' The ] key

Case "Comma": aCode = wdKeyComma ' The , key

Case "Delete": aCode = wdKeyDelete ' The Delete key

Case "End": aCode = wdKeyEnd ' The EndD key

Case "Equals": aCode = wdKeyEquals ' The = key

Case "Home": aCode = wdKeyHome ' The HOME key

Case "Hyphen": aCode = wdKeyHyphen ' The - key

Case "Insert": aCode = wdKeyInsert ' The Insert key

Case "NumAdd": aCode = wdKeyNumericAdd ' The + key on the keypad

Case "NumDecimal": aCode = wdKeyNumericDecimal

' The . key on the keypad

Case "NumDivide": aCode = wdKeyNumericDivide

' The / key on the keypad

Case "NumMultiply": aCode = wdKeyNumericMultiply

' The \* key on the keypad

Case "NumSubtract": aCode = wdKeyNumericSubtract

' The - key on the keypad

Case "OpenSquareBrace": aCode = wdKeyOpenSquareBrace

' The [ key

Case "PageDown": aCode = wdKeyPageDown ' The Page Down key

Case "PageUp": aCode = wdKeyPageUp ' The Page Up key

Case "Period": aCode = wdKeyPeriod ' The key

Case "Return": aCode = wdKeyReturn ' The Enter or Return key

Case "SemiColon": aCode = wdKeySemiColon ' The ; key

Case "SingleQuote": aCode = wdKeySingleQuote ' The ' key

Case "Slash": aCode = wdKeySlash ' The / key

Case "Spacebar": aCode = wdKeySpacebar ' The Spacebar key

Case Else: MsgBox ("Error - Can't find: " & myLine): Exit Sub

End Select

End If

myKeyCode = myKeyCode + aCode

End If

KeyBindings.Add KeyCode:=myKeyCode, KeyCategory:=wdKeyCategoryMacro, \_

Command:=myMacroName

End If

Next myPara

If keyCount > 0 Then

MsgBox keyCount & " shortcut keys programmed."

Else

Beep

MsgBox "Please reopen the Word file containing" & vbCr \_

& "all the macros: TheStarterMacros."

End If

End Sub

**Sub DocAlyse()**

**'** Paul Beverley - Version 07.05.24

' Analyses various aspects of a document

' prompts to count number of tests

cc = 53

myScreenOff = True

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox(" DocAlyse" & vbCr & vbCr & \_

"Analyse this document?", vbQuestion \_

+ vbYesNoCancel, "DocAlyse")

If myResponse <> vbYes Then Exit Sub

End If

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

ss = "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" & \_

"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rngOld = ActiveDocument.Content

Documents.Add

Set tempDoc = ActiveDocument

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

Selection.EndKey Unit:=wdStory

If ActiveDocument.Footnotes.Count > 0 Then

ActiveDocument.StoryRanges(wdFootnotesStory).Copy

Selection.Paste

Selection.Collapse wdCollapseEnd

End If

If ActiveDocument.Endnotes.Count > 0 Then

ActiveDocument.StoryRanges(wdEndnotesStory).Copy

Selection.Paste

Selection.Collapse wdCollapseEnd

End If

Set endTempfile = ActiveDocument.Content

endTempfile.Collapse wdCollapseEnd

' collect text in all the textboxes (if any)

sh = ActiveDocument.Shapes.Count

If sh > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

shp.TextFrame.TextRange.Copy

endTempfile.Select

Selection.Paste

End If

End If

Next

End If

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.Revisions.AcceptAll

DoEvents

DoEvents

' Use main file for italic 'et al' count...

myTot = ActiveDocument.Range.End

cc = cc - 1

DoEvents

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<et al>"

.Font.Italic = True

.Replacement.Text = "^&!"

.Wrap = wdFindContinue

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

italEtAls = ActiveDocument.Range.End - myTot

If italEtAls > 0 Then WordBasic.EditUndo

' ...and superscript degree count

cc = cc - 1

DoEvents

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[oO0]"

.Font.Superscript = True

.Replacement.Text = "vbvb"

.Replacement.Font.Superscript = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

funnyDegrees = (ActiveDocument.Range.End - myTot) / 3

.ClearFormatting

.Replacement.ClearFormatting

.Text = " vbvb"

.Replacement.Text = "^&!"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

funnyDegreesSp = ActiveDocument.Range.End - myTot - funnyDegrees \* 3

If funnyDegreesSp > 0 Then WordBasic.EditUndo

If funnyDegrees > 0 Then WordBasic.EditUndo

DoEvents

Selection.HomeKey Unit:=wdStory

Set rngOld = ActiveDocument.Content

ActiveDocument.TrackRevisions = myTrack

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

myEnd = rng.End

Set rng2 = ActiveDocument.Content

rng.Collapse wdCollapseEnd

rng.Text = rng2.Text

Set rng3 = ActiveDocument.Content

rng3.End = myEnd - 1

rng3.Select

Selection.Delete

myRslt = ""

Set rng = ActiveDocument.Content

myTot = ActiveDocument.Range.End

CR = vbCr: CR2 = CR & CR

tr = Chr(9) & "0zczc" & CR: SP = " "

Selection.HomeKey Unit:=wdStory

Set newDoc = ActiveDocument

' Ten or 10

cc = cc - 1

DoEvents

myTot = ActiveDocument.Range.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<ten>"

.Replacement.Text = "!^&"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = " <10>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "ten" & vbTab & \_

Trim(Str(i)) & CR & "10" & vbTab & Trim(Str(g)) & CR2

' spelt-out lower-case numbers over nine

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "<[efnst][efghinorvwx]{2,4}ty"

rng.Find.Execute Replace:=wdReplaceAll

aa = ActiveDocument.Range.End - myTot

If aa > 0 Then WordBasic.EditUndo

rng.Find.Text = "<ten>"

rng.Find.Execute Replace:=wdReplaceAll

ab = ActiveDocument.Range.End - myTot

If ab > 0 Then WordBasic.EditUndo

rng.Find.Text = "<eleven>"

rng.Find.Execute Replace:=wdReplaceAll

ac = ActiveDocument.Range.End - myTot

If ac > 0 Then WordBasic.EditUndo

rng.Find.Text = "<twelve>"

rng.Find.Execute Replace:=wdReplaceAll

ad = ActiveDocument.Range.End - myTot

If ad > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[efnst][efghinuorvwx]{2,4}teen>"

rng.Find.Execute Replace:=wdReplaceAll

ae = ActiveDocument.Range.End - myTot

If ae > 0 Then WordBasic.EditUndo

rng.Find.Text = "<hundred>"

rng.Find.Execute Replace:=wdReplaceAll

af = ActiveDocument.Range.End - myTot

If af > 0 Then WordBasic.EditUndo

If aa + ab + ac + ad + ae + af > 0 Then myRslt = myRslt & \_

"spelt-out numbers (11-999)" & vbTab & \_

Trim(Str(aa + ab + ac + ad + ae + af)) & CR2

' Four-digit numbers

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[!.]<[0-9]{4}>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

' take off 20xx dates

rng.Find.Text = "[!.]<20[0-9]{2}>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

' take off 13xx to 19xx dates

rng.Find.Text = "[!.]<1[3-9][0-9]{2}>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

i = i - g - k

If i < 0 Then i = 0

' Four figs with comma

rng.Find.Text = "[!.]<[0-9],[0-9]{3}>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

' Four figs with hard or ordinary space

rng.Find.Text = "[!.]<[0-9][^0160^32][0-9]{3}>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

If i + g + k > 0 Then

myRslt = myRslt & "Four-digit numbers:" & CR \_

& "nnnn" & vbTab & Trim(Str(i)) & CR \_

& "n,nnn" & vbTab & Trim(Str(g)) & CR \_

& "n nnn" & vbTab & Trim(Str(k)) & CR2

End If

' Dates with 'mid' in front

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "mid [0-9]{4}"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "mid-[0-9]{4}"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "mid[0-9]{4}"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

If i + g + k > 0 Then

myRslt = myRslt & "mid 1900(s)" & vbTab \_

& Trim(Str(i)) & CR & "mid-1900(s)" & vbTab & \_

Trim(Str(g)) & CR & "mid1900(s)" & vbTab & \_

Trim(Str(k)) & CR2

End If

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "mid [0-9]{2}[!0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "mid-[0-9]{2}[!0-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "mid[0-9]{2}[!0-9]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

If i + g + k > 0 Then

myRslt = myRslt & "mid 90(s)" & vbTab \_

& Trim(Str(i)) & CR & "mid-90(s)" & vbTab & \_

Trim(Str(g)) & CR & "mid90(s)" & vbTab & \_

Trim(Str(k)) & CR2

End If

' Serial comma/not serial comma

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[a-zA-Z\-]{1,}, [a-zA-Z\-]{1,}, and "

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt & "serial comma" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "[a-zA-Z\-]{1,}, [a-zA-Z\-]{1,} and "

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt & "no serial comma" & vbTab & Trim(Str(i)) & CR2

' hard spaces

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "^s"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

' hard hyphens

rng.Find.Text = "^~"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

myRslt = myRslt & "hard spaces" & vbTab & Trim(Str(i)) \_

& CR & "hard hyphens" & vbTab & Trim(Str(g)) & CR2

' Single/double quotes

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = ChrW(8216)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

singleCurl = i

myRslt = myRslt & "curly open single quote" & vbTab & \_

Trim(Str(i)) & CR

rng.Find.Text = ChrW(8220)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt & "curly open double quote" & vbTab & \_

Trim(Str(i)) & CR

rng.Find.Text = Chr(39)

rng.Find.MatchWildcards = True

rng.Find.MatchCase = True

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt & "straight single quote" & vbTab & \_

Trim(Str(i)) & CR

rng.Find.Text = Chr(34)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt & "straight double quote" & vbTab & \_

Trim(Str(i)) & CR2

' etc(.)

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "<etc[!.]"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "<etc."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<etc. [A-Z]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<etc.^13"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

If h + i + g + k > 0 Then myRslt = myRslt & "etc" & \_

vbTab & Trim(Str(h)) & CR & "etc." & vbTab & \_

Trim(Str(i - g - k)) & CR2

' et al(.)

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "<et al[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<et al."

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

If g + i + italEtAls > 0 Then myRslt = myRslt & "et al." \_

& vbTab & Trim(Str(g)) & CR & "et al (italic, total)" & \_

vbTab & Trim(Str(italEtAls)) & CR & "et al (no dot)" & \_

vbTab & Trim(Str(i)) & CR2

' i.e./ie

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "i.e."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt

rng.Find.Text = "<ie>"

rng.Find.MatchWildcards = True

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "ie" & vbTab & Trim(Str(g)) & CR \_

& "i.e." & vbTab & Trim(Str(i)) & CR2

' e.g./eg

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "e.g."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<eg>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "eg" & vbTab & Trim(Str(g)) & CR \_

& "e.g." & vbTab & Trim(Str(i)) & CR2

' Initials with surnames

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "<[A-Z]. [A-Z]. [A-Z][a-z]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[A-Z][a-z]{2,}, [A-Z]. [A-Z]. "

rng.Find.Execute Replace:=wdReplaceAll

i2 = ActiveDocument.Range.End - myTot

If i2 > 0 Then WordBasic.EditUndo

aBit = "J. L. B. Matekoni" & vbTab & Trim(Str(i + i2)) & CR

g = i + i2

rng.Find.Text = "<[A-Z].[A-Z]. [A-Z][a-z]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[A-Z][a-z]{2,}, [A-Z].[A-Z]."

rng.Find.Execute Replace:=wdReplaceAll

i2 = ActiveDocument.Range.End - myTot

If i2 > 0 Then WordBasic.EditUndo

aBit = aBit & "J.L.B. Matekoni" & vbTab & Trim(Str(i + i2)) & CR

g = g + i + i2

rng.Find.Text = "<[A-Z] [A-Z] [A-Z][a-z]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[A-Z][a-z]{2,}, [A-Z] [A-Z] "

rng.Find.Execute Replace:=wdReplaceAll

i2 = ActiveDocument.Range.End - myTot

If i2 > 0 Then WordBasic.EditUndo

aBit = aBit & "J L B Matekoni" & vbTab & Trim(Str(i + i2)) & CR

g = g + i + i2

rng.Find.Text = "<[A-Z]{2}> [A-Z][a-z]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[A-Z][a-z]{2,}, [A-Z]{2}"

rng.Find.Execute Replace:=wdReplaceAll

i2 = ActiveDocument.Range.End - myTot

If i2 > 0 Then WordBasic.EditUndo

aBit = aBit & "JLB Matekoni" & vbTab & Trim(Str(i + i2)) & \_

" (Beware! This can be inflated by, e.g. BBC Enterprises.)" & CR2

' Convention for page numbers

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "<p. [1-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<pp. [1-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

k = i + g

aBit = "p/pp. 123" & vbTab & Trim(Str(k)) & CR

rng.Find.Text = "<p.[1-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<pp.[1-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

aBit = aBit & "p/pp.123" & vbTab & Trim(Str(i + g)) & CR

k = k + i + g

rng.Find.Text = "<p [1-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<pp [1-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

aBit = aBit & "p/pp 123" & vbTab & Trim(Str(i + g)) & CR

k = k + i + g

rng.Find.Text = "<p[1-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<pp[1-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

aBit = aBit & "p/pp123" & vbTab & Trim(Str(i + g)) & CR2

If k + i + g > 0 Then myRslt = myRslt & aBit

' Convention for ed/eds/edn

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "<ed>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<eds>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<edn>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "<ed."

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "<eds."

rng.Find.Execute Replace:=wdReplaceAll

m = ActiveDocument.Range.End - myTot

If m > 0 Then WordBasic.EditUndo

rng.Find.Text = "<edn."

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

If k + m + j + i + g + h > 0 Then myRslt = myRslt \_

& "ed" & vbTab & Trim(Str(i)) & CR & "eds" \_

& vbTab & Trim(Str(g)) & CR & "edn" & vbTab & \_

Trim(Str(h)) & CR & "ed." \_

& vbTab & Trim(Str(k)) & CR & "eds." & vbTab & \_

Trim(Str(m)) & CR & "edn." \_

& vbTab & Trim(Str(j)) & CR2

' Convention for am/pm

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[1-9][ap]m"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

aBit = "2pm" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "[1-9][ap].m."

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

aBit = aBit & "2p.m." & vbTab & Trim(Str(g)) & CR

rng.Find.Text = "[1-9] [ap]m"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

aBit = aBit & "2 pm" & vbTab & Trim(Str(k)) & CR

rng.Find.Text = "[1-9] [ap].m."

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

aBit = aBit & "2 p.m." & vbTab & Trim(Str(h)) & CR2

If k + i + g + h > 0 Then myRslt = myRslt & aBit

' US/UK spelling

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[bpiv]our[ ,.s]"

rng.Find.Execute Replace:=wdReplaceAll

A = ActiveDocument.Range.End - myTot

If A > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-z]{3,}elling>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-z]{3,}elled>"

rng.Find.Execute Replace:=wdReplaceAll

f = ActiveDocument.Range.End - myTot

If f > 0 Then WordBasic.EditUndo

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[bpiv]or[ ,.s]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "rior[ ,.s]"

rng.Find.Execute Replace:=wdReplaceAll

q = ActiveDocument.Range.End - myTot

If q > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-z]{3,}eling>"

rng.Find.Execute Replace:=wdReplaceAll

v = ActiveDocument.Range.End - myTot

If v > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-z]{3,}eled>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

If A + g + f + i + q + v + k > 0 Then myRslt = \_

myRslt & "UK spelling (appx)" & vbTab & \_

Trim(Str(A + g + f)) & CR & \_

"US spelling (appx)" & vbTab & \_

Trim(Str(i - q + v + k)) & CR & \_

"(For a more accurate count, please use UKUScount.)" & CR2

' US/UK punctuation

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[a-zA-Z]['""" & ChrW(8217) & ChrW(8221) & "][,.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-zA-Z][,.]['""" & ChrW(8217) & ChrW(8221) & "][,.]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

If i + k > 0 Then myRslt = myRslt & \_

"UK punctuation (appx)" & vbTab & \_

Trim(Str(i)) & CR & "US punctuation (appx)" \_

& vbTab & Trim(Str(k)) & CR2

' Initial capital after colon?

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[a-zA-Z]: [A-Z][a-z]"

rng.Find.Execute Replace:=wdReplaceAll

dfgsdfg = ActiveDocument.Range.End

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-zA-Z]: [a-z]"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

If i + j > 0 Then myRslt = myRslt & \_

"Initial capital after colon" & vbTab & \_

Trim(Str(i)) & CR & "Lowercase after colon" \_

& vbTab & Trim(Str(j)) & CR2

' is/iz

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "ise>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "ise[sd]>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "ising>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "isation"

rng.Find.Execute Replace:=wdReplaceAll

l = ActiveDocument.Range.End - myTot

If l > 0 Then WordBasic.EditUndo

rng.Find.Text = "[armvt]ising"

rng.Find.Execute Replace:=wdReplaceAll

p = ActiveDocument.Range.End - myTot

If p > 0 Then WordBasic.EditUndo

rng.Find.Text = "[arvtw]ise"

rng.Find.Execute Replace:=wdReplaceAll

q = ActiveDocument.Range.End - myTot

If q > 0 Then WordBasic.EditUndo

rng.Find.Text = "ex[eo]rcis[ei]"

rng.Find.Execute Replace:=wdReplaceAll

r = ActiveDocument.Range.End - myTot

If r > 0 Then WordBasic.EditUndo

myRslt = myRslt & "-is- (very appx)" & vbTab & \_

Trim(Str(i + g + k + l - p - q - r)) & CR

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "ize>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "ize[sd]>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "izing>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "ization"

rng.Find.Execute Replace:=wdReplaceAll

l = ActiveDocument.Range.End - myTot

If l > 0 Then WordBasic.EditUndo

rng.Find.Text = "[Pp]riz[ie]"

rng.Find.Execute Replace:=wdReplaceAll

p = ActiveDocument.Range.End - myTot

If p > 0 Then WordBasic.EditUndo

rng.Find.Text = "[Sse]@iz[ie]"

rng.Find.Execute Replace:=wdReplaceAll

q = ActiveDocument.Range.End - myTot

If q > 0 Then WordBasic.EditUndo

myRslt = myRslt & "-iz- (very appx)" & vbTab \_

& Trim(Str(i + g + k + l - p - q)) & CR & \_

"(For a more accurate count, please use IZIScount.)" \_

& CR2

' data singular/plural

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "<data is>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<data has>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<data was>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[Tt]his data>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

myRslt = myRslt

l = i + g + h + k

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

' If useVoice = True Then speech.Speak cc, SVSFPurgeBeforeSpeak

rng.Find.Text = "<data are>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<data have>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<data were>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[Tt]hese data>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

If l + i + h + g + k > 0 Then myRslt = myRslt & \_

"data singular" & \_

vbTab & Trim(Str(l)) & CR & "data plural" & \_

vbTab & Trim(Str(i + g + h + k)) & CR2

' Is "first order" hyphenated?

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[snrt][tdh] order"

rng.Find.Execute Replace:=wdReplaceAll

dfgsdfg = ActiveDocument.Range.End

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[snrt][tdh]-order"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

If i + j > 0 Then myRslt = myRslt & \_

"'xxx order' with space" & vbTab & \_

Trim(Str(i)) & CR & "'xxx-order' with hyphen" \_

& vbTab & Trim(Str(j)) & CR2

' Is "three dimensional" hyphenated?

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[oweunvt ][eonrN1-4i] dimensional"

rng.Find.Execute Replace:=wdReplaceAll

dfgsdfg = ActiveDocument.Range.End

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[oweunvt ][eonrN1-4i]-dimensional"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

If i + j > 0 Then myRslt = myRslt & \_

"'xxx dimensional' with space" & vbTab & \_

Trim(Str(i)) & CR & "'xxx-dimensional' with hyphen" \_

& vbTab & Trim(Str(j)) & CR2

' Types of ellipsis

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = ChrW(8230)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "..."

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = ". . ."

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

If i + j + k > 0 Then

myRslt = myRslt & "Types of ellipsis:" & CR & \_

"proper ellipsis" & vbTab & Trim(Str(i)) & CR \_

& "triple dots" & vbTab & Trim(Str(j)) & CR \_

& "spaced triple dots" & vbTab & Trim(Str(k)) & CR2

End If

' Ellipsis, etc spacing

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

allChars = "/" & ChrW(8211) \_

& ChrW(8212) & "-" & ChrW(8230)

myNames = "Solidus En dash Em dash Hyphen Ellipsis Triple dotsSpaced dots "

For myGo = 0 To 6

sol = Mid(allChars, myGo + 1, 1)

If myGo = 5 Then sol = "..."

If myGo = 6 Then sol = ". . ."

myName = Trim(Mid(myNames, (11 \* myGo) + 1, 11))

rng.Find.Text = sol

rng.Find.Execute Replace:=wdReplaceAll

t = ActiveDocument.Range.End - myTot

If t > 0 Then

WordBasic.EditUndo

rng.Find.Text = " " & sol & " "

rng.Find.Execute Replace:=wdReplaceAll

bth = ActiveDocument.Range.End - myTot

If bth > 0 Then WordBasic.EditUndo

rng.Find.Text = "[! ]" & sol & " "

rng.Find.MatchWildcards = True

rng.Find.Execute Replace:=wdReplaceAll

ftr = ActiveDocument.Range.End - myTot

If ftr > 0 Then WordBasic.EditUndo

rng.Find.Text = " " & sol & "[! ]"

rng.Find.Execute Replace:=wdReplaceAll

bfr = ActiveDocument.Range.End - myTot

If bfr > 0 Then WordBasic.EditUndo

rng.Find.Text = "[! ]" & sol & "[! ]"

rng.Find.Execute Replace:=wdReplaceAll

nthr = ActiveDocument.Range.End - myTot

If nthr > 0 Then WordBasic.EditUndo

myRslt = myRslt & myName & " spacing:" & CR & "space before only" \_

& vbTab & Trim(Str(bfr)) & CR & "space after only" & \_

vbTab & Trim(Str(ftr)) & CR & "spaced both ends" & \_

vbTab & Trim(Str(bth)) & CR

If myGo <> 3 Then

myRslt = myRslt & "not spaced" & vbTab & Trim(Str(nthr)) & CR2

Else

myRslt = myRslt & CR

End If

myRslt = myRslt & CR

End If

Next myGo

' line breaks

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "^l"

rng.Find.MatchWildcards = False

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

' page breaks

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "^m"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

myRslt = myRslt & "line breaks" & vbTab & Trim(Str(i)) \_

& CR & "page breaks" & vbTab & Trim(Str(g)) & CR2

' fig/figure

aBit = ""

rng.Find.Text = "<fig>[!.]"

rng.Find.MatchWildcards = True

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "fig" & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<Fig>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "Fig" & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<fig."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "fig." & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<Fig."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "Fig." & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<figs>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "figs" & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<Figs>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "Figs" & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<figs."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "figs." & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "figure [0-9\(]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "figure" & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "[!.] Figure [0-9\(]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "Figure" & vbTab & Trim(Str(i)) & CR

End If

If aBit > "" Then myRslt = myRslt & aBit & CR

' Chapter/chapter

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[!.] Chapter [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "chapter [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then

myRslt = myRslt & "Chapter (number)" & vbTab & Trim(Str(i)) & CR \_

& "chapter (number)" & vbTab & Trim(Str(g)) & CR2

End If

' Section/section

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[!.] Section [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "section [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then

myRslt = myRslt & "Section (number)" & vbTab & \_

Trim(Str(i)) & CR & "section (number)" \_

& vbTab & Trim(Str(g)) & CR2

End If

' No./no.

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

' If useVoice = True Then speech.Speak cc, SVSFPurgeBeforeSpeak

rng.Find.Text = " No. [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = " No [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = " no. [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = " No.[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = " No[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

l = ActiveDocument.Range.End - myTot

If l > 0 Then WordBasic.EditUndo

rng.Find.Text = " no.[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

m = ActiveDocument.Range.End - myTot

If m > 0 Then WordBasic.EditUndo

If i + j + g + k + l + m > 0 Then

myRslt = myRslt & "No (number)" & vbTab & Trim(Str(i)) \_

& CR & "No. (number)" & vbTab & Trim(Str(j)) & CR \_

& "no. (number)" & vbTab & Trim(Str(g)) & CR

myRslt = myRslt & "No(number)" & vbTab & Trim(Str(k)) \_

& CR & "No.(number)" & vbTab & Trim(Str(l)) & CR \_

& "no.(number)" & vbTab & Trim(Str(m)) & CR2

End If

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = " Vol. [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = " Vol [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = " vol. [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = " Vol.[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = " Vol[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

l = ActiveDocument.Range.End - myTot

If l > 0 Then WordBasic.EditUndo

rng.Find.Text = " vol.[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

m = ActiveDocument.Range.End - myTot

If m > 0 Then WordBasic.EditUndo

If i + j + g + k + l + m > 0 Then

myRslt = myRslt & "Vol (number)" & vbTab & Trim(Str(i)) \_

& CR & "Vol. (number)" & vbTab & Trim(Str(j)) & CR \_

& "vol. (number)" & vbTab & Trim(Str(g)) & CR

myRslt = myRslt & "Vol(number)" & vbTab & Trim(Str(k)) \_

& CR & "Vol.(number)" & vbTab & Trim(Str(l)) & CR \_

& "vol.(number)" & vbTab & Trim(Str(m)) & CR

myRslt = myRslt & CR

End If

' equations

fText = ",<eq [0-9],<eq. [0-9],<eqn [0-9],<Eqn [0-9]," & \_

"eqns [0-9],eqs [0-9],<eq \(,<eq. \(,<Eq. \(" & \_

",<eqn \(,<Eqn \(,eqns \(,eqs \(,Eqs \(,Eqs. \(" & \_

",equation \(,[!.] Equation \(,equations \(" & \_

",[!.] Equations \(,equation [0-9]" & \_

",[!.] Equation [0-9],equations [0-9]" & \_

",[!.] Equations [0-9]"

resText = ",eq,eq.,eqn,Eqn,eqns,eqs,eq (n.n)," & \_

"eq. (n.n),Eq. (n.n),eqn (n.n)," & \_

"Eqn (n.n),eqns (n.n),eqs,Eqs,Eqs.," & \_

"equation (n.n),Equation (n.n)," & \_

"equations (n.n),Equations (n.n),equation," & \_

"Equation,equations,Equations"

myF = Split(fText, ",")

myRes = Split(resText, ",")

aBit = ""

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

myTot = ActiveDocument.Range.End

rng.Find.MatchWildcards = True

For q = 1 To 23

With rng.Find

.Text = myF(q)

.Replacement.Text = "^&!"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & myRes(q) & vbTab & Trim(Str(i)) & CR

Next q

If aBit > "" Then myRslt = myRslt & aBit & CR

' units

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[0-9][^32^160][kKcmM][NgAVm]>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9][^32^160][NgAVm]>"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9][kKcmM][NgAVm]>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9][NgAVm]>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

If i + j + g + h > 0 Then

myRslt = myRslt & "spaced units (3 mm)" & vbTab & \_

Trim(Str(i + j)) & CR & "unspaced units (3mm)" \_

& vbTab & Trim(Str(g + h)) & CR2

End If

' Ok, OK, ok, okay

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

myTot = ActiveDocument.Range.End

rng.Find.Text = "<OK>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<Ok>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<ok>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "<okay>"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

If i + h + g + j > 0 Then myRslt = myRslt & "OK" & \_

vbTab & Trim(Str(i)) & CR \_

& "Ok" & vbTab & Trim(Str(g)) & CR \_

& "ok" & vbTab & Trim(Str(h)) & CR \_

& "okay" & vbTab & Trim(Str(j)) & CR2

' Now go to all lowercase

rng.Case = wdLowerCase

myTot = ActiveDocument.Range.End

' Backward(s), forward(s) etc.

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

myTot = ActiveDocument.Range.End

rng.Find.Text = "[abcdfiknoprtuw]{2,4}ward>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[abcdfiknoprtuw]{2,4}wards>"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

If i + j > 0 Then myRslt = myRslt & "back/for/toward etc." & \_

vbTab & Trim(Str(i)) & CR \_

& "back/for/towards etc." & vbTab & Trim(Str(j)) & CR2

' amid(st), among(st), while(st)

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "<amid>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<among>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo: g = g + h

rng.Find.Text = "<while>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo: g = g + h

rng.Find.Text = "<amidst>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<amongst>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo: i = i + h

rng.Find.Text = "<whilst>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo: i = i + h

If i + g > 0 Then

myRslt = myRslt & "amid/among/while" & vbTab & Trim(Str(g)) & CR

myRslt = myRslt & "amidst/amongst/whilst" & vbTab & Trim(Str(i)) & CR2

End If

' past participle -rnt -elt

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "sp[oi]@lt>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "lea[np]t>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "[l ][be][ua]rnt>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[ds][wpm]elt>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "sp[oi]@[l]@ed>"

rng.Find.Execute Replace:=wdReplaceAll

p = ActiveDocument.Range.End - myTot

If p > 0 Then WordBasic.EditUndo

rng.Find.Text = "lea[np]ed>"

rng.Find.Execute Replace:=wdReplaceAll

q = ActiveDocument.Range.End - myTot

If q > 0 Then WordBasic.EditUndo

rng.Find.Text = "[l ][be][ua]rned>"

rng.Find.Execute Replace:=wdReplaceAll

r = ActiveDocument.Range.End - myTot

If r > 0 Then WordBasic.EditUndo

rng.Find.Text = "[ds][wpm]elled>"

rng.Find.Execute Replace:=wdReplaceAll

s = ActiveDocument.Range.End - myTot

If g + h + i + k + p + q + r + s > 0 Then myRslt = myRslt & \_

"-rnt -elt" & vbTab & Trim(Str(g + h + i + k)) & CR & \_

"-rned -elled" & vbTab & Trim(Str(p + q + r + s)) & CR2

' percentages

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

myTot = ActiveDocument.Range.End

rng.Find.Text = "[0-9]%"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9][^32^160]%"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9] per cent>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9] percent>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-z]{3,} per cent>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-z]{3,} percent>"

rng.Find.Execute Replace:=wdReplaceAll

m = ActiveDocument.Range.End - myTot

If m > 0 Then WordBasic.EditUndo

If i + j + g + h + k + m > 0 Then

myRslt = myRslt & "unspaced, e.g. 9%" & vbTab & \_

Trim(Str(i)) & CR & "spaced, e.g. 9 %" \_

& vbTab & Trim(Str(j)) & CR & "9 per cent" & vbTab & \_

Trim(Str(g)) & CR & "9 percent" \_

& vbTab & Trim(Str(h)) & CR & "nine per cent" & vbTab & \_

Trim(Str(k)) & CR & "nine percent" \_

& vbTab & Trim(Str(m)) & CR2

End If

' Feet and inches

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

myTot = ActiveDocument.Range.End

curlyOpt = Options.AutoFormatAsYouTypeReplaceQuotes

Options.AutoFormatAsYouTypeReplaceQuotes = False

rng.Find.Text = "[0-9]'"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9]"""

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9]" & ChrW(8242)

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9]" & ChrW(8243)

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

Options.AutoFormatAsYouTypeReplaceQuotes = curlyOpt

If i + j + g + h > 0 Then

myRslt = myRslt & "feet (straight) 9'" & vbTab & \_

Trim(Str(i)) & CR & "inches (straight) 9""" \_

& vbTab & Trim(Str(j)) & CR & "single prime 9" & \_

ChrW(8242) & vbTab & Trim(Str(g)) & CR & \_

"double prime 9" & ChrW(8243) & vbTab & \_

Trim(Str(h)) & CR2

End If

' focus(s)

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "focus[ei]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "focuss[ei]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "focus..." & \_

vbTab & Trim(Str(i)) & CR \_

& "focuss..." & vbTab & Trim(Str(g)) & CR2

' benefit(t)

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

myTot = ActiveDocument.Range.End

rng.Find.Text = "benefit[ei]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "benefitt[ei]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "benefit..." & \_

vbTab & Trim(Str(i)) & CR \_

& "benefitt..." & vbTab & Trim(Str(g)) & CR2

' co(-)oper...

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

myTot = ActiveDocument.Range.End

rng.Find.Text = "co-op[ei]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "coop[ei]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "co-oper..." & \_

vbTab & Trim(Str(i)) & CR \_

& "cooper..." & vbTab & Trim(Str(g)) & CR2

' Co-ordin

rng.Find.Text = "co-ord[ei]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "coord[ei]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "co-ord..." & \_

vbTab & Trim(Str(i)) & CR \_

& "coord..." & vbTab & Trim(Str(g)) & CR2

' Can't, cannot can not

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

myTot = ActiveDocument.Range.End

rng.Find.Text = "can[!a-z]t>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "cannot"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.Range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "can not"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

If i + h + g > 0 Then myRslt = myRslt & "can't" & \_

vbTab & Trim(Str(i)) & CR \_

& "cannot" & vbTab & Trim(Str(g)) & CR \_

& "can not" & vbTab & Trim(Str(h)) & CR2

' Wasn't, isn't, hasn't

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

myTot = ActiveDocument.Range.End

rng.Find.Text = "[owh ][aie]sn[!a-z]t>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[owh ][aie]s not"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

If i + h > 0 Then myRslt = myRslt & \_

"wasn't, isn't, hasn't" \_

& vbTab & Trim(Str(i)) & CR \_

& "was not, is not, has not" & vbTab & \_

Trim(Str(h)) & CR2

' Don't, won't

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "[dw]on[!a-z]t>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[dw][oil]{1,3} not"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

If i + h > 0 Then myRslt = myRslt & \_

"don't, won't" \_

& vbTab & Trim(Str(i)) & CR \_

& "do not, will not" & vbTab & \_

Trim(Str(h)) & CR2

' which/that

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = "which"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "that"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUndo

If i + h > 0 Then myRslt = myRslt & \_

"which" \_

& vbTab & Trim(Str(i)) & CR \_

& "that" & vbTab & \_

Trim(Str(h)) & CR2

' Funny characters

cc = cc - 1

DoEvents

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

rng.Find.Text = ChrW(178)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

myRslt = myRslt & "funny 'squared' character" \_

& vbTab & Trim(Str(i)) & CR2

myDiacritics = ""

For i = 192 To 255

If i <> 215 And i <> 247 Then

myDiacritics = myDiacritics & ChrW(i)

End If

Next i

rng.Find.Text = "[" & myDiacritics & "]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

myRslt = myRslt & "diacritics" & vbTab & Trim(Str(i)) & CR2

rng.Find.Text = "[" & ChrW(191) & ChrW(161) & ChrW(139) & \_

ChrW(155) & ChrW(171) & ChrW(187) & "]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo: myRslt = myRslt & \_

"Continental punctuation" & vbTab & Trim(Str(i)) & CR2

' Ordinary degree symbols

rng.Find.Text = ChrW(176)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = " " & ChrW(176)

rng.Find.Execute Replace:=wdReplaceAll

isp = ActiveDocument.Range.End - myTot

If isp > 0 Then WordBasic.EditUndo

If i > 0 Then myRslt = myRslt & "degree symbols closed" \_

& vbTab & Trim(Str(i - isp)) & CR \_

& "degree symbols spaced" \_

& vbTab & Trim(Str(isp)) & CR2

' Funny degrees

rng.Find.Text = ChrW(186)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = " " & ChrW(186)

rng.Find.Execute Replace:=wdReplaceAll

isp = ActiveDocument.Range.End - myTot

If isp > 0 Then WordBasic.EditUndo

If i + funnyDegrees > 0 Then

myRslt = myRslt & "funny degrees (0186) closed" \_

& vbTab & Trim(Str(i + funnyDegrees - isp - \_

funnyDegreesSp)) & CR \_

& "funny degrees (0186) spaced" \_

& vbTab & Trim(Str(isp + funnyDegreesSp)) & CR2

End If

appx = ""

If colouredText > 0 Then

If colourOverflow = True Then appx = " (I think)"

myRslt = myRslt & "text in coloured font" \_

& appx & vbTab & Trim(Str(colouredText - 1)) & CR2

End If

If lineBreaks > 0 Then

myRslt = myRslt & "line breaks" \_

& vbTab & Trim(Str(i + lineBreaks)) & CR2

End If

If pageBreaks > 0 Then

myRslt = myRslt & "page breaks" \_

& vbTab & Trim(Str(i + pageBreaks)) & CR2

End If

myRslt = myRslt & CR

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="DocAlyse" & vbCr & vbCr

newDoc.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading1)

newDoc.Paragraphs(3).Range.Select

Selection.End = newDoc.Content.End

Selection.TypeText CR & myRslt & CR2

Selection.Font.Bold = True

Set rng = ActiveDocument.Content

rng.ParagraphFormat.TabStops.ClearAll

rng.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(4.5), \_

Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces

' Grey out the zero lines

cc = cc - 1

DoEvents

Set rng = ActiveDocument.Content

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13([!^13]@)^t0"

.Wrap = wdFindContinue

.Replacement.Text = "^p\1^t^="

.Replacement.Font.Bold = False

.Replacement.Font.Color = wdColorGray25

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Text = "^t^=zczc"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With Selection.Find

.Text = ""

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

Selection.HomeKey Unit:=wdStory

tempDoc.Close SaveChanges:=False

If doingSeveralMacros = False Then

Beep

newDoc.Activate

Else

FUT.Activate

End If

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub DocAlyseForMac()**

' Paul Beverley - Version 28.09.19

' Analyses various aspects of a document

showWild = True

' prompts to count number of tests

' cc =50

For i = 1 To 15

spcs = " " & spcs

Next i

myResponse = MsgBox("Analyse this document?" \_

& vbCr & vbCr & "NOTE: Nothing will appear " \_

& "to happen, but please wait patiently!", vbQuestion \_

+ vbYesNoCancel, "DocAlyse For Mac")

If myResponse <> vbYes Then Exit Sub

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

' Use main file for italic 'et al' count...

myTot = ActiveDocument.range.End

Set rng = ActiveDocument.Content

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<et al>"

.Font.Italic = True

.Replacement.Text = "^&!"

.Wrap = wdFindContinue

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

italEtAls = ActiveDocument.range.End - myTot

If italEtAls > 0 Then WordBasic.EditUndo

' ...and superscript degree count

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[oO0]"

.Font.Superscript = True

.Replacement.Text = "vbvb"

.Replacement.Font.Superscript = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

funnyDegrees = (ActiveDocument.range.End - myTot) / 3

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " vbvb"

.Replacement.Text = "^&!"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

funnyDegreesSp = ActiveDocument.range.End - myTot - funnyDegrees \* 3

If funnyDegreesSp > 0 Then WordBasic.EditUndo

If funnyDegrees > 0 Then WordBasic.EditUndo

' StatusBar = spcs & myPrompt

DoEvents

Selection.HomeKey Unit:=wdStory

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

myEnd = rng.End

Set rng2 = ActiveDocument.Content

rng.Collapse wdCollapseEnd

rng.Text = rng2.Text

Set rng3 = ActiveDocument.Content

rng3.End = myEnd - 1

rng3.Select

Selection.Delete

myRslt = ""

Set rng = ActiveDocument.Content

myTot = ActiveDocument.range.End

CR = vbCr: CR2 = CR & CR

tr = Chr(9) & "0zczc" & CR: sp = " "

Selection.HomeKey Unit:=wdStory

' Ten or 10

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<ten>"

.Replacement.Text = "^&!"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = " <10>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "ten" & vbTab & \_

Trim(Str(i)) & CR & "10" & vbTab & Trim(Str(g)) & CR2

' spelt-out lower-case numbers over nine

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "<[efnst][efghinorvwx]{2,4}ty"

rng.Find.Execute Replace:=wdReplaceAll

a = ActiveDocument.range.End - myTot

If a > 0 Then WordBasic.EditUndo

rng.Find.Text = "<ten>"

rng.Find.Execute Replace:=wdReplaceAll

b = ActiveDocument.range.End - myTot

If b > 0 Then WordBasic.EditUndo

rng.Find.Text = "<eleven>"

rng.Find.Execute Replace:=wdReplaceAll

C = ActiveDocument.range.End - myTot

If C > 0 Then WordBasic.EditUndo

rng.Find.Text = "<twelve>"

rng.Find.Execute Replace:=wdReplaceAll

d = ActiveDocument.range.End - myTot

If d > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[efnst][efghinuorvwx]{2,4}teen>"

rng.Find.Execute Replace:=wdReplaceAll

E = ActiveDocument.range.End - myTot

If E > 0 Then WordBasic.EditUndo

rng.Find.Text = "<hundred>"

rng.Find.Execute Replace:=wdReplaceAll

f = ActiveDocument.range.End - myTot

If f > 0 Then WordBasic.EditUndo

If a + b + C + d + E + f > 0 Then myRslt = myRslt & \_

"spelt-out numbers (11-999)" & vbTab & \_

Trim(Str(a + b + C + d + E + f)) & CR2

' Four-digit numbers

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[!.]<[0-9]{4}>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

' take off 20xx dates

rng.Find.Text = "[!.]<20[0-9]{2}>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

' take off 13xx to 19xx dates

rng.Find.Text = "[!.]<1[3-9][0-9]{2}>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

i = i - g - k

If i < 1 Then i = 1

' Four figs with comma

rng.Find.Text = "[!.]<[0-9],[0-9]{3}>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

' Four figs with hard or ordinary space

rng.Find.Text = "[!.]<[0-9][^0160^32][0-9]{3}>[!,]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

If i + g + k > 0 Then

myRslt = myRslt & "Four-digit numbers:" & CR \_

& "nnnn" & vbTab & Trim(Str(i)) & CR \_

& "n,nnn" & vbTab & Trim(Str(g)) & CR \_

& "n nnn" & vbTab & Trim(Str(k)) & CR

If showWild = True Then myRslt = myRslt & \_

"<[0-9]{4}>" & sp \_

& "<[0-9],[0-9]{3}>" & sp \_

& "<[0-9][^0160^32][0-9]{3}>" & tr

myRslt = myRslt & CR

End If

' Dates with 'mid' in front

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "mid [0-9]{4}"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "mid-[0-9]{4}"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "mid[0-9]{4}"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

If i + g + k > 0 Then

myRslt = myRslt & "mid 1900(s)" & vbTab \_

& Trim(Str(i)) & CR & "mid-1900(s)" & vbTab & \_

Trim(Str(g)) & CR & "mid1900(s)" & vbTab & \_

Trim(Str(k)) & CR

If showWild = True Then myRslt = myRslt & \_

" mid[0-9]{4}" & tr

myRslt = myRslt & CR

End If

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "mid [0-9]{2}[!0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "mid-[0-9]{2}[!0-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "mid[0-9]{2}[!0-9]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

If i + g + k > 0 Then

myRslt = myRslt & "mid 90(s)" & vbTab \_

& Trim(Str(i)) & CR & "mid-90(s)" & vbTab & \_

Trim(Str(g)) & CR & "mid90(s)" & vbTab & \_

Trim(Str(k)) & CR

If showWild = True Then myRslt = myRslt & \_

" mid[0-9]{2}[!0-9]" & tr

myRslt = myRslt & CR

End If

' Serial comma/not serial comma

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[a-zA-Z\-]@, [a-zA-Z\-]@, and "

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt & "serial comma" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "[a-zA-Z\-]@, [a-zA-Z\-]@ and "

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt & "no serial comma" & vbTab & Trim(Str(i)) & CR

If showWild = True Then myRslt = myRslt & \_

"[a-zA-Z\-]@, [a-zA-Z\-]@, and " & sp & \_

"[a-zA-Z\-]@, [a-zA-Z\-]@ and " & tr

myRslt = myRslt & CR

' hard spaces

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "^s"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

' hard hyphens

rng.Find.Text = "^~"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

myRslt = myRslt & "hard spaces" & vbTab & Trim(Str(i)) \_

& CR & "hard hyphens" & vbTab & Trim(Str(g)) & CR2

' Types of ellipsis

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = ChrW(8230)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "..."

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = ". . ."

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

If i + j + k > 0 Then

myRslt = myRslt & "Types of ellipsis:" & CR & \_

"proper ellipsis" & vbTab & Trim(Str(i)) & CR \_

& "triple dots" & vbTab & Trim(Str(j)) & CR \_

& "spaced triple dots" & vbTab & Trim(Str(k)) & CR2

End If

' Ellipsis, etc spacing

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

allChars = ChrW(8230) & "/" & ChrW(8211) \_

& ChrW(8212) & "-"

myNames = "Ellipsis Solidus En dash Em dash Hyphen "

For myGo = 0 To 4

sol = Mid(allChars, myGo + 1, 1)

myName = Trim(Mid(myNames, (9 \* myGo) + 1, 9))

rng.Find.Text = sol

rng.Find.Execute Replace:=wdReplaceAll

t = ActiveDocument.range.End - myTot

If t > 0 Then

WordBasic.EditUndo

rng.Find.Text = " " & sol & " "

rng.Find.Execute Replace:=wdReplaceAll

bth = ActiveDocument.range.End - myTot

If bth > 0 Then WordBasic.EditUndo

rng.Find.Text = "[! ]" & sol & " "

rng.Find.MatchWildcards = True

rng.Find.Execute Replace:=wdReplaceAll

ftr = ActiveDocument.range.End - myTot

If ftr > 0 Then WordBasic.EditUndo

rng.Find.Text = " " & sol & "[! ]"

rng.Find.Execute Replace:=wdReplaceAll

bfr = ActiveDocument.range.End - myTot

If bfr > 0 Then WordBasic.EditUndo

rng.Find.Text = "[! ]" & sol & "[! ]"

rng.Find.Execute Replace:=wdReplaceAll

nthr = ActiveDocument.range.End - myTot

If nthr > 0 Then WordBasic.EditUndo

myRslt = myRslt & myName & " spacing:" & CR & "space before only" \_

& vbTab & Trim(Str(bfr)) & CR & "space after only" & \_

vbTab & Trim(Str(ftr)) & CR & "spaced both ends" & \_

vbTab & Trim(Str(bth)) & CR

If myGo < 4 Then

myRslt = myRslt & "not spaced" & vbTab & Trim(Str(nthr)) & CR2

Else

myRslt = myRslt & CR

End If

myRslt = myRslt & CR

End If

Next myGo

' line breaks

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "^l"

rng.Find.MatchWildcards = False

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

' page breaks

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "^m"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

myRslt = myRslt & "line breaks" & vbTab & Trim(Str(i)) \_

& CR & "page breaks" & vbTab & Trim(Str(g)) & CR2

' Single/double quotes

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = ChrW(8216)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

singleCurl = i

myRslt = myRslt & "curly open single quote" & vbTab & \_

Trim(Str(i)) & CR

rng.Find.Text = ChrW(8220)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt & "curly open double quote" & vbTab & \_

Trim(Str(i)) & CR

rng.Find.Text = Chr(39)

rng.Find.MatchWildcards = True

rng.Find.MatchCase = True

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt & "straight single quote" & vbTab & \_

Trim(Str(i)) & CR

rng.Find.Text = Chr(34)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt & "straight double quote" & vbTab & \_

Trim(Str(i)) & CR2

' etc(.)

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "<etc[!.]"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "<etc."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<etc. [A-Z]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<etc.^13"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

If h + i + g + k > 0 Then myRslt = myRslt & "etc" & \_

vbTab & Trim(Str(h)) & CR & "etc." & vbTab & \_

Trim(Str(i - g - k)) & CR2

' et al(.)

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "<et al[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<et al."

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

If g + i + italEtAls > 0 Then myRslt = myRslt & "et al." \_

& vbTab & Trim(Str(g)) & CR & "et al (italic, total)" & \_

vbTab & Trim(Str(italEtAls)) & CR & "et al (no dot)" & \_

vbTab & Trim(Str(i)) & CR2

' i.e./ie

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "i.e."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

myRslt = myRslt

rng.Find.Text = "<ie>"

rng.Find.MatchWildcards = True

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "ie" & vbTab & Trim(Str(g)) & CR \_

& "i.e." & vbTab & Trim(Str(i)) & CR2

' e.g./eg

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "e.g."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<eg>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "eg" & vbTab & Trim(Str(g)) & CR \_

& "e.g." & vbTab & Trim(Str(i)) & CR2

' Initials with surnames

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "<[A-Z]. [A-Z]. [A-Z][a-z]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[A-Z][a-z]{2,}, [A-Z]. [A-Z]. "

rng.Find.Execute Replace:=wdReplaceAll

i2 = ActiveDocument.range.End - myTot

If i2 > 0 Then WordBasic.EditUndo

aBit = "J. L. B. Matekoni" & vbTab & Trim(Str(i + i2)) & CR

g = i + i2

rng.Find.Text = "<[A-Z].[A-Z]. [A-Z][a-z]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[A-Z][a-z]{2,}, [A-Z].[A-Z]."

rng.Find.Execute Replace:=wdReplaceAll

i2 = ActiveDocument.range.End - myTot

If i2 > 0 Then WordBasic.EditUndo

aBit = aBit & "J.L.B. Matekoni" & vbTab & Trim(Str(i + i2)) & CR

g = g + i + i2

rng.Find.Text = "<[A-Z] [A-Z] [A-Z][a-z]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[A-Z][a-z]{2,}, [A-Z] [A-Z] "

rng.Find.Execute Replace:=wdReplaceAll

i2 = ActiveDocument.range.End - myTot

If i2 > 0 Then WordBasic.EditUndo

aBit = aBit & "J L B Matekoni" & vbTab & Trim(Str(i + i2)) & CR

g = g + i + i2

rng.Find.Text = "<[A-Z]{2}> [A-Z][a-z]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[A-Z][a-z]{2,}, [A-Z]{2}"

rng.Find.Execute Replace:=wdReplaceAll

i2 = ActiveDocument.range.End - myTot

If i2 > 0 Then WordBasic.EditUndo

aBit = aBit & "JLB Matekoni" & vbTab & Trim(Str(i + i2)) & \_

" (Beware! This can be inflated by, e.g. BBC Enterprises.)" & CR

If showWild = True Then aBit = aBit & "<[A-Z]. [A-Z]. [A-Z][a-z]" \_

& sp & "<[A-Z].[A-Z]. [A-Z][a-z]" & sp & "<[A-Z] [A-Z] [A-Z][a-z]" \_

& sp & "<[A-Z]{2}> [A-Z][a-z]" & tr

If showWild = True Then aBit = aBit & "<[A-Z][a-z]{2,}, [A-Z]. [A-Z]. " \_

& sp & "<[A-Z][a-z]{2,}, [A-Z].[A-Z]." & sp & "<[A-Z][a-z]{2,}, [A-Z] [A-Z] " \_

& sp & "<[A-Z][a-z]{2,}, [A-Z]{2}" & tr

aBit = aBit & CR

If g + i + i2 > 0 Then myRslt = myRslt & aBit

' Convention for page numbers

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "<p. [1-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<pp. [1-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

k = i + g

aBit = "p/pp. 123" & vbTab & Trim(Str(k)) & CR

rng.Find.Text = "<p.[1-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<pp.[1-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

aBit = aBit & "p/pp.123" & vbTab & Trim(Str(i + g)) & CR

k = k + i + g

rng.Find.Text = "<p [1-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<pp [1-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

aBit = aBit & "p/pp 123" & vbTab & Trim(Str(i + g)) & CR

k = k + i + g

rng.Find.Text = "<p[1-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<pp[1-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

aBit = aBit & "p/pp123" & vbTab & Trim(Str(i + g)) & CR

If showWild = True Then aBit = aBit & \_

" <pp[1-9]" & tr

aBit = aBit & CR

If k + i + g > 0 Then myRslt = myRslt & aBit

' Convention for ed/eds/edn

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "<ed>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<eds>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<edn>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "<ed."

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "<eds."

rng.Find.Execute Replace:=wdReplaceAll

m = ActiveDocument.range.End - myTot

If m > 0 Then WordBasic.EditUndo

rng.Find.Text = "<edn."

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.range.End - myTot

If j > 0 Then WordBasic.EditUndo

If k + m + j + i + g + h > 0 Then myRslt = myRslt \_

& "ed" & vbTab & Trim(Str(i)) & CR & "eds" \_

& vbTab & Trim(Str(g)) & CR & "edn" & vbTab & \_

Trim(Str(h)) & CR & "ed." \_

& vbTab & Trim(Str(k)) & CR & "eds." & vbTab & \_

Trim(Str(m)) & CR & "edn." \_

& vbTab & Trim(Str(j)) & CR2

' Convention for am/pm

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[1-9][ap]m"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

aBit = "2pm" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "[1-9][ap].m."

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

aBit = aBit & "2p.m." & vbTab & Trim(Str(g)) & CR

rng.Find.Text = "[1-9] [ap]m"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

aBit = aBit & "2 pm" & vbTab & Trim(Str(k)) & CR

rng.Find.Text = "[1-9] [ap].m."

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

aBit = aBit & "2 p.m." & vbTab & Trim(Str(h)) & CR2

If k + i + g + h > 0 Then myRslt = myRslt & aBit

' US/UK spelling

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[bpiv]our[ ,.s]"

rng.Find.Execute Replace:=wdReplaceAll

a = ActiveDocument.range.End - myTot

If a > 0 Then WordBasic.EditUndo

rng.Find.Text = "elling>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "elled>"

rng.Find.Execute Replace:=wdReplaceAll

f = ActiveDocument.range.End - myTot

If f > 0 Then WordBasic.EditUndo

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[bpiv]or[ ,.s]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "rior[ ,.s]"

rng.Find.Execute Replace:=wdReplaceAll

q = ActiveDocument.range.End - myTot

If q > 0 Then WordBasic.EditUndo

rng.Find.Text = "eling>"

rng.Find.Execute Replace:=wdReplaceAll

v = ActiveDocument.range.End - myTot

If v > 0 Then WordBasic.EditUndo

rng.Find.Text = "eled>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

If a + g + f + i + q + v + k > 0 Then myRslt = \_

myRslt & "UK spelling (appx)" & vbTab & \_

Trim(Str(a + g + f)) & CR & \_

"US spelling (appx)" & vbTab & \_

Trim(Str(i - q + v + k)) & CR & \_

"(For a more accurate count, please use UKUScount.)" & CR2

' US/UK punctuation

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[a-zA-Z][""" & ChrW(8221) & "][,.]"

rng.Find.Execute Replace:=wdReplaceAll

dfgsdfg = ActiveDocument.range.End

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-zA-Z]['" & ChrW(8217) & "][,.]"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-zA-Z][,.][""" & ChrW(8221) & "][,.]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-zA-Z]['" & ChrW(8217) & "][,.]"

rng.Find.Execute Replace:=wdReplaceAll

m = ActiveDocument.range.End - myTot

If m > 0 Then WordBasic.EditUndo

If i + j + k + m > 0 Then myRslt = myRslt & \_

"UK punctuation (appx)" & vbTab & \_

Trim(Str(i + j)) & CR & "US punctuation (appx)" \_

& vbTab & Trim(Str(k + m)) & CR2

' Initial capital after colon?

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[a-zA-Z]: [A-Z][a-z]"

rng.Find.Execute Replace:=wdReplaceAll

dfgsdfg = ActiveDocument.range.End

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-zA-Z]: [a-z]"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.range.End - myTot

If j > 0 Then WordBasic.EditUndo

If i + j > 0 Then myRslt = myRslt & \_

"Initial capital after colon" & vbTab & \_

Trim(Str(i)) & CR & "Lowercase after colon" \_

& vbTab & Trim(Str(j)) & CR2

' is/iz

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "ise>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "ise[sd]>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "ising>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "isation"

rng.Find.Execute Replace:=wdReplaceAll

l = ActiveDocument.range.End - myTot

If l > 0 Then WordBasic.EditUndo

rng.Find.Text = "[armvt]ising"

rng.Find.Execute Replace:=wdReplaceAll

p = ActiveDocument.range.End - myTot

If p > 0 Then WordBasic.EditUndo

rng.Find.Text = "[arvtw]ise"

rng.Find.Execute Replace:=wdReplaceAll

q = ActiveDocument.range.End - myTot

If q > 0 Then WordBasic.EditUndo

rng.Find.Text = "ex[eo]rcis[ei]"

rng.Find.Execute Replace:=wdReplaceAll

r = ActiveDocument.range.End - myTot

If r > 0 Then WordBasic.EditUndo

myRslt = myRslt & "-is- (very appx)" & vbTab & \_

Trim(Str(i + g + k + l - p - q - r)) & CR

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "ize>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "ize[sd]>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "izing>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "ization"

rng.Find.Execute Replace:=wdReplaceAll

l = ActiveDocument.range.End - myTot

If l > 0 Then WordBasic.EditUndo

rng.Find.Text = "[Pp]riz[ie]"

rng.Find.Execute Replace:=wdReplaceAll

p = ActiveDocument.range.End - myTot

If p > 0 Then WordBasic.EditUndo

rng.Find.Text = "[Sse]@iz[ie]"

rng.Find.Execute Replace:=wdReplaceAll

q = ActiveDocument.range.End - myTot

If q > 0 Then WordBasic.EditUndo

myRslt = myRslt & "-iz- (very appx)" & vbTab \_

& Trim(Str(i + g + k + l - p - q)) & CR & \_

"(For a more accurate count, please use IZIScount.)" \_

& CR2

' data singular/plural

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "<data is>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<data has>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<data was>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[Tt]his data>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

myRslt = myRslt

l = i + g + h + k

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "<data are>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<data have>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<data were>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[Tt]hese data>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

If l + i + h + g + k > 0 Then myRslt = myRslt & \_

"data singular" & \_

vbTab & Trim(Str(l)) & CR & "data plural" & \_

vbTab & Trim(Str(i + g + h + k)) & CR2

' past participle -rnt -elt

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "sp[oi]@lt>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "lea[np]t>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "[l ][be][ua]rnt>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[ds][wpm]elt>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "sp[oi]@[l]@ed>"

rng.Find.Execute Replace:=wdReplaceAll

p = ActiveDocument.range.End - myTot

If p > 0 Then WordBasic.EditUndo

rng.Find.Text = "lea[np]ed>"

rng.Find.Execute Replace:=wdReplaceAll

q = ActiveDocument.range.End - myTot

If q > 0 Then WordBasic.EditUndo

rng.Find.Text = "[l ][be][ua]rned>"

rng.Find.Execute Replace:=wdReplaceAll

r = ActiveDocument.range.End - myTot

If r > 0 Then WordBasic.EditUndo

rng.Find.Text = "[ds][wpm]elled>"

rng.Find.Execute Replace:=wdReplaceAll

s = ActiveDocument.range.End - myTot

If g + h + i + k + p + q + r + s > 0 Then myRslt = myRslt & \_

"-rnt -elt" & vbTab & Trim(Str(g + h + i + k)) & CR & \_

"-rned -elled" & vbTab & Trim(Str(p + q + r + s)) & CR2

' amid(st), among(st), while(st)

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "<[Aa]mid>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[Aa]mong>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo: g = g + h

rng.Find.Text = "<[Ww]hile>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo: g = g + h

rng.Find.Text = "<[Aa]midst>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[Aa]mongst>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo: i = i + h

rng.Find.Text = "<[Ww]hilst>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo: i = i + h

If i + g > 0 Then

myRslt = myRslt & "amid/among/while" & vbTab & Trim(Str(g)) & CR

myRslt = myRslt & "amidst/amongst/whilst" & vbTab & Trim(Str(i)) & CR

If showWild = True Then myRslt = myRslt & "[dgl]st>" & tr

myRslt = myRslt & CR

End If

' fig/figure

aBit = ""

rng.Find.Text = "<fig>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "fig" & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<Fig>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "Fig" & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<fig."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "fig." & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<Fig."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "Fig." & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<figs>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "figs" & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<Figs>[!.]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "Figs" & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "<figs."

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "figs." & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "figure [0-9\(]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "figure" & vbTab & Trim(Str(i)) & CR

End If

rng.Find.Text = "[!.] Figure [0-9\(]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then

WordBasic.EditUndo

aBit = aBit & "Figure" & vbTab & Trim(Str(i)) & CR

End If

If aBit > "" Then myRslt = myRslt & aBit & CR

' Chapter/chapter

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[!.] Chapter [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "chapter [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then

myRslt = myRslt & "Chapter (number)" & vbTab & Trim(Str(i)) & CR \_

& "chapter (number)" & vbTab & Trim(Str(g)) & CR

If showWild = True Then myRslt = myRslt & "chapter [0-9]" & tr

myRslt = myRslt & CR

End If

' Section/section

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[!.] Section [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "section [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then

myRslt = myRslt & "Section (number)" & vbTab & \_

Trim(Str(i)) & CR & "section (number)" \_

& vbTab & Trim(Str(g)) & CR

If showWild = True Then myRslt = myRslt & \_

"section [0-9]" & tr

myRslt = myRslt & CR

End If

' No./no.

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = " No. [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = " No [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = " no. [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = " No.[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = " No[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

l = ActiveDocument.range.End - myTot

If l > 0 Then WordBasic.EditUndo

rng.Find.Text = " no.[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

m = ActiveDocument.range.End - myTot

If m > 0 Then WordBasic.EditUndo

If i + j + g + k + l + m > 0 Then

myRslt = myRslt & "No (number)" & vbTab & Trim(Str(i)) \_

& CR & "No. (number)" & vbTab & Trim(Str(j)) & CR \_

& "no. (number)" & vbTab & Trim(Str(g)) & CR

myRslt = myRslt & "No(number)" & vbTab & Trim(Str(k)) \_

& CR & "No.(number)" & vbTab & Trim(Str(l)) & CR \_

& "no.(number)" & vbTab & Trim(Str(m)) & CR2

End If

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = " Vol. [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = " Vol [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = " vol. [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = " Vol.[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = " Vol[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

l = ActiveDocument.range.End - myTot

If l > 0 Then WordBasic.EditUndo

rng.Find.Text = " vol.[0-9]"

rng.Find.Execute Replace:=wdReplaceAll

m = ActiveDocument.range.End - myTot

If m > 0 Then WordBasic.EditUndo

If i + j + g + k + l + m > 0 Then

myRslt = myRslt & "Vol (number)" & vbTab & Trim(Str(i)) \_

& CR & "Vol. (number)" & vbTab & Trim(Str(j)) & CR \_

& "vol. (number)" & vbTab & Trim(Str(g)) & CR

myRslt = myRslt & "Vol(number)" & vbTab & Trim(Str(k)) \_

& CR & "Vol.(number)" & vbTab & Trim(Str(l)) & CR \_

& "vol.(number)" & vbTab & Trim(Str(m)) & CR

myRslt = myRslt & CR

End If

' equations

aBit = ""

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.MatchWildcards = True

rng.Find.Text = "<eq [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "eq" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "<eq. [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "eq." & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "<eqn [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "eqn" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "<Eqn [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "Eqn" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "eqns [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "eqns" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "eqs [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "eqs" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "<eq \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "eq (n.n)" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "<eq. \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "eq. (n.n)" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "<Eq. \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "Eq. (n.n)" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "<eqn \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "eqn (n.n)" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "<Eqn \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "Eqn (n.n)" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "eqns \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "eqns (n.n)" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "eqs \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "eqs" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "Eqs \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "Eqs" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "Eqs. \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "Eqs." & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "equation \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "equation (n.n)" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "[!.] Equation \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "Equation (n.n)" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "equations \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "equations (n.n)" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "[!.] Equations \("

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "Equations (n.n)" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "equation [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "equation" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "[!.] Equation [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "Equation" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "equations [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "equations" & vbTab & Trim(Str(i)) & CR

rng.Find.Text = "[!.] Equations [0-9]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

aBit = aBit & "Equations" & vbTab & Trim(Str(i)) & CR

If aBit > "" Then myRslt = myRslt & aBit & CR

' units

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[0-9][^32^160][kKcmM][NgAVm]>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9][^32^160][NgAVm]>"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9][kKcmM][NgAVm]>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9][NgAVm]>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

If i + j + g + h > 0 Then

myRslt = myRslt & "spaced units (3 mm)" & vbTab & \_

Trim(Str(i + j)) & CR & "unspaced units (3mm)" \_

& vbTab & Trim(Str(g + h)) & CR

If showWild = True Then myRslt = myRslt & \_

"[0-9][^32^160][kKcmM][NgAVm]>" & sp & \_

"[0-9][kKcmM][NgAVm]>" & sp & \_

"[0-9][NgAVm]>" & sp & \_

"[0-9][kKcmM][NgAVmg]>" & tr

myRslt = myRslt & CR

End If

' percentages

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[0-9]%"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9][^32^160]%"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9] per cent>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9] percent>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-z]{3,} per cent>"

rng.Find.Execute Replace:=wdReplaceAll

k = ActiveDocument.range.End - myTot

If k > 0 Then WordBasic.EditUndo

rng.Find.Text = "[a-z]{3,} percent>"

rng.Find.Execute Replace:=wdReplaceAll

m = ActiveDocument.range.End - myTot

If m > 0 Then WordBasic.EditUndo

If i + j + g + h + k + m > 0 Then

myRslt = myRslt & "unspaced, e.g. 9%" & vbTab & \_

Trim(Str(i)) & CR & "spaced, e.g. 9 %" \_

& vbTab & Trim(Str(j)) & CR & "9 per cent" & vbTab & \_

Trim(Str(g)) & CR & "9 percent" \_

& vbTab & Trim(Str(h)) & CR & "nine per cent" & vbTab & \_

Trim(Str(k)) & CR & "nine percent" \_

& vbTab & Trim(Str(m)) & CR2

End If

' Feet and inches

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

Options.AutoFormatAsYouTypeReplaceQuotes = False

rng.Find.Text = "[0-9]'"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9]"""

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.range.End - myTot

If j > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9]" & ChrW(8242)

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "[0-9]" & ChrW(8243)

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

If i + j + g + h > 0 Then

myRslt = myRslt & "feet (straight) 9'" & vbTab & \_

Trim(Str(i)) & CR & "inches (straight) 9""" \_

& vbTab & Trim(Str(j)) & CR & "single prime 9" & \_

ChrW(8242) & vbTab & Trim(Str(g)) & CR & \_

"double prime 9" & ChrW(8243) & vbTab & \_

Trim(Str(h)) & CR2

End If

' focus(s)

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[Ff]ocus[ei]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[Ff]ocuss[ei]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "focus..." & \_

vbTab & Trim(Str(i)) & CR \_

& "focuss..." & vbTab & Trim(Str(g)) & CR2

' benefit(t)

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[Bb]enefit[ei]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[Bb]enefitt[ei]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "benefit..." & \_

vbTab & Trim(Str(i)) & CR \_

& "benefitt..." & vbTab & Trim(Str(g)) & CR2

' co(-)oper...

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[Cc]o-op[ei]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[Cc]oop[ei]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "co-oper..." & \_

vbTab & Trim(Str(i)) & CR \_

& "cooper..." & vbTab & Trim(Str(g)) & CR2

' Co-ordin

rng.Find.Text = "[Cc]o-ord[ei]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[Cc]oord[ei]"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

If i + g > 0 Then myRslt = myRslt & "co-ord..." & \_

vbTab & Trim(Str(i)) & CR \_

& "coord..." & vbTab & Trim(Str(g)) & CR2

' Can't, cannot can not

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[Cc]an[!a-z]t>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[Cc]annot"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "[Cc]an not"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

If i + h + g > 0 Then myRslt = myRslt & "can't" & \_

vbTab & Trim(Str(i)) & CR \_

& "cannot" & vbTab & Trim(Str(g)) & CR \_

& "can not" & vbTab & Trim(Str(h)) & CR2

' Wasn't, isn't, hasn't etc

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "[owh ][aie]sn[!a-z]t>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "[owh ][aie]s not"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

If i + h > 0 Then myRslt = myRslt & \_

"wasn't, isn't, hasn't etc" \_

& vbTab & Trim(Str(i)) & CR \_

& "was not, is not, has not etc" & vbTab & \_

Trim(Str(h)) & CR2

' Ok, OK, ok, okay

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = "<OK>"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = "<Ok>"

rng.Find.Execute Replace:=wdReplaceAll

g = ActiveDocument.range.End - myTot

If g > 0 Then WordBasic.EditUndo

rng.Find.Text = "<ok>"

rng.Find.Execute Replace:=wdReplaceAll

h = ActiveDocument.range.End - myTot

If h > 0 Then WordBasic.EditUndo

rng.Find.Text = "<okay>"

rng.Find.Execute Replace:=wdReplaceAll

j = ActiveDocument.range.End - myTot

If j > 0 Then WordBasic.EditUndo

If i + h + g + j > 0 Then myRslt = myRslt & "OK" & \_

vbTab & Trim(Str(i)) & CR \_

& "Ok" & vbTab & Trim(Str(g)) & CR \_

& "ok" & vbTab & Trim(Str(h)) & CR \_

& "okay" & vbTab & Trim(Str(j)) & CR2

' Funny characters

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

rng.Find.Text = ChrW(178)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

myRslt = myRslt & "funny 'squared' character" \_

& vbTab & Trim(Str(i)) & CR2

rng.Find.Text = "[áÁàÀâÂäÄÃãÅåçÇéÉèÈêÊëËíÍìÌîÎ" & \_

"ñÑóÓòÒôÔöÖõÕøØßúÚùÙûÛüÜýÝÿŸ]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: \_

myRslt = myRslt & "diacritics" & vbTab & Trim(Str(i)) & CR2

rng.Find.Text = "[" & ChrW(161) & ChrW(191) & ChrW(8249) \_

& ChrW(8250) & ChrW(171) & ChrW(187) & "]"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo: myRslt = myRslt & \_

"Continental punctuation" & vbTab & Trim(Str(i)) & CR2

' Ordinary degree symbols

rng.Find.Text = ChrW(176)

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = " " & ChrW(176)

rng.Find.Execute Replace:=wdReplaceAll

isp = ActiveDocument.range.End - myTot

If isp > 0 Then WordBasic.EditUndo

If i > 0 Then myRslt = myRslt & "degree symbols closed" \_

& vbTab & Trim(Str(i - isp)) & CR \_

& "degree symbols spaced" \_

& vbTab & Trim(Str(isp)) & CR2

' Funny degrees

rng.Find.Text = "º"

rng.Find.Execute Replace:=wdReplaceAll

i = ActiveDocument.range.End - myTot

If i > 0 Then WordBasic.EditUndo

rng.Find.Text = " º"

rng.Find.Execute Replace:=wdReplaceAll

isp = ActiveDocument.range.End - myTot

If isp > 0 Then WordBasic.EditUndo

If i + funnyDegrees > 0 Then

myRslt = myRslt & "funny degrees closed" \_

& vbTab & Trim(Str(i + funnyDegrees - isp - \_

funnyDegreesSp)) & CR \_

& "funny degrees spaced" \_

& vbTab & Trim(Str(isp + funnyDegreesSp)) & CR2

End If

appx = ""

If colouredText > 0 Then

If colourOverflow = True Then appx = " (I think)"

myRslt = myRslt & "text in coloured font" \_

& appx & vbTab & Trim(Str(colouredText - 1)) & CR2

End If

If lineBreaks > 0 Then

myRslt = myRslt & "line breaks" \_

& vbTab & Trim(Str(i + lineBreaks)) & CR2

End If

If pageBreaks > 0 Then

myRslt = myRslt & "page breaks" \_

& vbTab & Trim(Str(i + pageBreaks)) & CR2

End If

' Medical bits go in here

myRslt = myRslt & CR

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:=CR & myRslt & CR2

Selection.Start = 0

Selection.Font.Bold = True

Selection.ParagraphFormat.TabStops(CentimetersToPoints(5#)).Position = \_

CentimetersToPoints(5#)

' Grey out the zero lines

' cc =cc - 1

DoEvents

' StatusBar = spcs & "Test number " & cc

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13([!^13]@)^t0"

.Wrap = wdFindContinue

.Replacement.Text = "^p\1^t^="

.Replacement.Font.Bold = False

.Replacement.Font.Color = wdColorGray25

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^t^=zczc"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Docalyse" & vbCr

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub MultiFileTextForMac()**

' Paul Beverley - Version 17.02.20

' Collects text plus simple formatting from multiple files

remindAboutOpening = True

listOff = True

acceptTCs = True

addFilename = True

convertMathsItems = True

On Error GoTo ReportIt

CR2 = vbCr & vbCr

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then navigate to the required folder

If remindAboutOpening = True Then myResponse = \_

MsgBox("Open the first file", , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

dirPath = Replace(ActiveDocument.FullName, \_

ActiveDocument.Name, "")

If Documents.Count > docCount Then ActiveDocument.Close

' Read the names of all the files in this directory

myFile = Dir(dirPath)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Collect unformatted text from ALL the files in" & \_

" directory:" & dirName & " ?", vbQuestion + vbYesNoCancel, \_

"Multifile Text Collection")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Collect unformatted text from the files listed here?", \_

vbQuestion + vbYesNoCancel, "Multifile Text Collection")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

Selection.HomeKey Unit:=wdStory

Set alltextDoc = Documents.Add

Set alltextRng = ActiveDocument.Content

Selection.TypeText "Loading first file" & vbCr & vbCr

Selection.HomeKey Unit:=wdStory

ActiveDocument.Paragraphs(1).range.Bold = True

ActiveDocument.Paragraphs(1).range.Font.Size = 40

gotLanguage = False

For i = 1 To numFiles

alltextDoc.Paragraphs(1).range = allMyFiles(i) & vbCr

' Get the folder name, and then the text for the files in the list

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set myDoc = Application.Documents.Open(FileName:=thisFile)

If gotLanguage = False Then

Selection.MoveEnd , 1

myLanguage = Selection.range.LanguageID

gotLanguage = True

End If

StatusBar = allMyFiles(i)

DoEvents

If acceptTCs = True Then myDoc.Revisions.AcceptAll

myDoc.TrackRevisions = False

If listOff = True Then myDoc.ConvertNumbersToText

If ActiveDocument.Endnotes.Count > 0 Then

Set thisDocRange = myDoc.Content

thisDocRange.Collapse wdCollapseEnd

thisDocRange.FormattedText = \_

myDoc.StoryRanges(wdEndnotesStory).FormattedText

End If

If ActiveDocument.Footnotes.Count > 0 Then

Set thisDocRange = myDoc.Content

thisDocRange.Collapse wdCollapseEnd

thisDocRange.FormattedText = \_

myDoc.StoryRanges(wdFootnotesStory).FormattedText

End If

' copy all the textboxes to the end of the text

shCount = myDoc.Shapes.Count

If shCount > 0 Then

Selection.EndKey Unit:=wdStory

Selection.TypeText vbCr & vbCr

For j = 1 To shCount

Set shp = ActiveDocument.Shapes(j)

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.hasText Then

Set rng = shp.TextFrame.TextRange

Selection.FormattedText = rng.FormattedText

Selection.EndKey Unit:=wdStory

End If

End If

Next

End If

Selection.HomeKey Unit:=wdStory

If addFilename = True Then Selection.TypeText CR2 & "[[[[ " & \_

allMyFiles(i) & " ]]]]]" & CR2

If convertMathsItems = True Then

Application.Visible = False

numMaths = ActiveDocument.OMaths.Count

If numMaths > 0 Then

For Each myMath In ActiveDocument.OMaths

myMath.range.Select

eqText = Replace(Selection.Text, vbCr, " ")

Selection.MoveStart , -1

Selection.Delete

startHere = Selection.Start

Selection.TypeText eqText

Selection.Start = startHere

Next myMath

End If

Application.Visible = True

End If

myDoc.Fields.Unlink

' Accept all track changes

myDoc.range.Revisions.AcceptAll

myDoc.TrackRevisions = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = wdFindContinue

.MatchWildcards = False

.MatchCase = True

.Replacement.Text = "zccz^&pqqp"

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Bold = True

.Replacement.Text = "bqqb^&zwvf"

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Superscript = True

.Replacement.Text = "yxzu^&qiwv"

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Subscript = True

.Replacement.Text = "xhwc^&yvxz"

.Execute Replace:=wdReplaceAll

End With

' Hide the screen to avoid mouse movement

Application.Visible = False

Set rng = ActiveDocument.Content

alltextRng.End = alltextDoc.range.End

alltextRng.Start = alltextDoc.range.End

alltextRng.Text = rng.Text

myDoc.Close SaveChanges:=wdDoNotSaveChanges

' Show the screen again

Application.Visible = True

Next i

alltextDoc.Paragraphs(1).range.Select

Selection.Delete

alltextDoc.Paragraphs(2).range.Delete

Set rng = alltextDoc.Content

rng.LanguageID = myLanguage

rng.NoProofing = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "yxzu(\*)qiwv"

.MatchWildcards = True

.MatchCase = False

.Replacement.Text = "^&"

.Replacement.Font.Superscript = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "xhwc(\*)yvxz"

.Replacement.Text = "^&"

.Replacement.Font.Subscript = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "bqqb(\*)zwvf"

.Replacement.Text = "^&"

.Replacement.Font.Bold = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zccz(\*)pqqp"

.Replacement.Text = "^&"

.Replacement.Font.Italic = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zccz"

.MatchWildcards = False

.MatchCase = False

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "yxzu"

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "bqqb"

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zwvf"

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "qiwv"

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "pqqp"

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "xhwc"

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "yvxz"

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

' Remove optional hyphens

.ClearFormatting

.Replacement.ClearFormatting

.Text = Chr(172)

.MatchWildcards = False

.MatchCase = True

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

totChars = rng.End

' Add one '[' for each file, to check they are all there

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[[[[ "

.Wrap = wdFindContinue

.MatchWildcards = False

.MatchWholeWord = False

.Replacement.Text = "[[[[[ "

.Execute Replace:=wdReplaceAll

End With

ActiveDocument.Paragraphs(3).range.Bold = True

Set rng = ActiveDocument.Content

totCharsNow = rng.End

Beep

MsgBox "Use FindFwd to check that all files have been included."

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[[[[["

.Wrap = wdFindContinue

.MatchWildcards = False

.MatchWholeWord = False

.Replacement.Text = ""

.Execute

End With

If totCharsNow - totChars <> numFiles Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

MsgBox "Warning: All files might not have been included."

End If

Exit Sub

ReportIt:

Application.Visible = True

On Error GoTo 0

Resume

End Sub

**Sub SerialCommaAlyse()**

' Paul Beverley - Version 25.01.22

' Highlights things that might be lists with serial commas (or not)

testRange = 50

myColourNS = wdYellow

myColourS = wdBrightGreen

oldColour = Options.DefaultHighlightColorIndex

CR = vbCr

CR2 = CR & CR

myRslt = "Rough indication:" & CR2

Application.ScreenUpdating = False

On Error GoTo ReportIt

st = Selection.Start

en = Selection.End

Set rng = ActiveDocument.Content

myTot = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Replacement.Text = "^&!"

.MatchWildcards = True

.Text = "[a-zA-Z\-]{1,}, [a-zA-Z\-]{1,}, and "

.Execute Replace:=wdReplaceAll

i = rng.End - myTot

If i > 0 Then WordBasic.EditUndo

.Text = "[a-zA-Z\-]{1,}, [a-zA-Z\-]{1,}, or "

.Execute Replace:=wdReplaceAll

j = rng.End - myTot

If j > 0 Then WordBasic.EditUndo

DoEvents

myRslt = myRslt & "Serial comma" & vbTab & Trim(Str(i + j)) & CR

.Text = "[a-zA-Z\-]{1,}, [a-zA-Z\-]{1,} and "

.Execute Replace:=wdReplaceAll

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUndo

.Text = "[a-zA-Z\-]{1,}, [a-zA-Z\-]{1,} or "

.Execute Replace:=wdReplaceAll

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUndo

End With

DoEvents

myRslt = myRslt & "No serial comma" & vbTab & Trim(Str(i + j)) & CR2

Beep

myResponse = MsgBox(myRslt & "Run full test?", vbQuestion + vbYesNo, "SerialCommaAlyse")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

If st = en Then

myResponse = MsgBox("Scan the whole document?!", vbQuestion + vbYesNo, "SerialCommaAlyse")

If myResponse <> vbYes Then Exit Sub

Else

rng.Start = st

rng.End = en

End If

Documents.Add

Selection.Text = rng.Text

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " and "

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

Application.ScreenUpdating = True

StatusBar = "Testing serial AND"

Application.ScreenUpdating = False

Do While rng.Find.Found = True

endNow = rng.End

startNow = rng.Start

rng.Collapse wdCollapseStart

rng.Start = rng.Start - 1

gotComma = (rng.Text = ",")

rng.Collapse wdCollapseStart

If rng.Start > testRange Then

rng.Start = rng.Start - testRange

Else

rng.Start = 0

End If

If InStr(rng, ",") > 0 And InStr(rng, ".") = 0 Then

rng.End = endNow

If gotComma = True Then

rng.HighlightColorIndex = myColourS

Else

rng.HighlightColorIndex = myColourNS

End If

rng.Start = startNow - 1

If Left(rng, 1) <> "," Then rng.Start = rng.Start + 1

rng.Font.Bold = True

End If

rng.Start = endNow

rng.Find.Execute

DoEvents

Loop

Application.ScreenUpdating = True

StatusBar = "Testing serial OR"

Application.ScreenUpdating = False

Beep

Set rng = ActiveDocument.Content

With rng.Find

.Text = " or "

.Replacement.Text = ""

.Execute

End With

Do While rng.Find.Found = True

myCount = myCount + 1

endNow = rng.End

startNow = rng.Start

rng.Collapse wdCollapseStart

rng.Start = rng.Start - 1

gotComma = (rng.Text = ",")

rng.Collapse wdCollapseStart

If rng.Start > testRange Then

rng.Start = rng.Start - testRange

Else

rng.Start = 0

End If

If InStr(rng, ",") > 0 And InStr(rng, ".") = 0 Then

rng.End = endNow

If gotComma = True Then

rng.HighlightColorIndex = myColourS

Else

rng.HighlightColorIndex = myColourNS

End If

rng.Start = startNow - 1

If Left(rng, 1) <> "," Then rng.Start = rng.Start + 1

rng.Font.Bold = True

End If

rng.Start = endNow

rng.Find.Execute

DoEvents

Loop

StatusBar = ""

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.Replacement.Font.Bold = True

.MatchWildcards = True

Options.DefaultHighlightColorIndex = myColourS

.Text = "[a-zA-Z\-]{1,}, [a-zA-Z\-]{1,}, and "

.Execute Replace:=wdReplaceAll

.Text = "[a-zA-Z\-]{1,}, [a-zA-Z\-]{1,}, or "

.Execute Replace:=wdReplaceAll

DoEvents

Options.DefaultHighlightColorIndex = myColourNS

.Text = "[a-zA-Z\-]{1,}, [a-zA-Z\-]{1,} and "

.Execute Replace:=wdReplaceAll

.Text = "[a-zA-Z\-]{1,}, [a-zA-Z\-]{1,} or "

.Execute Replace:=wdReplaceAll

DoEvents

End With

Application.ScreenUpdating = True

Options.DefaultHighlightColorIndex = oldColour

Beep

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub MegAlyse()**

' Paul Beverley - Version 20.01.22

' Launches a selected series of analysis macros

' Works with: AccentAlyse, CapitAlyse, CenturyAlyse, DocAlyse,

' FullNameAlyse, HyphenAlyse, ListAlyse, ProperNounAlyse,

' SpecialSortsLister, SpellAlyse, SpellingErrorListerBilingual,

' WordPairAlyse

myAlyses = "ListAlyse, DocAlyse"

myAlyses = "SpellAlyse, WordPairAlyse"

myAlyses = "DocAlyse, HyphenAlyse, ProperNounAlyse, SpellAlyse, WordPairAlyse"

saveResultsFiles = False

' On a Mac, you will need something like:

myFolder = "/Users/Paul/My Documents/Macro\_stuff/"

' On Windows, you will need something like:

myFolder = "C:\Documents and Settings\Paul\Macro\_stuff\"

myResponse = MsgBox("MegAlyse" & vbCr & vbCr & \_

"Run: " & myAlyses & "?", vbQuestion \_

+ vbYesNoCancel, "MegAlyse")

If myResponse <> vbYes Then Exit Sub

' Don't change this filename

tempFile = myFolder & "zzTestFile"

stTime = Time

thisLanguage = Selection.LanguageID

Set rng = ActiveDocument.Content

Documents.Add

Set testFile = ActiveDocument

Selection.FormattedText = rng.FormattedText

Selection.EndKey Unit:=wdStory

If ActiveDocument.Endnotes.Count > 0 Then

Set thisDocRange = testFile.Content

thisDocRange.Collapse wdCollapseEnd

thisDocRange.FormattedText = \_

testFile.StoryRanges(wdEndnotesStory).FormattedText

End If

If ActiveDocument.Footnotes.Count > 0 Then

Set thisDocRange = testFile.Content

thisDocRange.Collapse wdCollapseEnd

thisDocRange.FormattedText = \_

testFile.StoryRanges(wdFootnotesStory).FormattedText

End If

' copy all the textboxes to the end of the text

shCount = testFile.Shapes.Count

If shCount > 0 Then

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr & vbCr

For j = 1 To shCount

Set shp = ActiveDocument.Shapes(j)

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

Set rng = shp.TextFrame.TextRange

Selection.FormattedText = rng.FormattedText

Selection.EndKey Unit:=wdStory

End If

End If

Next

For j = shCount To 1 Step -1

ActiveDocument.Shapes(j).Delete

Next

End If

Set rng = ActiveDocument.Content

rng.Fields.Unlink

rng.Revisions.AcceptAll

For Each myPic In ActiveDocument.InlineShapes

myPic.Delete

Next myPic

ActiveDocument.Content.LanguageID = thisLanguage

ActiveDocument.SaveAs FileName:=tempFile

myAlyses = Replace("," & myAlyses & ",", ",,", ",")

myAlyses = Replace(myAlyses, " ", "")

thisArray = Split(myAlyses, ",")

For i = 1 To UBound(thisArray) - 1

rprt = thisArray(i) & " started about: " & Left(Time, 5) & vbCr

Debug.Print rprt

Application.Run MacroName:=thisArray(i)

DoEvents

Next i

rprt = vbCr & "Finished at: " & Left(Time, 5)

Debug.Print rprt

If saveResultsFiles Then

For Each myDoc In Documents

myName = myDoc.Name

If Left(myName, 8) = "Document" Then

Set rng = myDoc.Content

newName = Left(rng.Text, 40)

crPos = InStr(newName, vbCr)

If crPos > 3 Then

newName = Left(newName, crPos - 1)

myDoc.Activate

myFullFilename = myFolder & newName

ActiveDocument.SaveAs FileName:=myFullFilename

End If

End If

Next myDoc

End If

testFile.Activate

ActiveDocument.Close SaveChanges:=False

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

End Sub

**Sub** **CenturyAlyse()**

' Paul Beverley - Version 25.11.20

' Analyses how centuries are formatted in a document

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "-"

.Replacement.Text = " "

.Wrap = wdFindContinue

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[sth]{2}"

.Font.Superscript = True

.Replacement.Text = "zcthzc"

.Wrap = wdFindContinue

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totWas = ActiveDocument.Content.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "C[0-9]{2}>"

.Replacement.Text = "11"

.Replacement.Highlight = True

.Wrap = wdFindContinue

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n01 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "C[0-9]{2}[ths]{2}>"

.Replacement.Text = "2222"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n02 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "C[0-9]{2}zc[ths]{2}zc>"

.Replacement.Text = "33333333"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n03 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "[0-9]{2}zc[ths]{2}zc Ce"

.Replacement.Text = "6666666666"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n06 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "[0-9]{2}zc[ths]{2}zc ce"

.Replacement.Text = "7777777777"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n07 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "[XIV]{2}zc[ths]{2}zc Ce"

.Replacement.Text = "8888888888"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n08 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "[XIV]{2}zc[ths]{2}zc ce"

.Replacement.Text = "9999999999"

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n09 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "[XIV]{2}[ths]{2} Ce"

.Replacement.Text = "AAAAAA"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n10 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "[XIV]{2}[ths]{2} ce"

.Replacement.Text = "BBBBBB"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n11 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "[ienrlf]{2}[sth]{2} Ce"

.Replacement.Text = "CCCCCC"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n12 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "[ienrlf]{2}[sth]{2} ce"

.Replacement.Text = "DDDDDD"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n13 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "[ths]{2} Ce"

.Replacement.Text = "4444"

.Wrap = wdFindContinue

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n04 = totWas - totNow

totWas = totNow

With rng.Find

.Text = "[ths]{2} ce"

.Replacement.Text = "5555"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totNow = ActiveDocument.Content.End

n05 = totWas - totNow

myReport = "C19YYn01QQC19thYYn02QQC19^thYYn03QQNineteenth " & \_

"CenturyYYn12QQnineteenth centuryYYn13QQ19th CenturyYY" & \_

"n04QQ19th centuryYYn05QQ19^th CenturyYYn06QQ19^th " & \_

"centuryYYn07QQXIXth CenturyYYn10QQXIXth centuryYYn11QQ" & \_

"XIX^th CenturyYYn08QQXIX^th centuryYYn09"

myReport = Replace(myReport, "YY", vbTab)

myReport = Replace(myReport, "QQ", vbCr)

myReport = Replace(myReport, "n01", Str(n01))

myReport = Replace(myReport, "n02", Str(n02))

myReport = Replace(myReport, "n03", Str(n03))

myReport = Replace(myReport, "n04", Str(n04))

myReport = Replace(myReport, "n05", Str(n05))

myReport = Replace(myReport, "n06", Str(n06))

myReport = Replace(myReport, "n07", Str(n07))

myReport = Replace(myReport, "n08", Str(n08))

myReport = Replace(myReport, "n09", Str(n09))

myReport = Replace(myReport, "n10", Str(n10))

myReport = Replace(myReport, "n11", Str(n11))

myReport = Replace(myReport, "n12", Str(n12))

myReport = Replace(myReport, "n13", Str(n13))

Selection.WholeStory

Selection.Range.Style = ActiveDocument.Styles(wdStyleNormal)

Selection.Font.Reset

Selection.TypeText myReport

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^^th"

.Replacement.Text = "th"

.Replacement.Font.Superscript = True

.Wrap = wdFindContinue

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.WholeStory

Selection.MoveEnd , -1

Selection.ConvertToTable Separator:=wdSeparateByTabs

Selection.Tables(1).AutoFitBehavior (wdAutoFitContent)

Selection.Tables(1).Borders(wdBorderTop).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderLeft).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderBottom).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderRight).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderVertical).LineStyle = wdLineStyleNone

Selection.Collapse wdCollapseStart

If doingSeveralMacros = False Then

Beep

Else

FUT.Activate

End If

End Sub

**Sub DayDateAlyse()**

' Paul Beverley - Version 10.04.22

' Lists words out of a proper noun frequency list

myWds = " Mon Tue Tues Wed Weds Thu Thurs Fri Sat Sun X "

myWds = myWds & " Jan Feb Mar Apr May Jun Jul Aug Sep Sept Oct Nov Dec X "

myWds = myWds & " January February March April May June "

myWds = myWds & " July August September October November December "

' Main program

Set rng = ActiveDocument.Content

If InStr(rng.Text, " . . .") = 0 Then

Beep

myResponse = MsgBox("Please open the proper noun frequency list.", \_

vbOK, "DayDateAlyse")

Exit Sub

End If

myWds = Replace(myWds, " ", " ")

myWds = Trim(Replace(myWds, " ", " "))

myWds = Replace(myWds, " ", " .,")

wd = Split(myWds, ",")

For i = 0 To UBound(wd)

blah = wd(i)

With rng.Find

.Text = "^13" & wd(i) & "\*[0-9]{1,}"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

If .Found = True Then

wd(i) = Mid(rng.Text, 2)

Else

wd(i) = Replace(wd(i), " .", "")

End If

End With

DoEvents

Next i

myResult = ""

For i = 0 To UBound(wd)

If wd(i) = "X" Then wd(i) = ""

myResult = myResult & wd(i) & vbCr

Next i

Documents.Add

Selection.TypeText Text:=myResult

For Each pr In ActiveDocument.Paragraphs

txt = pr.Range.Text

txt = Left(Right(txt, 2), 1)

num = Val(txt)

If num = 0 And txt <> "0" Then pr.Range.Font.Color = wdColorGray25

DoEvents

Next pr

Selection.HomeKey Unit:=wdStory

Selection.TypeText "DayDate Analysis" & vbCr & vbCr

Set rng = ActiveDocument.Content.Paragraphs(1).Range

rng.Style = ActiveDocument.Styles(wdStyleHeading1)

Beep

End Sub

**Sub ListAlyse()**

' Paul Beverley - Version 11.11.20

' Makes a list of all the 'list' items - then you can analyse them!

Set orig = ActiveDocument.Content

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox(" ListAlyse" & vbCr & vbCr & \_

"Analyse this document?", vbQuestion \_

+ vbYesNoCancel, "ListAlyse")

If myResponse <> vbYes Then Exit Sub

End If

Documents.Add

' Selection.Text = orig.Text

Selection.FormattedText = orig.FormattedText

ActiveDocument.ConvertNumbersToText

Set rng = ActiveDocument.Content

Set wasRng = ActiveDocument.Paragraphs(1)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13{2,}"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

numTables = ActiveDocument.Tables.Count

If numTables > 0 Then

For i = numTables To 1 Step -1

ActiveDocument.Tables(i).Delete

Next i

End If

Set rng = ActiveDocument.Content

rng.Font.Underline = True

inList = False

For Each myPara In ActiveDocument.Paragraphs

isItem = False

' Bullet

If myPara.Range.Characters(1) = ChrW(8226) Then isItem = True

' Weird bullets from auto-lists

Debug.Print myPara.Range.Characters(1)

Debug.Print Asc(myPara.Range.Characters(1))

myPara.Range.Select

If Asc(myPara.Range.Characters(1)) = 63 Then isItem = True

' en dash

If myPara.Range.Characters(1) = ChrW(8211) Then isItem = True

' numbered line

If Val(myPara.Range.Words(1)) > 0 Then isItem = True

' a. or a)

If InStr("abcdefghijkl", myPara.Range.Words(1)) > 0 Then isItem = True

If isItem Then

myPara.Range.Font.Underline = False

wasRng.Font.Underline = False

inList = True

Else

If inList = True Then

wasRng.InsertAfter vbCr

wasRng.Start = wasRng.Start - 1

wasRng.Font.Underline = False

wasRng.Select

inList = False

End If

End If

DoEvents

Set wasRng = myPara.Range.Duplicate

Next myPara

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Selection.HomeKey Unit:=wdStory

If doingSeveralMacros = True Then

FUT.Activate

Else

Beep

End If

End Sub

**Sub AAnAlyse()**

' Paul Beverley - Version 16.01.21

' Checks a's and an's for agreement with the following word

OKwithA = ",europe,european,once,one,underline,unknown,unspaced,"

OKwithA = OKwithA & ",uniaxial,uniaxially,uniform,uniformly,"

OKwithA = OKwithA & ",unique,uniquely,unit,unitarian,united,"

OKwithA = OKwithA & ",university,union,united,universe,"

OKwithA = OKwithA & ",universal,universally,unilateral,unilaterally,"

OKwithA = OKwithA & ",uppercase,useful,usefully,useless,uselessly,user,"

OKwithA = OKwithA & ",usual,usually,,utility,utilities,utilitiarian,"

OKwithA = OKwithA & ",utilization,utilisation,"

OKwithAn = ",hour,hourly,honest,honestly,honor,honour,honorary,"

OKwithAn = OKwithAn & ",honorarium,honorific,"

strongColour = wdBrightGreen

mutedColour = wdGray25

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[anA]{1,2}>"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

OKwithA = "," & OKwithA & ","

OKwithAn = "," & OKwithAn & ","

qts = "'""" & ChrW(8216) & ChrW(8220)

Do While rng.Find.Found = True

endNow = rng.End

startArticle = rng.Start

article = LCase(rng)

rng.Start = endNow + 1

rng.End = endNow + 2

aOK = True

If Len(rng) > 0 Then

nextCharacter = Chr(Asc(rng))

Else

nextCharacter = ""

End If

rng.Expand wdWord

nextWord = rng

' Check for quotes before and after

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

If nextCharacter = "'" Then

rng.Start = rng.Start + 1

nextWord = rng

If Len(rng) > 0 Then

nextCharacter = Chr(Asc(rng))

Else

nextCharacter = ""

End If

End If

' Check for apostrophe-s

aposPosn = InStr(nextWord, ChrW(8217)) + InStr(nextWord, "'")

If aposPosn > 0 Then rng.End = rng.Start + aposPosn - 1

nextWord = rng

' Check for close quotes

If InStr(qts, nextWord) > 0 Then

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

nextWord = rng

If Len(rng) > 0 Then

nextCharacter = Chr(Asc(rng))

Else

nextCharacter = ""

End If

End If

endWord = rng.End

' Main check for agreement

If LCase(nextWord) <> UCase(nextWord) Then

If article = "a" Then

aOK = InStr("aAeEiIoO", nextCharacter) = 0

Else

aOK = InStr("aAeEiIoO", nextCharacter) > 0

End If

' Check single-letter words

If Len(nextWord) = 1 Then

If InStr("AaEeFfHhIiLlMmNnOoRrSsXx", nextCharacter) > 0 Then

aOK = (article = "an")

Else

aOK = (article = "a")

End If

End If

rng.Start = startArticle

rng.End = endWord

' Check words from lists above that are exceptions

testWord = "," & LCase(nextWord) & ","

If InStr(OKwithA, testWord) > 0 Then

If article = "a" Then aOK = True Else aOK = False

End If

If InStr(OKwithAn, testWord) > 0 Then

If article = "an" Then aOK = True Else aOK = False

End If

' Ignore people with the initial 'A.'

If InStr(rng, ".") > 0 Then

aOK = True

nextWord = "xxx"

End If

' Now highlight definite error

If aOK = False Then

rng.HighlightColorIndex = strongColour

rng.Select

End If

' Reduce highlight strength for acronyms

' that might not be wrong

If UCase(nextWord) = nextWord And Len(nextWord) > 1 Then

If InStr("FHLMRSX", nextCharacter) > 0 Then

If LCase(article) = "an" Then

rng.HighlightColorIndex = mutedColour

Else

rng.HighlightColorIndex = strongColour

End If

End If

If InStr("U", nextCharacter) > 0 Then

If LCase(article) = "an" Then

rng.HighlightColorIndex = strongColour

Else

rng.HighlightColorIndex = mutedColour

End If

End If

Selection.Collapse wdCollapseEnd

End If

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Beep

End Sub

**Sub FormatAlyse()**

' Paul Beverley - Version 07.09.19

' Highlights various formatting features

' Highlighting colour

strongHighlight = wdYellow

' strongHighlight = 0

boldHighlight = wdTurquoise

' boldHighlight = 0

italicHighlight = wdYellow

' italicHighlight = 0

BIHighlight = wdBrightGreen

' BIHighlight = 0

thinSpaceHighlight = wdGray50

' Font colouring

' strongColour = wdColorBlue

strongColour = 0

' boldColour = wdColorPink

boldColour = 0

' italicColour = wdColorRed

italicColour = 0

' BIColour = wdColorBrightGreen

BIColour = 0

myMarker = "||"

' Other characters NOT to highlight on Option 8

notThese = "\[\]\\*\@" & "\_+=&%"

nmlStyle = ActiveDocument.Styles(wdStyleNormal)

normalSize = ActiveDocument.Styles(nmlStyle).Font.Size

normalFont = ActiveDocument.Styles(nmlStyle).Font.Name

oldHighlight = Options.DefaultHighlightColorIndex

CR = vbCr

CR2 = CR & CR

menuText = "1 - Font size" & CR2 & \_

"2 - Font name" & CR2 & \_

"3 - Style" & CR2 & \_

"4 - Bold/italic" & CR2 & \_

"5 - Bold/italic " & ChrW(8211) & " not in a style" & CR2 & \_

"6 - Super/subscript" & CR2 & \_

"7 - Diacritics" & CR2 & \_

"8 - Various non-alpha characters (slow)" & CR2 & \_

"9 - Funny spaces" & CR2 & \_

"10 - Funny Symbol fonts" & CR

Do

myJob = Val(InputBox(menuText, "Format Highlighter"))

Loop Until myJob <= 10

If myJob = 0 Then Exit Sub

ActiveDocument.TrackRevisions = False

' Clear highlighting or colouration

Set rng = ActiveDocument.Content

If strongHighlight > 0 Then rng.HighlightColorIndex = wdNoHighlight

If strongColour > 0 Then rng.Font.Color = wdColorAutomatic

' Clear markers

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myMarker

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Font size

If myJob = 1 Then

Set rng = ActiveDocument.Content

If strongHighlight > 0 Then rng.HighlightColorIndex = strongHighlight

If strongColour > 0 Then rng.Font.Color = strongColour

With rng.Find

.Text = ""

.Font.Size = normalSize

.Replacement.Highlight = False

.Replacement.Font.Color = False

.Execute Replace:=wdReplaceAll

End With

End If

' Font name

If myJob = 2 Then

Set rng = ActiveDocument.Content

If strongHighlight > 0 Then rng.HighlightColorIndex = strongHighlight

If strongColour > 0 Then rng.Font.Color = strongColour

With rng.Find

.Text = ""

.Font.Name = normalFont

.Replacement.Highlight = False

.Replacement.Font.Color = False

.Execute Replace:=wdReplaceAll

End With

End If

' Style

If myJob = 3 Then

Set rng = ActiveDocument.Content

If strongHighlight > 0 Then rng.HighlightColorIndex = strongHighlight

If strongColour > 0 Then rng.Font.Color = strongColour

With rng.Find

.Text = ""

.Style = nmlStyle

.Replacement.Highlight = False

.Replacement.Font.Color = False

.Execute Replace:=wdReplaceAll

End With

End If

' bold/italic

If myJob = 4 Or myJob = 5 Then

Options.DefaultHighlightColorIndex = boldHighlight

With rng.Find

.Text = ""

.Font.Bold = True

If boldHighlight > 0 Then .Replacement.Highlight = True

If boldColour > 0 Then .Replacement.Font.Color = boldColour

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = italicHighlight

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

If italicHighlight > 0 Then .Replacement.Highlight = True

If italicColour > 0 Then .Replacement.Font.Color = italicColour

.Font.Italic = True

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = BIHighlight

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Font.Bold = True

If BIColour > 0 Then .Replacement.Font.Color = BIColour

If BIHighlight > 0 Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

' bold/italic - not in a style

If myJob = 5 Then

Set rng = ActiveDocument.Content

rng.Font.Shadow = True

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Style = nmlStyle

.Replacement.Font.Shadow = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Shadow = True

.Replacement.Font.Shadow = False

If boldHighlight > 0 Then .Replacement.Highlight = False

If boldColour > 0 Then .Replacement.Font.Color = wdColorAutomatic

.Execute Replace:=wdReplaceAll

End With

End If

End If

' Super/Subscript

If myJob = 6 Then

Options.DefaultHighlightColorIndex = strongHighlight

With rng.Find

.Text = ""

.Font.Superscript = True

If strongHighlight > 0 Then .Replacement.Highlight = True

.Replacement.Font.Color = strongColour

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Text = ""

.Font.Subscript = True

If strongHighlight > 0 Then .Replacement.Highlight = True

.Replacement.Font.Color = strongColour

.Execute Replace:=wdReplaceAll

End With

End If

' Diacritics

If myJob = 7 Then

Options.DefaultHighlightColorIndex = strongHighlight

With rng.Find

.Text = "[a-zA-Z ,.0-9\-/\(\);:]{1,}"

If strongHighlight > 0 Then .Replacement.Highlight = True

.Replacement.Font.Color = strongColour

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.Text = "[abcdefghijklmnopqrstuvwxyz ,.0-9" & \_

"\-/\(\);:ABCDEFGHIJKLMNOPQRSTUVWXYZ]{1,}"

.Replacement.Highlight = False

.Replacement.Font.Color = wdColorAutomatic

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

' Various non-alpha characters

If myJob = 8 Then

Set rng = ActiveDocument.Content

If strongHighlight > 0 Then rng.HighlightColorIndex = strongHighlight

Options.DefaultHighlightColorIndex = strongHighlight

With rng.Find

.Text = "[a-zA-Z ,.0-9^13^t" & ChrW(8211) & ChrW(8216) \_

& ChrW(8217) & ChrW(8220) & notThese \_

& ChrW(8221) & ChrW(8226) \_

& Chr(39) & "\-/\(\);:\!\?"" ]{1,}"

.Replacement.Highlight = False

.Replacement.Font.Color = wdColorAutomatic

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.Text = "^13{1,}"

.Replacement.Highlight = False

.Replacement.Font.Color = wdColorAutomatic

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' A funny space on the end of the line is invisible, so add markers

With rng.Find

.Text = "([" & ChrW(8192) & "-" & ChrW(8201) & "])" & "^13"

.Replacement.Text = myMarker & "\1" & myMarker & "^p"

.MatchWildcards = True

If strongHighlight > 0 Then .Replacement.Highlight = True

If strongColour > 0 Then .Replacement.Font.Color = strongColour

.Execute Replace:=wdReplaceAll

End With

' Give thin spaces a better Highlight

Options.DefaultHighlightColorIndex = thinSpaceHighlight

With rng.Find

.Text = ChrW(8201)

If strongHighlight > 0 Then .Replacement.Highlight = True

If strongColour > 0 Then .Replacement.Font.Color = strongColour

.Replacement.Text = "^&"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Beep

End If

' Special spaces

If myJob = 9 Then

Options.DefaultHighlightColorIndex = strongHighlight

Set rng = ActiveDocument.Content

With rng.Find

.Text = "[" & ChrW(8192) & "-" & ChrW(8201) & "]"

If strongHighlight > 0 Then .Replacement.Highlight = True

If strongColour > 0 Then .Replacement.Font.Color = strongColour

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' A funny space on the end of the line is invisible, so add markers

With rng.Find

.Text = "([" & ChrW(8192) & "-" & ChrW(8201) & "])" & "^13"

If strongHighlight > 0 Then .Replacement.Highlight = True

If strongColour > 0 Then .Replacement.Font.Color = strongColour

.Replacement.Text = myMarker & "\1" & myMarker & "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Give thin spaces a better Highlight

Options.DefaultHighlightColorIndex = thinSpaceHighlight

With rng.Find

.Text = ChrW(8201)

If strongHighlight > 0 Then .Replacement.Highlight = True

If strongColour > 0 Then .Replacement.Font.Color = strongColour

.Replacement.Text = "^&"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Give non-breaking spaces a highlight

Options.DefaultHighlightColorIndex = italicHighlight

With rng.Find

.Text = ChrW(160)

If strongHighlight > 0 Then .Replacement.Highlight = True

If strongColour > 0 Then .Replacement.Font.Color = strongColour

.Replacement.Text = "^&"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

' Symbol fonts

If myJob = 10 Then

Options.DefaultHighlightColorIndex = strongHighlight

Set rng = ActiveDocument.Content

With rng.Find

.Text = "[" & ChrW(-4096) & "-" & ChrW(-3000) & "]"

If strongHighlight > 0 Then .Replacement.Highlight = True

If strongColour > 0 Then .Replacement.Font.Color = strongColour

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

Options.DefaultHighlightColorIndex = oldHighlight

End Sub

**Sub PunctAlyse()**

**'** Paul Beverley - Version 18.10.11

' Produces usage statistics of various punctuation

Dim wasLen As Long

CR = vbCrLf: CR2 = CR & CR

Selection.WholeStory

Selection.Copy

Selection.HomeKey Unit:=wdStory

Documents.Add

' Selection.PasteAndFormat (wdFormatPlainText)

Selection.PasteSpecial DataType:=wdPasteText

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = "[,.][0-9]"

.Replacement.Text = " 0"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

myResults = "Punctuation Use" & CR2

wasLen = ActiveDocument.Range.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "."

.Replacement.Text = "^&!"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

pnts = ActiveDocument.Range.End - wasLen

If pnts > 0 Then WordBasic.editunDo

myResults = myResults & "Full points" & vbTab & Trim(Str(pnts)) & CR

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ","

.Replacement.Text = "^&!"

.Execute Replace:=wdReplaceAll

End With

coms = ActiveDocument.Range.End - wasLen

If coms > 0 Then WordBasic.editunDo

myResults = myResults & "Commas" & vbTab & Trim(Str(coms)) & CR

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ";"

.Replacement.Text = "^&!"

.Execute Replace:=wdReplaceAll

End With

semis = ActiveDocument.Range.End - wasLen

If semis > 0 Then WordBasic.editunDo

myResults = myResults & "Semicolons" & vbTab & Trim(Str(semis)) & CR

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ":"

.Replacement.Text = "^&!"

.Execute Replace:=wdReplaceAll

End With

colons = ActiveDocument.Range.End - wasLen

If colons > 0 Then WordBasic.editunDo

myResults = myResults & "Colons" & vbTab & Trim(Str(colons)) & CR

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "?"

.Replacement.Text = "^&!"

.Execute Replace:=wdReplaceAll

End With

qns = ActiveDocument.Range.End - wasLen

If qns > 0 Then WordBasic.editunDo

myResults = myResults & "Question marks" & vbTab & Trim(Str(qns)) & CR

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "!"

.Replacement.Text = "^&!"

.Execute Replace:=wdReplaceAll

End With

exclams = ActiveDocument.Range.End - wasLen

If exclams > 0 Then WordBasic.editunDo

myResults = myResults & "Exclamations" & vbTab & Trim(Str(exclams)) & CR2

sentCount = pnts + qns + exclams

commaFactor = Int(100 \* coms / sentCount) / 10

colonFactor = Int(1000 \* colons / sentCount) / 10

semiFactor = Int(1000 \* semis / sentCount) / 10

qnFactor = Int(1000 \* qns / sentCount) / 10

exclamFactor = Int(1000 \* exclams / sentCount) / 10

printMe = Trim(Str(commaFactor))

If Left(printMe, 1) = "." Then printMe = "0" & printMe

If InStr(printMe, ".") = 0 Then printMe = printMe & ".0"

myResults = myResults & "Comma Factor" & vbTab & printMe & CR

printMe = Trim(Str(semiFactor))

If Left(printMe, 1) = "." Then printMe = "0" & printMe

If InStr(printMe, ".") = 0 Then printMe = printMe & ".0"

myResults = myResults & "Semicolon Factor" & vbTab & printMe & CR

printMe = Trim(Str(colonFactor))

If Left(printMe, 1) = "." Then printMe = "0" & printMe

If InStr(printMe, ".") = 0 Then printMe = printMe & ".0"

myResults = myResults & "Colon Factor" & vbTab & printMe & CR

printMe = Trim(Str(qnFactor))

If Left(printMe, 1) = "." Then printMe = "0" & printMe

If InStr(printMe, ".") = 0 Then printMe = printMe & ".0"

myResults = myResults & "Question Factor" & vbTab & printMe & CR

printMe = Trim(Str(exclamFactor))

If Left(printMe, 1) = "." Then printMe = "0" & printMe

If InStr(printMe, ".") = 0 Then printMe = printMe & ".0"

myResults = myResults & "Exclamation Factor" & vbTab & printMe & CR2

Selection.WholeStory

Selection.TypeText Text:=myResults

Selection.WholeStory

Selection.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(4)

Selection.HomeKey Unit:=wdStory

End Sub

**Sub HighlightOffWord()**

' Paul Beverley - Version 16.01.21

' Remove highlight from all occurrences of this word

doNotes = True

' Select current word

If Selection.Start = Selection.End Then Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

theWord = Trim(Replace(Selection, ChrW(160), " "))

theWord = Replace(theWord, ChrW(8217) & "s", "")

theWord = Replace(theWord, Chr(9), "^t")

Selection.Collapse wdCollapseEnd

' Now unhighlight all occurrences of the word ...

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

If theWord <> "^t" Then

.Text = "<" & theWord & ">??"

Else

.Text = theWord

End If

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Replacement.Text = ""

.Replacement.Highlight = False

.Forward = True

.Execute Replace:=wdReplaceAll

End With

' ... and all occurrences of the word with a single (uncurly) quote

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<" & theWord & "'??"

.MatchWildcards = True

.Replacement.Text = ""

.Replacement.Highlight = False

.Execute Replace:=wdReplaceAll

End With

' ... and all occurrences of the word with a single quote

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<" & theWord & ChrW(8217) & "??"

.MatchWildcards = True

.Replacement.Text = ""

.Replacement.Highlight = False

.Execute Replace:=wdReplaceAll

End With

If doNotes = True And ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<" & theWord & ">??"

.MatchWildcards = True

.Replacement.Text = ""

.Replacement.Highlight = False

.Forward = True

.Execute Replace:=wdReplaceAll

End With

' ... and all occurrences of the word with a single quote

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<" & theWord & ChrW(8217) & "??"

.MatchWildcards = True

.Replacement.Text = ""

.Replacement.Highlight = False

.Execute Replace:=wdReplaceAll

End With

End If

If doNotes = True And ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<" & theWord & ">??"

.MatchWildcards = True

.Replacement.Text = ""

.Replacement.Highlight = False

.Forward = True

.Execute Replace:=wdReplaceAll

End With

' ... and all occurrences of the word with a single quote

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<" & theWord & ChrW(8217) & "??"

.MatchWildcards = True

.Replacement.Text = ""

.Replacement.Highlight = False

.Execute Replace:=wdReplaceAll

End With

End If

End Sub

**Sub SpellingErrorLister()**

' Paul Beverley - Version 02.03.22

' Generates an alphabetic list all the spelling 'errors'

spellingListName = "SpellingErrors"

myFind = "´a,´e,¨a,¨e,¨o,¨u,ˆo,"

myReplace = "á,é,ä,ë,ö,ü,ô,"

CR = vbCr

CR2 = CR & CR

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

' List possible spelling errors

thisLanguage = Selection.LanguageID

Select Case thisLanguage

Case wdEnglishUK: myLang = "UK spelling"

Case wdEnglishUS: myLang = "US spelling"

Case wdEnglishCanadian: myLang = "Canadian spelling"

Case Else: myLang = "unknown language"

End Select

myLang = "Using " & myLang & " dictionary. OK?"

If doingSeveralMacros = False Then

myResponse = MsgBox(myLang, vbQuestion + vbYesNoCancel, \_

"Spelling Error Lister")

If myResponse <> vbYes Then Exit Sub

End If

timeStart = Timer

langName = Languages(thisLanguage).NameLocal

Set rngOK = ActiveDocument.Content

OKstart = InStr(rngOK.Text, "OKwords")

If OKstart > 0 Then

rngOK.Start = OKstart + 6

OKwords = rngOK.Text

Else

OKwords = ""

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^2"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = "([!a-zA-Z])['" & ChrW(8216) & "]"

.Replacement.Text = "\1"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Changes ligature characters into character pairs

myFind = myFind & "," & ChrW(-1280) & "," & ChrW(-1279) & \_

"," & ChrW(-1278) & "," & ChrW(-1277) & "," \_

& ChrW(-1276)

myReplace = myReplace & ",ff,fi,fl,ffi,ffl"

fnd = Split(myFind, ",")

rpl = Split(myReplace, ",")

For i = 0 To UBound(fnd)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

Next i

' Create spelling error list

erList1 = CR

erList2 = CR

numFootnotes = ActiveDocument.Footnotes.Count

numEndnotes = ActiveDocument.Endnotes.Count

myEnd = ActiveDocument.Content.End

For i = 1 To 3

If myResponse = vbNo Then i = 3

If i = 1 And numFootnotes = 0 Then i = 2

If i = 1 Then Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

If i = 2 And numEndnotes = 0 Then i = 3

If i = 2 Then Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

If i = 3 Then Set rng = ActiveDocument.Content

For Each wd In rng.Words

If Len(Trim(wd)) > 2 And LCase(wd) <> UCase(wd) And \_

wd.Font.StrikeThrough = False And wd <> "OKwords" Then

padWd = " " & Trim(wd) & " "

OKword = (InStr(OKwords, CR & Trim(wd) & CR) > 0)

DoEvents

If Application.CheckSpelling(wd, MainDictionary:=langName) = \_

False And OKword = False Then

pCent = Int((myEnd - wd.End) / myEnd \* 100)

' Report progress

If i = 1 Then myPrompt = "Checking footnote text."

If i = 2 Then myPrompt = "Checking endnote text."

If i = 3 Then myPrompt = "Checking main text."

StatusBar = "Generating errors list. " & myPrompt & \_

" To go: " & Trim(Str(pCent)) & "%"

Debug.Print "Generating errors list. " & myPrompt & \_

" To go: " & Trim(Str(pCent)) & "%"

erWord = Trim(wd)

lastChar = Right(erWord, 1)

If lastChar = "'" Or lastChar = ChrW(8217) Then

erWord = Left(erWord, Len(erWord) - 1)

End If

myCap = Left(erWord, 1)

If UCase(myCap) = myCap Then

If InStr(erList1, CR & erWord & CR) = 0 Then erList1 = erList1 \_

& erWord & CR

Else

If InStr(erList2, CR & erWord & CR) = 0 Then erList2 = erList2 \_

& erWord & CR

End If

End If

End If

Next wd

Next i

mainFileName = ActiveDocument.Name

Documents.Add

Selection.TypeText erList2

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

If erList1 <> CR Then

Selection.EndKey Unit:=wdStory

Selection.TypeText CR2

listStart = Selection.Start

Selection.TypeText erList1

Selection.Start = listStart

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

End If

Selection.WholeStory

Selection.LanguageID = thisLanguage

Selection.Style = wdStyleNormal

Selection.Collapse wdCollapseStart

If numFootnotes > 0 Then

Selection.TypeText CR & "| footnotes = yes" & CR

End If

If numEndnotes > 0 Then

Selection.TypeText CR & "| endnotes = yes" & CR

End If

StatusBar = ""

Selection.HomeKey Unit:=wdStory

Selection.TypeText spellingListName & vbCr

ActiveDocument.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading1)

If doingSeveralMacros = False Then

totTime = Int(10 \* (Timer - timeStart) / 60) / 10

If totTime > 2 Then myResponse = MsgBox((totTime & " minutes"), \_

vbOKOnly, "Spelling Error Lister")

Beep

Else

FUT.Activate

End If

End Sub

**Sub SpellingErrorHighlighter()**

**'** Paul Beverley - Version 11.04.21

' Highlights all spelling errors

spellingListName = "SpellingErrors"

spellingListName2 = "SpellAlyse"

anyCase = False

CR = vbCr

CR2 = CR & CR

SP = "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"

' Find errors list

Set mainDoc = ActiveDocument

gottaList = False

For Each myDoc In Documents

myFirstLine = myDoc.Paragraphs(1).Range.Text

If InStr(myFirstLine, spellingListName & CR) > 0 \_

Or InStr(myFirstLine, spellingListName2 & CR) > 0 Then

myDoc.Activate

gottaList = True

Exit For

End If

Next myDoc

If gottaList = False Then

Beep

MsgBox ("Please load file: """ & spellingListName & """")

Exit Sub

End If

' Create list of words needing highlighting in each colour

Dim myWordHighlightList(16) As String

myCount = 0

For Each myPar In ActiveDocument.Paragraphs

Set rng = myPar.Range.Duplicate

If Len(rng) > 2 Then

thisWord = Replace(rng.Text, CR, "")

myCol = rng.HighlightColorIndex

If myCol > 0 And myCol < 17 Then

myWordHighlightList(myCol) = myWordHighlightList(myCol) & \_

thisWord & "\_"

myCount = myCount + 1

StatusBar = SP & SP & SP & myCount

End If

End If

Next myPar

totCount = myCount

If myCount = 0 Then

Beep

myResponse = MsgBox("Please highlight at least one word in the list!", \_

vbQuestion + vbOK, "SpellingErrorHighlighter")

Exit Sub

End If

mainDoc.Activate

Set rng = ActiveDocument.Content

' To speed up search

Selection.HomeKey Unit:=wdStory

fnNum = ActiveDocument.Footnotes.count

enNum = ActiveDocument.Endnotes.count

ActiveDocument.TrackRevisions = False

' For each highlight colour

oldColour = Options.DefaultHighlightColorIndex

If anyCase = True Then

For myCol = 1 To 16

If Len(myWordHighlightList(myCol)) > 0 Then

Options.DefaultHighlightColorIndex = myCol

myWds = Split(myWordHighlightList(myCol), "\_")

For Each fWord In myWds

For j = 1 To 3

If j = 1 And fnNum = 0 Then j = 2

If j = 2 And enNum = 0 Then j = 3

Select Case j

Case 1: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

DoEvents

If Len(fWord) > 3 Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fWord

.Replacement.Text = ""

.Font.StrikeThrough = False

.Forward = True

.MatchCase = False

.Replacement.Highlight = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

If j = 3 Then

myCount = myCount - 1

StatusBar = SP & SP & SP & "To go: " & myCount

End If

Next j

Debug.Print fWord

Next fWord

End If

Next myCol

Else

For myCol = 1 To 16

If Len(myWordHighlightList(myCol)) > 0 Then

Options.DefaultHighlightColorIndex = myCol

myWds = Split(myWordHighlightList(myCol), "\_")

For Each fWord In myWds

fWord = "<" & fWord & ">"

fWord = Replace(fWord, "(", "\(")

fWord = Replace(fWord, ")", "\)")

fWord = Replace(fWord, "\)>", "\)")

For j = 1 To 3

If j = 1 And fnNum = 0 Then j = 2

If j = 2 And enNum = 0 Then j = 3

Select Case j

Case 1: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

DoEvents

If Len(fWord) > 3 Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fWord

.Replacement.Text = "^&"

.Font.StrikeThrough = False

.Forward = True

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "-" & fWord

.Replacement.Text = "^&"

.Font.StrikeThrough = False

.Forward = True

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

If j = 3 Then

myCount = myCount - 1

StatusBar = SP & SP & SP & "To go: " & myCount

End If

Next j

Debug.Print fWord

Next fWord

End If

Next myCol

End If

Options.DefaultHighlightColorIndex = oldColour

Beep

StatusBar = " "

myResponse = MsgBox("All those errors have been highlighted", \_

vbOKOnly, "Spelling Error Highlighter")

End Sub

**Sub SpellingErrorListerBilingual()**

' Paul Beverley - Version 03.09.20

' Generates an alphabetic list all the bilingual spelling 'errors'

myLanguage\_1 = wdEnglishUK

' myLanguage\_2 = wdPortugueseBrazil

' myLanguage\_2 = wdPortugueseBoth

myLanguage\_2 = wdFrench

spellingListName = "SpellingErrors"

myFind = "´a,´e,¨a,¨e,¨o,¨u,ˆo,"

myReplace = "á,é,ä,ë,ö,ü,ô,"

CR = vbCr

CR2 = CR & CR

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

lang1 = Languages(myLanguage\_1).NameLocal

lang2 = Languages(myLanguage\_2).NameLocal

myLang = "Using " & lang1 & " and " & lang2 & " dictionary. OK?"

If doingSeveralMacros = False Then

myResponse = MsgBox(myLang, vbQuestion + vbYesNoCancel, \_

"Spelling Error Lister")

If myResponse <> vbYes Then Exit Sub

End If

timeStart = Timer

' lang = Languages(thisLanguage).NameLocal

Set rngOK = ActiveDocument.Content

OKstart = InStr(rngOK.Text, "OKwords")

If OKstart > 0 Then

rngOK.Start = OKstart + 6

OKwords = rngOK.Text

Else

OKwords = ""

End If

' Changes ligature characters into character pairs

myFind = myFind & "," & ChrW(-1280) & "," & ChrW(-1279) & \_

"," & ChrW(-1278) & "," & ChrW(-1277) & "," \_

& ChrW(-1276)

myReplace = myReplace & ",ff,fi,fl,ffi,ffl"

fnd = Split(myFind, ",")

rpl = Split(myReplace, ",")

For i = 0 To UBound(fnd)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

Next i

' Create spelling error list

erList1 = CR

erList2 = CR

numFootnotes = ActiveDocument.Footnotes.Count

numEndnotes = ActiveDocument.Endnotes.Count

myEnd = ActiveDocument.Content.End

For i = 1 To 3

If myResponse = vbNo Then i = 3

If i = 1 And numFootnotes = 0 Then i = 2

If i = 1 Then Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

If i = 2 And numEndnotes = 0 Then i = 3

If i = 2 Then Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

If i = 3 Then Set rng = ActiveDocument.Content

For Each wd In rng.Words

If Len(Trim(wd)) > 2 And LCase(wd) <> UCase(wd) And \_

wd.Font.StrikeThrough = False And wd <> "OKwords" Then

padWd = " " & Trim(wd) & " "

OKword = (InStr(OKwords, CR & Trim(wd) & CR) > 0)

DoEvents

If Application.CheckSpelling(wd, MainDictionary:=lang1) = False \_

And Application.CheckSpelling(wd, MainDictionary:=lang2) = \_

False And OKword = False Then

pCent = Int((myEnd - wd.End) / myEnd \* 100)

' Report progress

If i = 1 Then myPrompt = "Checking footnote text."

If i = 2 Then myPrompt = "Checking endnote text."

If i = 3 Then myPrompt = "Checking main text."

StatusBar = "Generating errors list. " & myPrompt & \_

" To go: " & Trim(Str(pCent)) & "%"

Debug.Print "Generating errors list. " & myPrompt & \_

" To go: " & Trim(Str(pCent)) & "%"

erWord = Trim(wd)

lastChar = Right(erWord, 1)

If lastChar = "'" Or lastChar = ChrW(8217) Then \_

erWord = Left(erWord, Len(erWord) - 1)

myCap = Left(erWord, 1)

If UCase(myCap) = myCap Then

If InStr(erList2, CR & erWord & CR) = 0 Then \_

erList2 = erList2 & erWord & CR

Else

If InStr(erList1, CR & erWord & CR) = 0 Then \_

erList1 = erList1 & erWord & CR

End If

End If

End If

Next wd

Next i

mainFileName = ActiveDocument.Name

Documents.Add

Selection.TypeText erList1

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

erList1 = Selection

Selection.Delete

Selection.TypeText erList2

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

erList2 = Selection

Selection.Delete

Selection.TypeText Text:=erList1

'Selection.TypeText CR2

Selection.TypeText Text:=erList2

'Selection.TypeText CR2

Selection.HomeKey Unit:=wdStory

If numFootnotes > 0 Then

Selection.TypeText CR & "| footnotes = yes" & CR

End If

If numEndnotes > 0 Then

Selection.TypeText CR & "| endnotes = yes" & CR

End If

StatusBar = ""

Selection.TypeText spellingListName & vbCr

ActiveDocument.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading1)

If doingSeveralMacros = False Then

totTime = Int(10 \* (Timer - timeStart) / 60) / 10

If totTime > 2 Then myResponse = MsgBox((totTime & " minutes"), \_

vbOKOnly, "Spelling Error Lister")

Beep

Else

FUT.Activate

End If

End Sub

**Sub ConcordanceMaker()**

' Paul Beverley - Version 17.04.20

' Creates a concordance list

sortByLength = True

minLength = 4

lengthAscending = False

CR = vbCr

myList = CR

maxLen = 1

For Each myPara In ActiveDocument.Paragraphs

For Each wd In myPara.Range.Words

w = Trim(wd)

w = Replace(w, "'", "")

w = Replace(w, ChrW(8217), "")

Debug.Print w

If Len(w) > minLength - 1 Then

wCap = Trim(UCase(wd))

w = LCase(w)

If InStr(myList, CR & w & CR) = 0 And w <> wCap Then

myList = myList & w & CR

wLen = Len(w)

If wLen > maxLen Then maxLen = wLen

End If

End If

DoEvents

myPara.Range.Select

Next wd

Next myPara

Documents.Add

Selection.TypeText Text:=myList

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

Selection.HomeKey Unit:=wdStory

ReDim p(maxLen) As String

If lengthAscending = True Then

For i = maxLen To 1 Step -1

pees = pees & "+"

p(i) = pees

Next i

Else

For i = 1 To maxLen

pees = pees & "+"

p(i) = pees

Next i

End If

If sortByLength = True Then

Documents.Add

Selection.Text = rng.Text

For Each pa In ActiveDocument.Content.Paragraphs

wLen = Len(pa.Range.Words(1))

pa.Range.InsertBefore Text:=p(wLen)

Next pa

End If

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[+]{1,}"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Beep

End Sub

**Sub DuplicatedWordsHighlight()**

' Paul Beverley - Version 03.02.16

' Add a highlight to any duplicate words in a text, e.g. "the the"

myColour1 = wdGray25

myColour2 = wdBrightGreen

myColour3 = wdYellow

doThreeWords = True

find1 = "(<[a-zA-Z]{1,})[ .,\!\?:;]{1,}\1[ .,\!\?:;]"

find2 = "(<[a-zA-Z]{1,}^32[a-zA-Z]{1,})[ .,\!\?:;]{1,}\1[ .,\!\?:;]"

find3 = "(<[a-zA-Z]{1,}^32[a-zA-Z]{1,}^32[a-zA-Z]{1,})" \_

& "[ .,\!\?:;]{1,}\1[ .,\!\?:;]"

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour1

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = find1

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = myColour2

With rng.Find

.Text = find2

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

If doThreeWords = True Then

Options.DefaultHighlightColorIndex = myColour3

With rng.Find

.Text = find3

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub DuplicatedWordsFind()**

' Paul Beverley - Version 03.02.16

' Find next consecutive duplicate word (idea from Douglas Vipond)

With Selection.Find

.ClearFormatting

.Text = "(<[a-zA-Z]{1,})[ .,\!\?:;]{1,}\1[ .,\!\?:;]"

.Replacement.Text = ""

.Forward = True

.Wrap = False

.Format = False

.MatchCase = False

.MatchWildcards = True

.Execute

End With

If Selection.Find.Found = False Then Beep

End Sub

**Sub DuplicatedWordsFind2()**

' Paul Beverley - Version 03.02.16

' Finds next consecutive two-word duplicate

With Selection.Find

.ClearFormatting

.Text = "(<[a-zA-Z]{1,}^32[a-zA-Z]{1,})[ .,\!\?:;]{1,}\1[ .,\!\?:;]"

.Replacement.Text = ""

.Forward = True

.Wrap = False

.Format = False

.MatchCase = False

.MatchWildcards = True

.Execute

End With

If Selection.Find.Found = False Then Beep

End Sub

**Sub DuplicatedWordsFind3()**

' Paul Beverley - Version 03.02.16

' Find next consecutive three-word duplicate

With Selection.Find

.ClearFormatting

.Text = "(<[a-zA-Z]{1,}^32[a-zA-Z]{1,}^32[a-zA-Z]{1,})" \_

& "[ .,\!\?:;]{1,}\1[ .,\!\?:;]"

.Replacement.Text = ""

.Forward = True

.Wrap = False

.Format = False

.MatchCase = False

.MatchWildcards = True

.Execute

End With

If Selection.Find.Found = False Then Beep

End Sub

**Sub ProperNounAlyse()**

**'** Paul Beverley - Version 21.12.24

' Analyses similar proper nouns

minLengthCheck = 5

includeAcronyms = True

myLanguage = "English(United Kingdom)"

' myLanguage = "English(United States)"

ignoreWords = "The This There Those Their They Then These That"

similarChars = "bb,b; b,p; sch,sh; ch,sh; c,k; ph,f; ss,z; s,z;" & \_

" mp,m; ll,l; nn,n; nd,n; nt,n;"

' With non-English languages, you might need to make this False

ignorePlurals = True

CR = vbCr: CR2 = CR & CR

myScreenOff = True

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") + \_

InStr(FUT.Name, "Document") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox(" ProperNounAlyse" & CR2 & \_

"Analyse this document?", vbQuestion \_

+ vbYesNoCancel, "ProperNounAlyse")

If myResponse <> vbYes Then Exit Sub

End If

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

myDummy = ChrW(222)

For i = 1 To 100

spcs = " " & spcs

Next i

checkFinalLetters = True

' checkFinalLetters = False

' Grey on word only

thisHighlight = wdGray25

doMissingLetter = True

' doMissingLetter = False

' Bold And blue

switchTest = True

' switchTest = False

' double strikethrough

doSimilarLetters = True

' doSimilarLetters = False

' various highlight colours + underline

doVowelTest = True

' doVowelTest = False

' various highlight colours + italic

' These last two tests cycle through these colours:

maxCol = 6

ReDim myCol(maxCol) As Integer

myCol(1) = wdYellow

myCol(2) = wdBrightGreen

myCol(3) = wdTurquoise

myCol(4) = wdRed

myCol(5) = wdPink

myCol(6) = wdGray25

colcode = 0

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = wdGray25

leadDots = " . . . "

title1 = "Proper noun list"

title2 = "Proper noun queries"

convCharsUC = "AAAAAAA.EEEEIIII..OOOOO.OUUUU" & \_

"...aaaaaaa.eeeeiiiio.ooooo.ouuuu......"

convCharsLC = LCase(convCharsUC)

timeStart = Timer

' collect notes text, if any

endText = ""

footText = ""

If ActiveDocument.Endnotes.count > 0 Then

endText = ActiveDocument.StoryRanges(wdEndnotesStory).Text

End If

If ActiveDocument.Footnotes.count > 0 Then

footText = ActiveDocument.StoryRanges(wdFootnotesStory).Text

End If

' collect text in all the textboxes (if any)

sh = ActiveDocument.Shapes.count

If sh > 0 Then

ReDim shText(sh)

i = 0

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

i = i + 1

shText(i) = shp.TextFrame.TextRange.Text

End If

End If

Next

shCount = i

End If

' Create various documents

Set rng = ActiveDocument.Content

Set firstDoc = Documents.Add

Set fnl = ActiveDocument.Content

Set allText = Documents.Add

Selection.FormattedText = rng.FormattedText

Selection.Collapse wdCollapseEnd

' Add notes + shape text

Selection.TypeText endText & CR & footText & CR

If shCount > 0 Then

For i = 1 To shCount

Selection.TypeText shText(i) & CR

Next i

End If

Selection.HomeKey Unit:=wdStory

Set rng = allText.Content

rng.Revisions.AcceptAll

DoEvents

StatusBar = spcs & "Preparing copied file - 1"

' Delete struck-through text

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.MatchWildcards = False

.Font.StrikeThrough = True

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "["

.MatchWildcards = False

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

' Remove strange unicode characters

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & ChrW(&HA000) & "-" & ChrW(&HD6FF) & "]{1,}"

.MatchWildcards = True

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

DoEvents

StatusBar = spcs & "Preparing copied file - 2"

DoEvents

' Cut all and replace as pure text

Set rng = allText.Content

allJustText = allText.Content.Text

allText.Content.Delete

rng.Text = allJustText

DoEvents

StatusBar = spcs & "Preparing copied file - 3"

' Use qqq for apostrophe

Set rng = allText.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "n" & ChrW(8217) & "t"

.MatchWildcards = False

.Replacement.Text = "nqqqt"

.Execute Replace:=wdReplaceAll

End With

' Use qq for apostrophe

With rng.Find

.Text = "O'"

.MatchCase = True

.Replacement.Text = "Oqqq"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Find initial cap words

DoEvents

StatusBar = spcs & "Preparing copied file - 4"

myChopNum = minLengthCheck - 2

If myChop < 1 Then myChop = 1

myChop = Trim(Str(myChopNum))

myFind = "<[A-Z][a-z][a-zA-Z]{" & myChop & ",}"

If includeAcronyms = True Then myFind = \_

"<[A-Z][a-zA-Z][a-zA-Z]{" & myChop & ",}"

Set rng = allText.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

myPNlist = ""

ignoreWords = ignoreWords & " "

Do While rng.Find.Found = True

myWd = rng.Text

If InStr(ignoreWords, myWd & " ") = 0 Then \_

myPNlist = myPNlist & myWd & CR

rng.Collapse wdCollapseEnd

i = i + 1: If i Mod 100 = 0 Then DoEvents

rng.Find.Execute

DoEvents

Loop

allText.Content.Text = myPNlist

StatusBar = spcs & "Sorting whole file"

Set allText = ActiveDocument

wordArray = Split(allText.Content.Text, CR)

' Sort the array alphabetically

Call QuickSort(wordArray, LBound(wordArray), UBound(wordArray))

StatusBar = spcs & "After sorting"

' Clear the content of the document

allText.Content.Delete

' Insert sorted words into the document

For i = LBound(wordArray) To UBound(wordArray)

allText.Content.InsertAfter wordArray(i) & CR

Next i

' delete initial blank line

If Len(allText.Paragraphs(1)) < 3 Then \_

allText.Paragraphs(1).Range.Delete

If Len(allText.Paragraphs(1)) < 3 Then \_

allText.Paragraphs(1).Range.Delete

' Create a frequency for each highlighted word

thisWord = ""

myCount = 0

For Each myPara In allText.Paragraphs

nextWord = myPara.Range.Words(1)

If nextWord <> thisWord Then

' This is a new word

If Len(thisWord) > 1 Then

fnl.InsertAfter Text:=thisWord \_

& leadDots & Trim(Str(myCount)) & CR

End If

thisWord = nextWord

myCount = 1

Else

myCount = myCount + 1

End If

i = i + 1: If i Mod 400 = 4 Then DoEvents

Next myPara

allText.Close SaveChanges:=False

firstDoc.Activate

' Remove blank lines

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{2,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Resort case insensitively

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

CaseSensitive:=False

' Delete rubbish from top and bottom of list

Do

Set rng = ActiveDocument.Paragraphs(1).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until myLen > 9

Do

lastLine = ActiveDocument.Paragraphs.count

Set rng = ActiveDocument.Paragraphs(lastLine).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until Len(rng.Text) >= 2

' Word list now has freq. count.

Do

lastLine = ActiveDocument.Paragraphs.count

Set rng = ActiveDocument.Paragraphs(lastLine).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until Len(rng.Text) >= 2

' Create another copy for doing extra tests

Set rng = ActiveDocument.Content

Set finalList = Documents.Add

finalList.Range.Text = rng.Text

Selection.HomeKey Unit:=wdStory

' Prepare data for other tests

numWords = ActiveDocument.Paragraphs.count

For i = 1 To numWords

aWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(aWord)

thisChar = ChrW(n)

If n > 129 Then

If n >= 217 Then aWord = Replace(aWord, thisChar, "U")

If n >= 210 Then aWord = Replace(aWord, thisChar, "O")

If n >= 204 Then aWord = Replace(aWord, thisChar, "I")

If n >= 200 Then aWord = Replace(aWord, thisChar, "E")

If n >= 192 Then aWord = Replace(aWord, thisChar, "A")

End If

allWords = allWords & aWord

jmp = 100

If i Mod jmp = 1 Then

pq = pq + 1

DoEvents

StatusBar = spcs & \_

"Preparing data for other tests - 1 - " & pq

DoEvents

End If

Next i

' ...for the vowel test below

DoEvents

StatusBar = spcs & "Preparing data for other tests - 2"

DoEvents

noVowelWords = " " & allWords

noVowelWords = Replace(noVowelWords, " A", "\_1")

noVowelWords = Replace(noVowelWords, " E", "\_2")

noVowelWords = Replace(noVowelWords, " I", "\_3")

noVowelWords = Replace(noVowelWords, " O", "\_4")

noVowelWords = Replace(noVowelWords, " U", "\_5")

noVowelWords = Replace(noVowelWords, " Y", "\_6")

For k = 2 To Len(noVowelWords) - 1

thisChar = Mid(noVowelWords, k, 1)

n = AscW(thisChar)

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsLC, n - 191, 1)

If myNewChar <> "." Then noVowelWords = \_

Replace(noVowelWords, thisChar, myNewChar)

End If

Next k

noVowelWords = Replace(noVowelWords, "a", "")

noVowelWords = Replace(noVowelWords, "e", "")

noVowelWords = Replace(noVowelWords, "i", "")

noVowelWords = Replace(noVowelWords, "o", "")

noVowelWords = Replace(noVowelWords, "u", "")

noVowelWords = Replace(noVowelWords, "y", "")

noVowelWords = Replace(noVowelWords, "A", "")

noVowelWords = Replace(noVowelWords, "E", "")

noVowelWords = Replace(noVowelWords, "I", "")

noVowelWords = Replace(noVowelWords, "O", "")

noVowelWords = Replace(noVowelWords, "U", "")

noVowelWords = Replace(noVowelWords, "Y", "")

noVowelWords = Replace(noVowelWords, "\_1", " A")

noVowelWords = Replace(noVowelWords, "\_2", " E")

noVowelWords = Replace(noVowelWords, "\_3", " I")

noVowelWords = Replace(noVowelWords, "\_4", " O")

noVowelWords = Replace(noVowelWords, "\_5", " U")

noVowelWords = Replace(noVowelWords, "\_6", " Y")

' ...for the similar words test

DoEvents

StatusBar = spcs & "Preparing data for other tests - 3"

DoEvents

similarAllWords = " " & LCase(allWords)

similarChars = Replace(similarChars, " ", "")

sChars = Replace(similarChars, " ", "")

Do

commaPos = InStr(sChars, ",")

charWas = Left(sChars, commaPos - 1)

sChars = Mid(sChars, commaPos + 1)

semicolonPos = InStr(sChars, ";")

charNew = Left(sChars, semicolonPos - 1)

sChars = Mid(sChars, semicolonPos + 1)

similarAllWords = Replace(similarAllWords, charWas, charNew)

Loop Until Len(sChars) < 2

' Changes all the accented characters to non-accented

DoEvents

StatusBar = spcs & "Preparing data for other tests - 4"

DoEvents

sWd = similarAllWords

For k = 1 To Len(sWd) - 1

thisChar = Mid(sWd, k, 1)

n = AscW(thisChar)

myNewChar = "."

If n > 191 And n < 256 Then

myNewChar = Mid(convCharsLC, n - 191, 1)

If myNewChar <> "." Then sWd = Replace(sWd, \_

thisChar, myNewChar)

End If

Next k

similarAllWords = sWd

' Catch words with only the final two letters the same

i = 0

If checkFinalLetters = True Then

For Each myPara In ActiveDocument.Paragraphs

gotOne = False

myWord = Trim(myPara.Range.Words(1))

myLen = Len(myWord)

If myLen > 6 Then

myTarget = "^p" & Left(myWord, myLen - 2) & "^$^$ "

myCut = 2

Else

myTarget = "^p" & Left(myWord, myLen - 1) & "^$ "

myCut = 1

End If

Set rng = ActiveDocument.Content

rng.start = myPara.Range.End - 3

rng.Collapse wdCollapseStart

With rng.Find

.Replacement.ClearFormatting

.ClearFormatting

.Text = myTarget

.Replacement.Text = ""

.Forward = True

.MatchCase = True

.MatchWildcards = False

.Wrap = wdFindStop

End With

rng.Find.Execute

Do While rng.Find.Found

gotOne = True

rng.MoveStart 1

rng.End = rng.start + myLen - myCut

rng.HighlightColorIndex = thisHighlight

rng.Font.Bold = True

rng.Find.Execute

Loop

If gotOne = True Then

Set rng = myPara.Range.Words(1)

rng.End = rng.start + myLen - myCut

rng.HighlightColorIndex = thisHighlight

rng.Font.Bold = True

End If

i = i + 1

If i Mod 100 = 1 Then

DoEvents

StatusBar = spcs & "Doing test (5) on " & myWord

DoEvents

End If

Next myPara

End If

If doMissingLetter = True Then

' Start of test

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Changes the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (4) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' Check if word reappears with one letter missing (1)

For k = 2 To Len(myWord) - 1

testWord = " " & Left(myWord, k - 1) & Mid(myWord, k + 1)

wordPos = InStr(allWords, testWord)

If wordPos > 0 Then

lastLetter = Mid(myWord, Len(myWord) - 1, 1)

' but not "s" at the end, unless it's a spelling error

If lastLetter = "s" Then

ignoreIt = (Application.CheckSpelling(myWord, \_

MainDictionary:=myLanguage) = True)

Else

ignoreIt = False

End If

If ignoreIt = False And ignorePlurals = True Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

' mark the pair

leftBit = Left(allWords, InStr(allWords, testWord) \_

+ Len(testWord) - 1)

j = Len(leftBit) - Len(Replace(leftBit, " ", ""))

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Bold = True

rng.Font.Color = wdColorBlue

Set rng = ActiveDocument.Paragraphs(j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Bold = True

rng.Font.Color = wdColorBlue

End If

End If

Next k

If Left(myWord, 2) = "Mc" Or Left(myWord, 3) = "Mac" Or \_

Left(myWord, 3) = "Mag" Then

McList = McList & ActiveDocument.Paragraphs(i).Range

End If

Next i

End If

If doSimilarLetters = True Then

doneWords = ""

doneSimilarWords = ""

For i = 1 To ActiveDocument.Paragraphs.count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Changes the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (3) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check similar spellings: Perutz/Peruts or Chebyshev/Chevychev

similarWord = " " & LCase(myWord)

sChars = similarChars

Do

commaPos = InStr(sChars, ",")

charWas = Left(sChars, commaPos - 1)

sChars = Mid(sChars, commaPos + 1)

semicolonPos = InStr(sChars, ";")

charNew = Left(sChars, semicolonPos - 1)

sChars = Mid(sChars, semicolonPos + 1)

similarWord = Replace(similarWord, charWas, charNew)

Loop Until Len(sChars) < 2

' Changes all the accented characters to non-accented

For k = 1 To Len(myWord) - 1

thisChar = Mid(myWord, k, 1)

n = AscW(thisChar)

If n > 191 And n < 256 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

Next k

similarAllWords = Mid(similarAllWords, Len(similarWord))

theseWords = similarAllWords

If InStr(doneSimilarWords, similarWord) = 0 And \_

InStr(theseWords, similarWord) > 0 Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Underline = True

doneSimilarWords = doneSimilarWords & similarWord

' search through all the following words

theseWords = similarAllWords

For j = 1 To numWords - i

spPos = InStr(Trim(theseWords) & " ", " ")

If Left(theseWords, spPos + 1) = similarWord Then

Set rng = ActiveDocument.Paragraphs(i + j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Underline = True

End If

theseWords = Mid(theseWords, spPos + 1)

capThisLetter = Mid(theseWords, 2, 1)

If capThisLetter <> LCase(captestLetters) Then Exit For

Next j

End If

Next i

End If

If switchTest = True Then

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Changes the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (2) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check for switched chars

wordLen = Len(myWord) - 1

For k = 1 To Len(myWord) - 3

otherWord = Left(myWord, k) & Mid(myWord, k + 2, 1) & \_

Mid(myWord, k + 1, 1) & Mid(myWord, k + 3)

wordPos = InStr(testWords, otherWord)

If wordPos > 0 Then

' Find the position of the matching word

matchWord = Mid(testWords, wordPos, Len(myWord))

leftBit = Left(allWords, InStr(allWords, matchWord) + 1)

j = Len(leftBit) - Len(Replace(leftBit, " ", "")) + 1

ActiveDocument.Paragraphs(i).Range.Font.DoubleStrikeThrough \_

= True

ActiveDocument.Paragraphs(i).Range.HighlightColorIndex \_

= thisCol

ActiveDocument.Paragraphs(j).Range.Font.DoubleStrikeThrough \_

= True

ActiveDocument.Paragraphs(j).Range.HighlightColorIndex \_

= thisCol

End If

Next k

Next i

End If

If doVowelTest = True Then

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Changes the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (1) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check if there's a word with different vowels

otherWord = " " & Replace(myWord, "a", "")

otherWord = Replace(otherWord, "e", "")

otherWord = Replace(otherWord, "i", "")

otherWord = Replace(otherWord, "o", "")

otherWord = Replace(otherWord, "u", "")

otherWord = Replace(otherWord, "y", "")

' Delete all the accented characters

For k = 3 To Len(otherWord) - 1

thisChar = Mid(otherWord, k, 1)

n = AscW(thisChar)

If InStr("AEIOUY", thisChar) > 0 Then

otherWord = Left(otherWord, k - 1) & "=" & Mid(otherWord, k + 1)

Else

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then

otherWord = Replace(otherWord, thisChar, "=")

End If

End If

End If

Next k

otherWord = Replace(otherWord, "=", "")

' otherWord is now the word under test (vowel-less)

otherWord = Replace(otherWord, ".", "")

noVowelWords = Mid(noVowelWords, Len(otherWord))

If Left(noVowelWords, 1) <> " " Then noVowelWords = \_

" " & noVowelWords

theseWords = noVowelWords

wordPos = InStr(noVowelWords, otherWord)

If InStr(doneWords, otherWord) = 0 And wordPos > 0 Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Italic = True

doneWords = doneWords & otherWord

For j = 1 To numWords - i

spPos = InStr(Trim(theseWords) & " ", " ")

firstWord = Left(theseWords, spPos + 1)

theseWords = Mid(theseWords, spPos + 1)

If firstWord = otherWord Then

Set rng = ActiveDocument.Paragraphs(i + j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Italic = True

End If

capThisLetter = Mid(theseWords, 2, 1)

If capThisLetter > "" And capThisLetter <> \_

captestLetters Then Exit For

Next j

End If

Next i

End If

finishOff:

Selection.EndKey Unit:=wdStory

Selection.TypeText CR2 & McList

Selection.HomeKey Unit:=wdStory

Selection.TypeText title1 & CR

Do

Selection.Expand wdParagraph

If Len(Selection) < 3 Or LCase(Selection) = \_

UCase(Selection) Then Selection.Delete

Loop Until LCase(Selection) <> UCase(Selection)

Selection.HomeKey Unit:=wdStory, Extend:=wdExtend

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

' Restore apostrophes

Set rng = finalList.Range

With rng.Find

.Text = "qqq"

.MatchCase = False

.Replacement.Text = "'"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Find first highlight

Set rng = finalList.Content

With rng.Find

.Text = "Zzzzz"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

Set rng = finalList.Content

With rng.Find

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

rng.Select

Selection.Collapse wdCollapseStart

Set finalList = ActiveDocument

firstDoc.Activate

' Find sets of sounds-like words

StatusBar = spcs & "Sounds-like tests"

k = 0

For Each myPara In ActiveDocument.Paragraphs

myWord = Trim(myPara.Range.Words(1))

k = k + 1

If k Mod 40 = 1 Then

DoEvents

StatusBar = spcs & "Sounds-like test: " & myWord

DoEvents

End If

hasAccent = False

For i = 1 To Len(myWord)

ascChar = AscW(Mid(myWord, i))

If ascChar > 128 Or ascChar = Asc("?") Then hasAccent = True

Next i

' Go and find the first sounds-like word

initLetter = Left(myWord, 1)

If Len(myWord) > 2 And myPara.Range.HighlightColorIndex > 0 And \_

hasAccent = False And InStr(allSets, myWord & leadDots) \_

= 0 Then

Set rng = ActiveDocument.Content

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myWord

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = False

.MatchSoundsLike = True

.Execute

End With

Set myPara = rng.Paragraphs(1).Range

rng.Collapse wdCollapseEnd

Loop Until Left(myPara, 1) = initLetter

setOfWords = myPara

gottaSet = False

rng.Collapse wdCollapseEnd

rng.Find.Execute

Do While rng.Find.Found = True

Set myPara = rng.Paragraphs(1).Range

If Left(myPara, 1) = initLetter Then

gottaSet = True

setOfWords = setOfWords & myPara

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

If gottaSet = True Then allSets = allSets & setOfWords & CR

End If

Next myPara

Selection.WholeStory

If Len(allSets) < 2 Then

Selection.TypeText "None found with this test"

Else

Selection.TypeText allSets

End If

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Proper nouns by sound" & CR

Selection.HomeKey Unit:=wdStory, Extend:=wdExtend

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = 0

rng.Copy

ActiveDocument.Close SaveChanges:=False

finalList.Activate

' Remove highlighting from second half of words

' that are only case changes of one another

totParas = ActiveDocument.Paragraphs.count

For i = 1 To totParas - 1

A = Trim(ActiveDocument.Paragraphs(i).Range.Words(1))

b = Trim(ActiveDocument.Paragraphs(i + 1).Range.Words(1))

A = Mid(A, 2)

b = Mid(b, 2)

If LCase(A) = LCase(b) And Len(A) > 2 Then

If (UCase(A) = A And LCase(b) = b) Or (UCase(b) = b And \_

LCase(A) = A) Then

ActiveDocument.Paragraphs(i).Range.Words(1).HighlightColorIndex = 0

ActiveDocument.Paragraphs(i + 1).Range.Words(1).HighlightColorIndex \_

= 0

End If

End If

If i Mod 50 = 0 Then

DoEvents

StatusBar = spcs & "Final checks: " & totParas - i

DoEvents

End If

Next i

myOnames = ""

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13O[!a-z]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchSoundsLike = False

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rng.Collapse wdCollapseEnd

rng.Expand wdWord

wd = Mid(rng.Text, 3)

rng.Expand wdParagraph

pa = rng.Text

Set rng2 = ActiveDocument.Content

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13" & wd

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

If rng2.Find.Found Then

rng2.Collapse wdCollapseEnd

rng2.Expand wdParagraph

pa2 = rng2.Text

myOnames = myOnames & pa2 & pa & vbCr

End If

rng.Collapse wdCollapseEnd

rng.End = rng.End - 2

rng.Find.Execute

Loop

If myOnames > "" Then

Selection.EndKey Unit:=wdStory

Selection.TypeText "Possible O'<something> errors" & vbCr

Selection.MoveUp , 1

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

Selection.EndKey Unit:=wdStory

Selection.TypeText myOnames

Selection.HomeKey Unit:=wdStory

End If

Set rng = ActiveDocument.Content

finalList.Activate

Selection.EndKey Unit:=wdStory

Selection.TypeText vbCr & vbCr & vbCr

Selection.Paste

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myDummy

.Wrap = wdFindContinue

.Replacement.Text = " "

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^$zzz^$" & leadDots & "1" & vbCr

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Clear clipboard

Set rng = ActiveDocument.Content

rng.End = 2

rng.Copy

Set finalList = ActiveDocument

StatusBar = "Creating queries list"

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Set queriesDoc = ActiveDocument

ActiveDocument.Paragraphs(1).Range.Delete

Set rng = ActiveDocument.Content

rng.Font.StrikeThrough = True

For Each myPara In ActiveDocument.Paragraphs

Set ch = myPara.Range.Characters(1)

chCol = ch.HighlightColorIndex

If chCol > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

myLen = Len(myPara.Range.Text)

If myLen > 4 Then

If chCol > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

Set che = myPara.Range.Characters(myLen - 2)

If che.HighlightColorIndex > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

End If

Next myPara

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = True

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

rng.Font.StrikeThrough = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{3,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p^p"

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

For Each myPara In ActiveDocument.Paragraphs

myText = myPara.Range.Text

If Len(myText) > 4 Then

Set ch = myPara.Range.Characters(1)

numChars = myPara.Range.Characters.count

Set myEnd = myPara.Range.Characters(numChars)

colNum = ch.HighlightColorIndex Mod 8

If ch.Font.Bold = True Then

myTxt = "qcqc " & Str(colNum + 1) & " = zczc"

Else

myTxt = "qcqc zczc"

End If

If ch.Font.Underline > 0 And colNum > 0 Then

myBit = "\* "

myTxt = Replace(myTxt, " = ", "")

Else

myBit = ""

End If

myPara.Range.InsertBefore myBit & myTxt

If ch.Font.Italic = True Then

myEnd.InsertBefore "qpqp= " & Chr(65 + colNum)

End If

End If

i = i + 1

If i Mod 20 = 0 And Len(myText) > 4 Then

myText = Replace(myText, vbCr, "")

StatusBar = spcs & "Creating queries list: " & myText

End If

DoEvents

Next myPara

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\\* qcqc(\*)zczc"

.Wrap = wdFindContinue

.Replacement.Text = "\* \1^t"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "qcqc(\*)zczc"

.Wrap = wdFindContinue

.Replacement.Text = "\1^t"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "qpqp(\*)^13"

.Replacement.Text = "^t\1^p"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "= ^$"

.Replacement.Text = ""

.Replacement.Font.Bold = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

rng.Font.Bold = False

rng.Font.Italic = False

rng.Font.DoubleStrikeThrough = False

rng.Font.Underline = False

rng.Font.Color = wdColorBlack

Selection.HomeKey Unit:=wdStory

Selection.TypeText title2 & CR

Set rng = ActiveDocument.Content.Paragraphs(2).Range

If rng.Text = vbCr Then rng.Delete

Set rng = ActiveDocument.Content.Paragraphs(1).Range

rng.Style = ActiveDocument.Styles(wdStyleHeading1)

StatusBar = " "

Options.DefaultHighlightColorIndex = oldColour

lighterColour = wdGray25

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "= ^$"

.Replacement.Text = ""

.Replacement.Font.ColorIndex = lighterColour

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^# ="

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

DoEvents

End With

Application.ScreenUpdating = True

If doingSeveralMacros = False Then

myTime = (Int(10 \* (Timer - timeStart) / 60) / 10)

Beep

If myTime > 0 Then MsgBox myTime & " minutes"

End If

If InStr(FUT.Name, "zzTestFile") > 0 Then FUT.Activate

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub ProperNounAlyseES()**

' Paul Beverley - Version 30.08.22

' Analyses similar proper nouns

minLengthCheck = 4

includeAcronyms = True

ignoreWords = "Esto Esta Este Estos Estas Esos Esas "

ignoreWords = ignoreWords & " Unos Unas Ellos Ellas Hacia Hasta Para"

ignoreWords = ignoreWords & " Según Pero Entre Tras Como Cuando Donde Porque "

similarChars = "bb,b; b,p; sch,sh; ch,sh; c,k; ph,f; ss,z; s,z;" & \_

" mp,m; ll,l; nn,n; nd,n; nt,n;"

' With non-English languages, you might need to make this False

ignorePlurals = False

myScreenOff = True

ignoreWords = Replace(ignoreWords, " ", " ")

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox(" ProperNounAlyse" & vbCr & vbCr & \_

"Analyse this document?", vbQuestion \_

+ vbYesNoCancel, "ProperNounAlyse")

If myResponse <> vbYes Then Exit Sub

End If

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

myDummy = ChrW(222)

For i = 1 To 100

spcs = " " & spcs

Next i

dummyText = ChrW(197) & "zzzx "

For i = 65 To 90

dummyText = dummyText & ChrW(i) & "zzzz "

Next i

checkFinalLetters = True

' checkFinalLetters = False

' Grey on word only

thisHighlight = wdGray25

doMissingLetter = True

' doMissingLetter = False

' Bold And blue

switchTest = True

' switchTest = False

' double strikethrough

doSimilarLetters = True

' doSimilarLetters = False

' various highlight colours + underline

doVowelTest = True

' doVowelTest = False

' various highlight colours + italic

' These last two tests cycle through these colours:

maxCol = 6

ReDim myCol(maxCol) As Integer

myCol(1) = wdYellow

myCol(2) = wdBrightGreen

myCol(3) = wdTurquoise

myCol(4) = wdRed

myCol(5) = wdPink

myCol(6) = wdGray25

colcode = 0

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = wdGray25

leadDots = " . . . "

title1 = "Proper noun list"

title2 = "Proper noun queries"

CR = vbCr: CR2 = CR & CR

convCharsUC = "AAAAAA..EEEEIIII..OOOOO..UUUU" & \_

"...aaaaaa..eeeeiiiio.ooooo.ouuuu......"

convCharsLC = LCase(convCharsUC)

timeStart = Timer

' collect notes text, if any

endText = ""

footText = ""

If ActiveDocument.Endnotes.Count > 0 Then

endText = ActiveDocument.StoryRanges(wdEndnotesStory).Text

End If

If ActiveDocument.Footnotes.Count > 0 Then

footText = ActiveDocument.StoryRanges(wdFootnotesStory).Text

End If

' collect text in all the textboxes (if any)

sh = ActiveDocument.Shapes.Count

If sh > 0 Then

ReDim shText(sh)

i = 0

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

i = i + 1

shText(i) = shp.TextFrame.TextRange.Text

End If

End If

Next

shCount = i

End If

' Create various documents

Set rng = ActiveDocument.Content

Documents.Add

Set firstDoc = ActiveDocument

Set fnl = ActiveDocument.Content

Documents.Add

Set tempDoc = ActiveDocument

Set tmp = ActiveDocument.Content

Documents.Add

Set allText = ActiveDocument

Selection.TypeText dummyText & vbCr

Selection.FormattedText = rng.FormattedText

Selection.Collapse wdCollapseEnd

' Add notes + shape text

Selection.TypeText endText & CR & footText & CR

If shCount > 0 Then

For i = 1 To shCount

Selection.TypeText shText(i) & CR

Next i

End If

Selection.HomeKey Unit:=wdStory

Set rng = allText.Content

rng.Revisions.AcceptAll

DoEvents

StatusBar = spcs & "Preparing copied file - 1"

DoEvents

' Delete struck-through text

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.MatchWildcards = False

.Font.StrikeThrough = True

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "["

.MatchWildcards = False

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

' Remove strange unicode characters

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & ChrW(&HA000) & "-" & ChrW(&HD6FF) & "]{1,}"

.MatchWildcards = True

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

DoEvents

StatusBar = spcs & "Preparing copied file - 2"

DoEvents

' Cut all and replace as pure text

Set rng = allText.Content

tmp.FormattedText = rng.FormattedText

rng.Text = tmp.Text

tmp.Delete

DoEvents

StatusBar = spcs & "Preparing copied file - 3"

' Use qqq for apostrophe

Set rng = allText.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "n" & ChrW(8217) & "t"

.MatchWildcards = False

.Replacement.Text = "nqqqt"

.Execute Replace:=wdReplaceAll

End With

' Use qq for apostrophe

With rng.Find

.Text = "O'"

.MatchCase = True

.Replacement.Text = "Oqqq"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Find initial cap words

DoEvents

StatusBar = spcs & "Preparing copied file - 4"

DoEvents

myChopNum = minLengthCheck - 2

If myChop < 1 Then myChop = 1

myChop = Trim(Str(myChopNum))

myFind = "<[A-Z][a-z][a-zA-Z]{" & myChop & ",}"

If includeAcronyms = True Then myFind = \_

"<[A-Z][a-zA-Z][a-zA-Z]{" & myChop & ",}"

Set rng = allText.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.MatchWildcards = True

.MatchCase = True

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Replacement.Font.StrikeThrough = True

.Execute Replace:=wdReplaceAll

End With

' Delete all non-strikethrough words

DoEvents

StatusBar = spcs & "Preparing copied file - 5"

DoEvents

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = False

.MatchWildcards = False

.MatchCase = True

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

' Delete the unwanted "proper nouns"

DoEvents

StatusBar = spcs & "Preparing copied file - 6"

igWords = Split(Trim(ignoreWords), " ")

For Each wd In igWords

Set rng = allText.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = wd & "^p"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next wd

StatusBar = spcs & "Sorting whole file"

DoEvents

i = 0

For ch = 65 To 90

For Each myPara In allText.Paragraphs

If Asc(myPara.Range) = ch Then

DoEvents

myPara.Range.Font.StrikeThrough = False

tmp.InsertAfter myPara.Range.Text

End If

Next myPara

tmp.InsertAfter Text:="Zzzzz" & CR

Set rng = tempDoc.Content

rng.Sort SortOrder:=wdSortOrderAscending, CaseSensitive:=True

' delete initial blank line

If Len(tempDoc.Paragraphs(1)) < 3 Then \_

tempDoc.Paragraphs(1).Range.Delete

' Create a frequency for each highlighted word

thisWord = ""

myCount = 0

For Each myPara In tempDoc.Paragraphs

Set rng = myPara.Range.Words(1)

DoEvents

nextWord = rng

If nextWord <> thisWord Then

' This is a new word

If Len(thisWord) > 1 Then

fnl.InsertAfter Text:=thisWord \_

& leadDots & Trim(Str(myCount)) & CR

End If

thisWord = nextWord

myCount = 1

Else

myCount = myCount + 1

End If

If nextWord = "Zzzzz" Then Exit For

i = i + 1:

If i Mod 400 = 4 Then

DoEvents

prmt = Left(thisWord, 1) & " "

prmt = prmt & prmt & prmt & prmt

StatusBar = spcs & \_

"Preparing words for frequency list - " & prmt

DoEvents

End If

Next myPara

' Remove all words except frequency counts

Set rng = tempDoc.Content

rng.Delete

Next ch

' Find any unaccounted-for words, e.g. Ångstrom

For Each myPara In allText.Paragraphs

If myPara.Range.Words(1).Font.StrikeThrough = True Then

tmp.InsertAfter myPara.Range.Text

End If

Next myPara

tempDoc.Close SaveChanges:=False

allText.Close SaveChanges:=False

firstDoc.Activate

' Remove blank lines

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{2,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Resort case insensitively

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

CaseSensitive:=False

' Delete rubbish from top and bottom of list

Do

Set rng = ActiveDocument.Paragraphs(1).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until myLen > 9

Do

lastLine = ActiveDocument.Paragraphs.Count

Set rng = ActiveDocument.Paragraphs(lastLine).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until Len(rng.Text) >= 2

' Word list now has freq. count.

Do

lastLine = ActiveDocument.Paragraphs.Count

Set rng = ActiveDocument.Paragraphs(lastLine).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until Len(rng.Text) >= 2

' Create another copy for doing extra tests

Set rng = ActiveDocument.Content

Documents.Add

Set finalList = ActiveDocument

finalList.Range.Text = rng.Text

Selection.HomeKey Unit:=wdStory

' Prepare data for other tests

numWords = ActiveDocument.Paragraphs.Count

For i = 1 To numWords

aWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(aWord)

thisChar = ChrW(n)

If n > 129 Then

If n >= 217 Then aWord = Replace(aWord, thisChar, "U")

If n >= 210 Then aWord = Replace(aWord, thisChar, "O")

If n >= 204 Then aWord = Replace(aWord, thisChar, "I")

If n >= 200 Then aWord = Replace(aWord, thisChar, "E")

If n >= 192 Then aWord = Replace(aWord, thisChar, "A")

End If

allWords = allWords & aWord

jmp = 100

If i Mod jmp = 1 Then

pq = pq + 1

DoEvents

StatusBar = spcs & \_

"Preparing data for other tests - 1 - " & pq

DoEvents

End If

Next i

' ...for the vowel test below

DoEvents

StatusBar = spcs & "Preparing data for other tests - 2"

DoEvents

noVowelWords = " " & allWords

noVowelWords = Replace(noVowelWords, " A", "\_1")

noVowelWords = Replace(noVowelWords, " E", "\_2")

noVowelWords = Replace(noVowelWords, " I", "\_3")

noVowelWords = Replace(noVowelWords, " O", "\_4")

noVowelWords = Replace(noVowelWords, " U", "\_5")

noVowelWords = Replace(noVowelWords, " Y", "\_6")

For k = 2 To Len(noVowelWords) - 1

thisChar = Mid(noVowelWords, k, 1)

n = AscW(thisChar)

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsLC, n - 191, 1)

If myNewChar <> "." Then noVowelWords = \_

Replace(noVowelWords, thisChar, myNewChar)

End If

Next k

noVowelWords = Replace(noVowelWords, "a", "")

noVowelWords = Replace(noVowelWords, "e", "")

noVowelWords = Replace(noVowelWords, "i", "")

noVowelWords = Replace(noVowelWords, "o", "")

noVowelWords = Replace(noVowelWords, "u", "")

noVowelWords = Replace(noVowelWords, "y", "")

noVowelWords = Replace(noVowelWords, "A", "")

noVowelWords = Replace(noVowelWords, "E", "")

noVowelWords = Replace(noVowelWords, "I", "")

noVowelWords = Replace(noVowelWords, "O", "")

noVowelWords = Replace(noVowelWords, "U", "")

noVowelWords = Replace(noVowelWords, "Y", "")

noVowelWords = Replace(noVowelWords, "\_1", " A")

noVowelWords = Replace(noVowelWords, "\_2", " E")

noVowelWords = Replace(noVowelWords, "\_3", " I")

noVowelWords = Replace(noVowelWords, "\_4", " O")

noVowelWords = Replace(noVowelWords, "\_5", " U")

noVowelWords = Replace(noVowelWords, "\_6", " Y")

' ...for the similar words test

DoEvents

StatusBar = spcs & "Preparing data for other tests - 3"

DoEvents

similarAllWords = " " & LCase(allWords)

similarChars = Replace(similarChars, " ", "")

sChars = Replace(similarChars, " ", "")

Do

commaPos = InStr(sChars, ",")

charWas = Left(sChars, commaPos - 1)

sChars = Mid(sChars, commaPos + 1)

semicolonPos = InStr(sChars, ";")

charNew = Left(sChars, semicolonPos - 1)

sChars = Mid(sChars, semicolonPos + 1)

similarAllWords = Replace(similarAllWords, charWas, charNew)

Loop Until Len(sChars) < 2

' Change all the accented characters to non-accented

DoEvents

StatusBar = spcs & "Preparing data for other tests - 4"

DoEvents

sWd = similarAllWords

For k = 1 To Len(sWd) - 1

thisChar = Mid(sWd, k, 1)

n = AscW(thisChar)

myNewChar = "."

If n > 191 And n < 256 Then

myNewChar = Mid(convCharsLC, n - 191, 1)

If myNewChar <> "." Then sWd = Replace(sWd, \_

thisChar, myNewChar)

End If

Next k

similarAllWords = sWd

' Catch words with only the final two letters the same

i = 0

If checkFinalLetters = True Then

For Each myPara In ActiveDocument.Paragraphs

gotOne = False

myWord = Trim(myPara.Range.Words(1))

myLen = Len(myWord)

If myLen > 6 Then

myTarget = "^p" & Left(myWord, myLen - 2) & "^$^$ "

myCut = 2

Else

myTarget = "^p" & Left(myWord, myLen - 1) & "^$ "

myCut = 1

End If

Set rng = ActiveDocument.Content

rng.Start = myPara.Range.End - 3

rng.Collapse wdCollapseStart

With rng.Find

.Replacement.ClearFormatting

.ClearFormatting

.Text = myTarget

.Replacement.Text = ""

.Forward = True

.MatchCase = True

.MatchWildcards = False

.Wrap = wdFindStop

End With

rng.Find.Execute

Do While rng.Find.Found

gotOne = True

rng.MoveStart 1

rng.End = rng.Start + myLen - myCut

rng.HighlightColorIndex = thisHighlight

rng.Font.Bold = True

rng.Find.Execute

Loop

If gotOne = True Then

Set rng = myPara.Range.Words(1)

rng.End = rng.Start + myLen - myCut

rng.HighlightColorIndex = thisHighlight

rng.Font.Bold = True

End If

i = i + 1

If i Mod 100 = 1 Then

DoEvents

StatusBar = spcs & "Doing test (5) on " & myWord

DoEvents

End If

Next myPara

End If

If doMissingLetter = True Then

' Start of test

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Change the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (4) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' Check if word reappears with one letter missing (1)

For k = 2 To Len(myWord) - 1

testWord = " " & Left(myWord, k - 1) & Mid(myWord, k + 1)

wordPos = InStr(allWords, testWord)

If wordPos > 0 Then

lastLetter = Mid(myWord, Len(myWord) - 1, 1)

' but not "s" at the end, unless it's a spelling error

If lastLetter = "s" Then

ignoreIt = (Application.CheckSpelling(myWord, \_

MainDictionary:=Languages(Selection.LanguageID).NameLocal) \_

= True)

Else

ignoreIt = False

End If

If ignoreIt = False And ignorePlurals = True Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

' mark the pair

leftBit = Left(allWords, InStr(allWords, testWord) \_

+ Len(testWord) - 1)

j = Len(leftBit) - Len(Replace(leftBit, " ", ""))

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Bold = True

rng.Font.Color = wdColorBlue

Set rng = ActiveDocument.Paragraphs(j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Bold = True

rng.Font.Color = wdColorBlue

End If

End If

Next k

If Left(myWord, 2) = "Mc" Or Left(myWord, 3) = "Mac" Or \_

Left(myWord, 3) = "Mag" Then

McList = McList & ActiveDocument.Paragraphs(i).Range

End If

Next i

End If

If doSimilarLetters = True Then

doneWords = ""

doneSimilarWords = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Change the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (3) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check similar spellings: Perutz/Peruts or Chebyshev/Chevychev

similarWord = " " & LCase(myWord)

sChars = similarChars

Do

commaPos = InStr(sChars, ",")

charWas = Left(sChars, commaPos - 1)

sChars = Mid(sChars, commaPos + 1)

semicolonPos = InStr(sChars, ";")

charNew = Left(sChars, semicolonPos - 1)

sChars = Mid(sChars, semicolonPos + 1)

similarWord = Replace(similarWord, charWas, charNew)

Loop Until Len(sChars) < 2

' Change all the accented characters to non-accented

For k = 1 To Len(myWord) - 1

thisChar = Mid(myWord, k, 1)

n = AscW(thisChar)

If n > 191 And n < 256 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

Next k

similarAllWords = Mid(similarAllWords, Len(similarWord))

theseWords = similarAllWords

If InStr(doneSimilarWords, similarWord) = 0 And \_

InStr(theseWords, similarWord) > 0 Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Underline = True

doneSimilarWords = doneSimilarWords & similarWord

' search through all the following words

theseWords = similarAllWords

For j = 1 To numWords - i

spPos = InStr(Trim(theseWords) & " ", " ")

If Left(theseWords, spPos + 1) = similarWord Then

Set rng = ActiveDocument.Paragraphs(i + j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Underline = True

End If

theseWords = Mid(theseWords, spPos + 1)

capThisLetter = Mid(theseWords, 2, 1)

If capThisLetter <> LCase(captestLetters) Then Exit For

Next j

End If

Next i

End If

If switchTest = True Then

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Change the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (2) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check for switched chars

wordLen = Len(myWord) - 1

For k = 1 To Len(myWord) - 3

otherWord = Left(myWord, k) & Mid(myWord, k + 2, 1) & \_

Mid(myWord, k + 1, 1) & Mid(myWord, k + 3)

wordPos = InStr(testWords, otherWord)

If wordPos > 0 Then

' Find the position of the matching word

matchWord = Mid(testWords, wordPos, Len(myWord))

leftBit = Left(allWords, InStr(allWords, matchWord) + 1)

j = Len(leftBit) - Len(Replace(leftBit, " ", "")) + 1

ActiveDocument.Paragraphs(i).Range.Font.DoubleStrikeThrough \_

= True

ActiveDocument.Paragraphs(i).Range.HighlightColorIndex \_

= thisCol

ActiveDocument.Paragraphs(j).Range.Font.DoubleStrikeThrough \_

= True

ActiveDocument.Paragraphs(j).Range.HighlightColorIndex \_

= thisCol

End If

Next k

Next i

End If

If doVowelTest = True Then

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Change the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (1) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check if there's a word with different vowels

otherWord = " " & Replace(myWord, "a", "")

otherWord = Replace(otherWord, "e", "")

otherWord = Replace(otherWord, "i", "")

otherWord = Replace(otherWord, "o", "")

otherWord = Replace(otherWord, "u", "")

otherWord = Replace(otherWord, "y", "")

' Delete all the accented characters

For k = 3 To Len(otherWord) - 1

thisChar = Mid(otherWord, k, 1)

n = AscW(thisChar)

If InStr("AEIOUY", thisChar) > 0 Then

otherWord = Left(otherWord, k - 1) & "=" & Mid(otherWord, k + 1)

Else

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then

otherWord = Replace(otherWord, thisChar, "=")

End If

End If

End If

Next k

otherWord = Replace(otherWord, "=", "")

' otherWord is now the word under test (vowel-less)

otherWord = Replace(otherWord, ".", "")

noVowelWords = Mid(noVowelWords, Len(otherWord))

If Left(noVowelWords, 1) <> " " Then noVowelWords = \_

" " & noVowelWords

theseWords = noVowelWords

wordPos = InStr(noVowelWords, otherWord)

If InStr(doneWords, otherWord) = 0 And wordPos > 0 Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Italic = True

doneWords = doneWords & otherWord

For j = 1 To numWords - i

spPos = InStr(Trim(theseWords) & " ", " ")

firstWord = Left(theseWords, spPos + 1)

theseWords = Mid(theseWords, spPos + 1)

If firstWord = otherWord Then

Set rng = ActiveDocument.Paragraphs(i + j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Italic = True

End If

capThisLetter = Mid(theseWords, 2, 1)

If capThisLetter > "" And capThisLetter <> \_

captestLetters Then Exit For

Next j

End If

Next i

End If

finishOff:

Selection.EndKey Unit:=wdStory

Selection.TypeText CR2 & McList

Selection.HomeKey Unit:=wdStory

Selection.TypeText title1 & CR

Do

Selection.Expand wdParagraph

If Len(Selection) < 3 Or LCase(Selection) = \_

UCase(Selection) Then Selection.Delete

Loop Until LCase(Selection) <> UCase(Selection)

Selection.HomeKey Unit:=wdStory, Extend:=wdExtend

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

' Restore apostrophes

Set rng = finalList.Range

With rng.Find

.Text = "qqq"

.MatchCase = False

.Replacement.Text = "'"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Find first highlight

Set rng = finalList.Content

With rng.Find

.Text = "Zzzzz"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

Set rng = finalList.Content

With rng.Find

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

rng.Select

Selection.Collapse wdCollapseStart

Set finalList = ActiveDocument

firstDoc.Activate

' Find sets of sounds-like words

StatusBar = spcs & "Sounds-like tests"

k = 0

For Each myPara In ActiveDocument.Paragraphs

myWord = Trim(myPara.Range.Words(1))

k = k + 1

If k Mod 40 = 1 Then

DoEvents

StatusBar = spcs & "Sounds-like test: " & myWord

DoEvents

End If

hasAccent = False

For i = 1 To Len(myWord)

ascChar = AscW(Mid(myWord, i))

If ascChar > 128 Or ascChar = Asc("?") Then hasAccent = True

Next i

' Go and find the first sounds-like word

initLetter = Left(myWord, 1)

If Len(myWord) > 2 And myPara.Range.HighlightColorIndex > 0 And \_

hasAccent = False And InStr(allSets, myWord & leadDots) \_

= 0 Then

Set rng = ActiveDocument.Content

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myWord

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = False

.MatchSoundsLike = True

.Execute

End With

Set myPara = rng.Paragraphs(1).Range

rng.Collapse wdCollapseEnd

Loop Until Left(myPara, 1) = initLetter

setOfWords = myPara

gottaSet = False

rng.Collapse wdCollapseEnd

rng.Find.Execute

Do While rng.Find.Found = True

Set myPara = rng.Paragraphs(1).Range

If Left(myPara, 1) = initLetter Then

gottaSet = True

setOfWords = setOfWords & myPara

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

If gottaSet = True Then allSets = allSets & setOfWords & CR

End If

Next myPara

Selection.WholeStory

If Len(allSets) < 2 Then

Selection.TypeText "None found with this test"

Else

Selection.TypeText allSets

End If

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Proper nouns by sound" & CR

Selection.HomeKey Unit:=wdStory, Extend:=wdExtend

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = 0

rng.Copy

ActiveDocument.Close SaveChanges:=False

finalList.Activate

' Remove highlighting from second half of words

' that are only case changes of one another

totParas = ActiveDocument.Paragraphs.Count

For i = 1 To totParas - 1

A = Trim(ActiveDocument.Paragraphs(i).Range.Words(1))

b = Trim(ActiveDocument.Paragraphs(i + 1).Range.Words(1))

A = Mid(A, 2)

b = Mid(b, 2)

If LCase(A) = LCase(b) And Len(A) > 2 Then

If (UCase(A) = A And LCase(b) = b) Or (UCase(b) = b And \_

LCase(A) = A) Then

ActiveDocument.Paragraphs(i).Range.Words(1).HighlightColorIndex = 0

ActiveDocument.Paragraphs(i + 1).Range.Words(1).HighlightColorIndex \_

= 0

End If

End If

If i Mod 50 = 0 Then

DoEvents

StatusBar = spcs & "Final checks: " & totParas - i

DoEvents

End If

Next i

myOnames = ""

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13O[!a-z]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchSoundsLike = False

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rng.Collapse wdCollapseEnd

rng.Expand wdWord

wd = Mid(rng.Text, 3)

rng.Expand wdParagraph

pa = rng.Text

Set rng2 = ActiveDocument.Content

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13" & wd

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

If rng2.Find.Found Then

rng2.Collapse wdCollapseEnd

rng2.Expand wdParagraph

pa2 = rng2.Text

myOnames = myOnames & pa2 & pa & vbCr

End If

rng.Collapse wdCollapseEnd

rng.End = rng.End - 2

rng.Find.Execute

Loop

If myOnames > "" Then

Selection.EndKey Unit:=wdStory

Selection.TypeText "Possible O'<something> errors" & vbCr

Selection.MoveUp , 1

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

Selection.EndKey Unit:=wdStory

Selection.TypeText myOnames

Selection.HomeKey Unit:=wdStory

End If

Set rng = ActiveDocument.Content

finalList.Activate

Selection.EndKey Unit:=wdStory

Selection.TypeText vbCr & vbCr & vbCr

Selection.Paste

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myDummy

.Wrap = wdFindContinue

.Replacement.Text = " "

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^$zzz^$" & leadDots & "1" & vbCr

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Clear clipboard

Set rng = ActiveDocument.Content

rng.End = 2

rng.Copy

Set finalList = ActiveDocument

StatusBar = "Creating queries list"

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Set queriesDoc = ActiveDocument

ActiveDocument.Paragraphs(1).Range.Delete

Set rng = ActiveDocument.Content

rng.Font.StrikeThrough = True

For Each myPara In ActiveDocument.Paragraphs

Set ch = myPara.Range.Characters(1)

chCol = ch.HighlightColorIndex

If chCol > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

myLen = Len(myPara.Range.Text)

If myLen > 4 Then

If chCol > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

Set che = myPara.Range.Characters(myLen - 2)

If che.HighlightColorIndex > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

End If

Next myPara

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = True

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

rng.Font.StrikeThrough = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{3,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p^p"

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

For Each myPara In ActiveDocument.Paragraphs

myText = myPara.Range.Text

If Len(myText) > 4 Then

Set ch = myPara.Range.Characters(1)

numChars = myPara.Range.Characters.Count

Set myEnd = myPara.Range.Characters(numChars)

colNum = ch.HighlightColorIndex Mod 8

If ch.Font.Bold = True Then

myTxt = "qcqc " & Str(colNum + 1) & " = zczc"

Else

myTxt = "qcqc zczc"

End If

If ch.Font.Underline > 0 And colNum > 0 Then

myBit = "\* "

myTxt = Replace(myTxt, " = ", "")

Else

myBit = ""

End If

myPara.Range.InsertBefore myBit & myTxt

If ch.Font.Italic = True Then

myEnd.InsertBefore "qpqp= " & Chr(65 + colNum)

End If

End If

i = i + 1

If i Mod 20 = 0 And Len(myText) > 4 Then

myText = Replace(myText, vbCr, "")

StatusBar = spcs & "Creating queries list: " & myText

End If

DoEvents

Next myPara

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\\* qcqc(\*)zczc"

.Wrap = wdFindContinue

.Replacement.Text = "\* \1^t"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "qcqc(\*)zczc"

.Wrap = wdFindContinue

.Replacement.Text = "\1^t"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "qpqp(\*)^13"

.Replacement.Text = "^t\1^p"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "= ^$"

.Replacement.Text = ""

.Replacement.Font.Bold = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

rng.Font.Bold = False

rng.Font.Italic = False

rng.Font.DoubleStrikeThrough = False

rng.Font.Underline = False

rng.Font.Color = wdColorBlack

Selection.HomeKey Unit:=wdStory

Selection.TypeText title2 & CR

Set rng = ActiveDocument.Content.Paragraphs(2).Range

If rng.Text = vbCr Then rng.Delete

Set rng = ActiveDocument.Content.Paragraphs(1).Range

rng.Style = ActiveDocument.Styles(wdStyleHeading1)

StatusBar = " "

Options.DefaultHighlightColorIndex = oldColour

lighterColour = wdGray25

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "= ^$"

.Replacement.Text = ""

.Replacement.Font.ColorIndex = lighterColour

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^# ="

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

DoEvents

End With

If doingSeveralMacros = False Then

Debug.Print Timer - timeStart

myTime = (Int(10 \* (Timer - timeStart) / 60) / 10)

Beep

If myTime > 0 Then MsgBox myTime & " minutes"

Else

FUT.Activate

End If

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub SpecialWordSpellAlyse()**

' Paul Beverley - Version 01.12.20

' Does a ProperNounAlyse of all long 'spelling error' words

minLen = 5

Selection.Collapse wdCollapseStart

thisLanguage = Selection.LanguageID

Select Case thisLanguage

Case wdEnglishUK: myLang = "UK spelling"

Case wdEnglishUS: myLang = "US spelling"

Case wdEnglishCanadian: myLang = "Canadian spelling"

Case Else: myLang = "unknown language"

End Select

myLang = "Using " & myLang & " dictionary. OK?"

If doingSeveralMacros = False Then

myResponse = MsgBox(myLang, vbQuestion + vbYesNoCancel, \_

"SpecialWordSpellAlyse")

If myResponse <> vbYes Then Exit Sub

End If

langName = Languages(thisLanguage).NameLocal

Set rng = ActiveDocument.Content

Documents.Add

Set testDoc = ActiveDocument

Selection.Text = rng.Text

Set rng = ActiveDocument.Content

rng.Case = wdLowerCase

rng.Font.StrikeThrough = True

myEnd = rng.End

For Each wd In rng.Words

If Len(Trim(wd)) > (minLen - 1) Then

DoEvents

If Application.CheckSpelling(wd, MainDictionary:=langName) = False Then

pCent = Int((myEnd - wd.End) / myEnd \* 100)

' Report progress

StatusBar = "Generating errors list. To go: " \_

& Trim(Str(pCent)) & "%"

Debug.Print "Generating errors list. To go: " \_

& Trim(Str(pCent)) & "%"

wd.Font.StrikeThrough = False

wd.Characters(1) = UCase(wd.Characters(1))

End If

End If

DoEvents

Next wd

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Replacement.Text = " "

.Forward = True

.Font.StrikeThrough = True

.MatchCase = False

.Execute Replace:=wdReplaceAll

End With

Call ProperNounAlyse

testDoc.Close SaveChanges:=False

End Sub

**Sub FullNameAlyse()**

' Paul Beverley - Version 24.02.21

' Creates a frequency list of all full names

IncludeNamesWithInitials = vbYes

' In this list, make sure every word has a space after it

allowAbbrevs = "Mr. Mrs. Dr."

nonoWords = "About After Although An And Any As At Before Because " & \_

"But By For Has Have However If In Is Like My Since So Some " & \_

"That The Then These This Those Though Through Unlike " & \_

"Was We What When While Who Why Yet "

nonoWords2 = "an and are do no nor on one or v "

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

myResponse = IncludeNamesWithInitials

If doingSeveralMacros = False Then

myResponse = MsgBox("Include names with initials?", vbQuestion \_

+ vbYesNoCancel, "FullNameAlyse")

If myResponse = vbCancel Then Exit Sub

End If

Set rng = ActiveDocument.Content

Documents.Add

Set originalDoc = ActiveDocument

Selection.FormattedText = rng.FormattedText

' Now prepare the text

numberCmnts = ActiveDocument.Comments.Count

If numberCmnts > 0 Then ActiveDocument.DeleteAllComments

Set rng = ActiveDocument.Content

myEnd = rng.End

' Make apostrophes straight

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8217)

.Wrap = wdFindContinue

.Replacement.Text = "'"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "'s"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

thisArray = Split(Trim(allowAbbrevs), " ")

For i = 0 To UBound(thisArray)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = thisArray(i)

.Wrap = wdFindContinue

.Replacement.Text = Replace(thisArray(i), ".", "")

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

Documents.Add

CR = vbCr

' First mark all two-word proper nouns, in order

' to detect four-word names (= two + two)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z\-']{1,} [A-Z][a-zA-Z\-']{1,}?"

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Font.DoubleStrikeThrough = True

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Find four-word names

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = False

.Font.DoubleStrikeThrough = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Set firstDoc = ActiveDocument

Do While rng.Find.Found = True

numWords = rng.Words.Count

If numWords > 2 And numWords < 7 Then

myText = Left(rng.Text, Len(rng.Text) - 1)

Selection.TypeText myText & CR

rng.Font.Shadow = True

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Find three-word names

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z\-']{1,} [A-Z][a-zA-Z\-']{1,} [A-Z][a-zA-Z\-']{1,}"

.Font.StrikeThrough = False

.Font.Shadow = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

CR = vbCr

Set firstDoc = ActiveDocument

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Font.Shadow = True

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Find three-word names with van, von, der, de etc

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z]{1,} [A-Z][a-zA-Z]{1,} [vanderol]{1,} [A-Z][a-zA-Z]{1,}"

.Font.StrikeThrough = False

.Font.Shadow = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

CR = vbCr

Set firstDoc = ActiveDocument

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Font.Shadow = True

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

rng.Start = 0

rng.End = 0

' Two-word names

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z\-']{1,} [A-Z][a-zA-Z\-']{1,}"

.Font.StrikeThrough = False

.Font.Shadow = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

rng.Start = 0

rng.End = 0

' Two-word names with van, von, der, de etc

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z]{1,} [vanderol]{1,} [A-Z][a-zA-Z]{1,}>"

.Font.StrikeThrough = False

.Font.Shadow = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

If myResponse = vbYes Then

' Find such as P.E. Beverley

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z.]{1,} [A-Z][a-zA-Z]{1,}>"

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Find such as Paul E. Beverley

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z]{1,} [A-Z.]{1,} [A-Z][a-zA-Z]{1,}>"

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Find such as P E Beverley

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z ]{1,} [A-Z][a-zA-Z]{1,}>"

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Find such as Paul E H Beverley

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z]{1,} [A-Z ]{1,} [A-Z][a-zA-Z]{1,}>"

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Find such as P.E. Beverley + van der etc

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z.]{1,} [vanderol]{1,} [A-Z][a-zA-Z]{1,}>"

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Find such as Paul E. Beverley + van der etc

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z]{1,} [A-Z.]{1,} [vanderol]{1,} [A-Z][a-zA-Z]{1,}>"

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Find such as P E Beverley + van der etc

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z ]{1,} [vanderol]{1,} [A-Z][a-zA-Z]{1,}>"

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Find such as Paul E H Beverley + van der etc

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z]{1,} [A-Z ]{1,} [vanderol]{1,} [A-Z][a-zA-Z]{1,}>"

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Selection.TypeText rng.Text & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Find such as Beverley, P.E.

rng.Start = 0

rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z]{1,}, [A-Z. ]{1,}>"

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

nameInits = rng.Text

commaPos = InStr(nameInits, ",")

initsName = Mid(nameInits, commaPos - 1) & " " & Left(nameInits, commaPos - 1)

Selection.TypeText initsName & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

End If

rng.Start = 0

rng.End = myEnd

rng.Font.Shadow = False

rng.Font.DoubleStrikeThrough = False

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeText CR

Selection.HomeKey Unit:=wdStory

Selection.MoveEnd , 1

Selection.Delete

Dim myName(8000) As String

Dim itemCount As Long

Dim myCount As Integer

Dim thisPara As String

Dim prevPara As String

myCount = 0

prevName = ""

For Each myPara In ActiveDocument.Paragraphs

thisPara = Replace(myPara.Range.Text, CR, "")

If thisPara <> prevPara And prevPara <> "" Then

itemCount = itemCount + 1

myName(itemCount) = prevPara & vbTab & Trim(Str(myCount))

myCount = 1

Else

myCount = myCount + 1

End If

prevPara = thisPara

DoEvents

Next myPara

Documents.Add

Set secondDoc = ActiveDocument

For i = 1 To itemCount

If UCase(myName(i)) <> myName(i) Then

Selection.TypeText myName(i) & CR

End If

DoEvents

Next i

maxLine = ActiveDocument.Paragraphs.Count

nonoWords = nonoWords & " "

For i = maxLine To 1 Step -1

firstWord = ActiveDocument.Paragraphs(i).Range.Words(1)

DeleteIt = (InStr(nonoWords, firstWord) > 0)

For j = 2 To ActiveDocument.Paragraphs(i).Range.Words.Count - 1

thisWord = Trim(ActiveDocument.Paragraphs(i).Range.Words(j))

If InStr(nonoWords2, thisWord & " ") > 0 Then DeleteIt = True

Next j

If DeleteIt = True Then ActiveDocument.Paragraphs(i).Range.Delete

Next i

totalItems = ActiveDocument.Paragraphs.Count - 1

' Copy the list and paste into the first document

' as a place to manipulate it

Selection.WholeStory

Selection.Copy

firstDoc.Activate

Selection.WholeStory

Selection.Delete

Selection.Paste

' Move the surname to the beginning of the line

For Each myPara In ActiveDocument.Paragraphs

If Len(myPara.Range.Text) > 2 Then

surnamePosn = myPara.Range.Words.Count - 3

If InStr(myPara.Range.Text, "-") = 0 Then

Surname = Trim(myPara.Range.Words(surnamePosn))

myPara.Range.Words(surnamePosn) = ""

myPara.Range.Words(1) = Surname & ", " & myPara.Range.Words(1)

Else

myPara.Range.Words(surnamePosn).Select

Selection.MoveStartUntil cset:=" ", Count:=wdBackward

Selection.MoveStart , -1

fullSurname = Trim(Selection.Text)

Selection.Delete

Selection.HomeKey Unit:=wdLine

Selection.TypeText fullSurname & ", "

asdgfdfg = 0

End If

End If

DoEvents

Next myPara

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " ^t"

.Replacement.Text = "^t"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Format the list

Selection.HomeKey Unit:=wdStory

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.HomeKey Unit:=wdStory

Selection.MoveEnd , 2

Selection.Delete

Selection.TypeText "Fullname List" & vbCr & vbCr

Selection.TypeText "Sorted by last name" & vbCr

startTable = Selection.End

ActiveDocument.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading2)

ActiveDocument.Paragraphs(3).Style = ActiveDocument.Styles(wdStyleHeading2)

Selection.Start = startTable

Selection.End = ActiveDocument.Range.End

Selection.ConvertToTable Separator:=wdSeparateByTabs

Selection.Tables(1).AutoFitBehavior (wdAutoFitContent)

Selection.Tables(1).Borders(wdBorderTop).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderLeft).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderBottom).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderRight).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderVertical).LineStyle = wdLineStyleNone

Selection.WholeStory

Selection.Copy

ActiveDocument.Close SaveChanges:=False

' Format other list

secondDoc.Activate

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Sorted by first name" & vbCr

startTable = Selection.End

ActiveDocument.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading2)

Selection.Start = startTable

Selection.End = ActiveDocument.Range.End

Selection.ConvertToTable Separator:=wdSeparateByTabs

Selection.Tables(1).AutoFitBehavior (wdAutoFitContent)

Selection.Tables(1).Borders(wdBorderTop).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderLeft).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderBottom).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderRight).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderVertical).LineStyle = wdLineStyleNone

' Copy the other list in here

Selection.HomeKey Unit:=wdStory

Selection.Paste

Selection.HomeKey Unit:=wdStory

' Dummy copy to clear clipboard

Set rng = ActiveDocument.Content

rng.End = rng.Start + 1

rng.Copy

originalDoc.Activate

ActiveDocument.Close SaveChanges:=False

If doingSeveralMacros = False Then

Beep

MsgBox (Str(totalItems) & " names found")

Else

FUT.Activate

End If

End Sub

**Sub CapitAlyse()**

**'** Paul Beverley - Version 21.06.24

' Analyses capitalised words

ignoreHeadings = True

' ignoreHeadings = False

ignoreWords = ",After,All,Although,Also,An,And,As,At,By,For,From,If,In,It,"

ignoreWords = ignoreWords & "Of,On,Our,The,This,Those,There,These,They,Up,We,"

timeStart = Timer

showTime = True

mySpeed = 50

myScreenOff = True

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

' Count case after hyphenation

myTot = ActiveDocument.Range.End

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "([A-Z][a-z]{1,}-[A-Z][a-z]{1,})"

.Replacement.Text = "^&!"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

numCaps = ActiveDocument.Range.End - myTot

If numCaps > 0 Then WordBasic.EditUndo

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "([A-Z][a-z]{1,}-[a-z]{1,})"

.Replacement.Text = "^&!"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

numNoCaps = ActiveDocument.Range.End - myTot

If numNoCaps > 0 Then WordBasic.EditUndo

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ": "

.Wrap = wdFindContinue

.Replacement.Text = ". "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = """"

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[.]{2,}"

.Replacement.Text = "."

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "(Figure [0-9]{1,}.[0-9]{1,})"

.Replacement.Text = "\1. "

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "(Fig. [0-9]{1,}.[0-9]{1,})"

.Replacement.Text = "\1. "

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[^32]{1,}^13"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^13[0-9.\)^t^32" & ChrW(8211) & "]{1,}([A-Z])"

.Replacement.Text = "^p\1"

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^13[a-z][.\)\(^t^32" & ChrW(8211) & "]{1,}([A-Z])"

.Replacement.Text = "^p\1"

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^t"

.Wrap = wdFindContinue

.Replacement.Text = ". "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = ""

.Wrap = wdFindContinue

.Font.StrikeThrough = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

StatusBar = "Preparing the text for searching..."

' Underline headings to potentially ignore them

If ignoreHeadings = True Then

maxWdsInHeading = 20

For Each pa In ActiveDocument.Paragraphs

myText = pa.Range.Text

wdsCount = pa.Range.Words.Count

If Len(myText) > 3 And wdsCount < maxWdsInHeading Then

ch = Mid(myText, Len(myText) - 1, 1)

If InStr(".:", ch) = 0 Then pa.Range.Font.Underline = True

End If

i = i + 1: If i Mod mySpeed = 0 Then DoEvents

Next pa

End If

For Each se In ActiveDocument.Sentences

If Len(se) > 4 Then

If InStr("""'(" & ChrW(8216) & ChrW(8220), \_

Trim(se.Words(1))) = 0 Then

se.Words(1).Font.Underline = True

Else

se.Words(2).Font.Underline = True

End If

i = i + 1: If i Mod 500 = 0 Then DoEvents

End If

Next se

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][a-zA-Z]{1,}"

.Font.Underline = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

myBars = "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"

allWords = "," & ignoreWords & ","

myResult = ""

Set tst = ActiveDocument.Content

myTot = tst.End

Do While rng.Find.Found = True

endNow = rng.End

If InStr(allWords, rng) = 0 Then

StatusBar = myBars & myBars & myExtra & \_

" >>> " & Int((myTot - endNow) / 1000)

testWd = rng.Text

allWords = allWords & testWd & ","

lc = LCase(testWd)

With tst.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = lc

.MatchCase = True

.Replacement.Text = "^&!"

.MatchWildcards = False

.MatchWholeWord = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

numLC = ActiveDocument.Range.End - myTot

If numLC > 0 Then

WordBasic.EditUndo

With tst.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = testWd

.MatchCase = True

.Replacement.Text = "^&!"

.Execute Replace:=wdReplaceAll

End With

i = i + 1: If i Mod 20 = 0 Then DoEvents

numCapAll = ActiveDocument.Range.End - myTot

If numCapAll > 0 Then WordBasic.EditUndo

With tst.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = testWd

.Replacement.Text = "^&!"

.Font.Underline = True

.Execute Replace:=wdReplaceAll

End With

If i Mod 20 = 0 Then DoEvents

numCapStart = ActiveDocument.Range.End - myTot

numCapMid = numCapAll - numCapStart

myExtra = lc & " . ." & Str(numLC) & "\_\_\_\_:\_\_\_\_"

myExtra = myExtra & testWd & " . ." & Str(numCapMid)

If numCapStart > 0 Then

WordBasic.EditUndo

myExtra = myExtra & " (+" & Trim(Str(numCapStart)) & ")"

End If

myResult = myResult & myExtra & ":" & vbCr

myCount = myCount + 1

End If

rng.start = endNow

rng.End = endNow

End If

rng.Find.Execute

Loop

Selection.WholeStory

Selection.TypeText myResult

Selection.WholeStory

Selection.Range.Style = ActiveDocument.Styles(wdStyleNormal)

Selection.Font.Reset

Selection.Sort

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ":"

.Replacement.Text = vbCr

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\_"

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Selection.MoveEndWhile cset:=vbCr, Count:=wdForward

Selection.Delete

Selection.TypeText "Capitalisation" & vbCr & vbCr

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\(\*\)"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Color = wdColorGray50

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

If numCaps + numNoCaps > 0 Then

Selection.EndKey Unit:=wdStory

myText = vbCr & vbCr & "Lowercase after hyphen (Non-linear): " & \_

Trim(Str(numNoCaps)) & vbCr & \_

"Uppercase after hyphen (Non-Linear): " & Trim(Str(numCaps)) \_

& vbCr & vbCr

Selection.TypeText Text:=myText

Selection.HomeKey Unit:=wdStory

End If

Application.ScreenUpdating = True

If doingSeveralMacros = False Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

totTime = Timer - timeStart

If showTime = True Then \_

MsgBox ((Int(10 \* totTime / 60) / 10) & \_

" minutes") & vbCr & vbCr & "Word pairs: " & myCount

Else

FUT.Activate

End If

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub CatchPhrase()**

' Paul Beverley - Version 18.05.18

' Searches for repeated phrases/sentences

group\_a = "25, 6(4), 5(8)"

group\_a = "4(4), 3(9)"

group\_b = "6(3), 5(8), 4(10)"

group\_c = "7(3), 6(5), 5(10), 4(15)"

' Number of spaces times n

mySpaces = " ": n = 20

highlightFinds = False

highlightFinds = True

myColour = wdYellow

goExtraFast = True

goExtraFast = False

giveSpeedWarning = True

stopAndShowTime = False

myDots = ".... "

If Application.Visible = False Then Application.Visible \_

= True: Exit Sub

myLap = 1

Set rng = ActiveDocument.Content

rng.End = 200

If LCase(rng.Text) <> rng.Text Then

If giveSpeedWarning = True Then

myResponse = MsgBox("Preparing words file. This may take some time." & vbCr \_

& vbCr & "Please ignore any ""Not Responding"" warnings." \_

& vbCr & vbCr & "Click Yes to start.", vbQuestion \_

+ vbYesNo, "WordsPhrasesInContext")

If myResponse <> vbYes Then Exit Sub

Else

StatusBar = "Preparing words file. This may take some time."

End If

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = LCase(rng.Text)

For i = 1 To 6

sps = sps & " "

Next i

' Remove all except pure text, hyphens and apostrophes

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "'"

.Wrap = wdFindContinue

.Replacement.Text = "jqjq"

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

StatusBar = sps & "Preparing words file. This may take some time. Six..."

DoEvents

.Text = "[!a-zA-Z,\- ]"

.MatchWildcards = True

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

DoEvents

StatusBar = sps & "Preparing words file. This may take some time. Five..."

DoEvents

.Text = " [ ,-]{1,}"

.Replacement.Text = " "

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

StatusBar = sps & "Preparing words file. This may take some time. Four..."

DoEvents

.Text = ","

.Replacement.Text = "cmcm"

.Execute Replace:=wdReplaceAll

DoEvents

StatusBar = sps & "Preparing words file. This may take some time. Three.."

DoEvents

.MatchWildcards = True

.Text = "-{1,}"

.Replacement.Text = "cqcq"

.Execute Replace:=wdReplaceAll

DoEvents

StatusBar = sps & "Preparing words file. This may take some time. Two..."

DoEvents

.Text = " [a-hj-z] "

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

DoEvents

StatusBar = sps & "Preparing words file. This may take some time. One!"

DoEvents

.Text = " {2,}"

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

End If

CR = vbCr: CR2 = CR & CR

For j = 1 To n

sps = sps & mySpaces

Next j

myPrompt = "a = " & group\_a & CR2

myPrompt = myPrompt & "b = " & group\_b & CR2

myPrompt = myPrompt & "c = " & group\_c & CR2 & CR

myPrompt = myPrompt & "t = Test to estimate the ETA" & CR2

Do

myChoice = InputBox(myPrompt, "CatchPhrase", "a")

If myChoice = "" Then Beep: Exit Sub

Loop Until InStr("abct", myChoice) > 0 \_

Or InStr("123456789", Left(myChoice, 1)) > 0

Set resultsDoc = ActiveDocument

Set rng0 = ActiveDocument.Content

Documents.Add

Set wordsDoc = ActiveDocument

Set rng = wordsDoc.Content

rng.Text = LCase(rng0.Text)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "@@@@@"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute

End With

If rng.Find.Found Then

rng.End = wordsDoc.Content.End

rng.Delete

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " {2,}"

.MatchWildcards = True

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Dim numWds(20) As Integer

Dim numShows(20) As Integer

Dim myList(4) As String

Dim myCount As Integer

myList(1) = group\_a

myList(2) = group\_b

myList(3) = group\_c

myList(4) = "5,"

If LCase(myChoice) <> UCase(myChoice) Then

If myChoice = "t" Then

myWdsList = "5,"

jumpFwd = 200

ActiveDocument.Words(jumpFwd).Select

Selection.Collapse wdCollapseStart

Selection.TypeText Text:="this is pauljqjqs speed " \_

& "test rhubarb this is pauljqjqs speed test "

Else

myWdsList = myList(Asc(myChoice) - 96)

myWdsList = Replace(myWdsList, " ", "") & ","

myWdsList = Replace(myWdsList, ",,", ",")

End If

Else

myWdsList = myChoice & ","

End If

myRun = Split(myWdsList, ",")

numRuns = UBound(myRun) - 1

myOutput = ""

st0 = Timer

If goExtraFast = True Then Application.Visible = False

For j = 0 To numRuns

Selection.HomeKey Unit:=wdStory

myTask = myRun(j)

' Search for phrases

myPhrases = ""

shownOne = False

totWds = wordsDoc.Words.Count

phrLen = Val(myTask)

myMinWds = 2

bktPos = InStr(myTask, "(")

myTask = Mid(myTask, bktPos + 1)

If bktPos > 0 Then

myMinWds = Val(myTask)

ignoreSubPhrases = False

Else

ignoreSubPhrases = True

End If

tstPhrase = ""

For n = 1 To phrLen

tstPhrase = tstPhrase & "dummy "

Next n

i = 1

st = Timer

myPrompt = "No duplicate phrases found yet" \_

& " Wds: " & phrLen & "(" & myMinWds & ")"

displayPhrase = myPrompt

For Each wd In wordsDoc.Words

spPos = InStr(tstPhrase, " ")

tstPhrase = Mid(tstPhrase, spPos + 1) & wd.Text

i = i + 1

nw = Timer

pc = Str(Int(1000 \* i / totWds) / 10)

If InStr(pc, ".") = 0 Then pc = pc & ".0"

StatusBar = sps & pc & "% " & phrLen & "(" & \_

myMinWds & ")" & " ETA " & predictedTime

If InStr(myPhrases, tstPhrase & myDots) = 0 Then

DoEvents

Set rng = wordsDoc.Content

' Find the first one

tstLen = Len(tstPhrase)

If tstLen > 255 Then tstPhrase = Left(tstPhrase, 254)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = tstPhrase

If ignoreSubPhrases = True Then .Font.Underline = False

.Replacement.Text = ""

.MatchWildcards = False

End With

phrFreq = -1

Do

phrFreq = phrFreq + 1

rng.Find.Execute

rng.Collapse wdCollapseEnd

Loop Until rng.Find.Found = False

If phrFreq > 1 Then

newPhrase = tstPhrase & myDots & Trim(Str(phrFreq))

displayPhrase = Replace(newPhrase, "cmcm", ",")

displayPhrase = Replace(displayPhrase, "cqcq", "-")

displayPhrase = Replace(displayPhrase, "jqjq", "'")

DoEvents

ActiveDocument.ActiveWindow.Caption = \_

sps & "LATEST FIND: " & displayPhrase & sps & sps

myPrompt = " Wds: " & phrLen & "(" & myMinWds & ")"

timeToGo = (nw - st) \* (totWds - i) / i

myTime = Time

myETA = DateAdd("s", timeToGo, myTime)

predictedTime = Left(myETA, 5)

If myChoice = "t" Then

MsgBox "ETA: " & predictedTime & " = " & \_

Int(timeToGo / 6) / 10 & " min"

wordsDoc.Close SaveChanges:=False

Exit Sub

End If

myPrompt = myPrompt & " ETA " & predictedTime

If i > totWds Then

ahfkjhasdkjgf = 0

End If

pc = Str(Int(1000 \* i / totWds) / 10)

If InStr(pc, ".") = 0 Then pc = pc & ".0"

spd = Str(Int(10 \* i / (nw - st)) / 10)

If InStr(spd, ".") = 0 Then spd = spd & ".0"

Debug.Print spd & " " & pc & "% " & myPrompt & \_

" " & displayPhrase

myPhrases = myPhrases & newPhrase & vbCr

If phrFreq > myMinWds - 1 Then

myOutput = myOutput & newPhrase & vbCr

If highlightFinds Then

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

With rng0.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = tstPhrase

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

End If

End If

DoEvents

End If

End If

Next wd

myOutput = Replace(myOutput, "cqcq", "-")

myOutput = Replace(myOutput, "cmcm", ",")

myOutput = Replace(myOutput, "jqjq", ChrW(8217))

rng0.InsertAfter Text:=vbCr & "@@@@@@@@@@@@@@@@@@@@@ " & \_

ChrW(8211) & " " & phrLen & vbCr & myOutput & vbCr

myPhrases = ""

myOutput = ""

t = Timer - st

If t > 600 Then

ttot = Int(t / 6) / 10

tText = Str(ttot) & " min"

Else

ttot = Int(t \* 10) / 10

tText = Str(ttot) & " sec"

End If

myResult = "Ave wds/sec: " & Int(10 \* i / t) / 10 & vbCr & vbCr

myResult = myResult & "Time: " & tText

rng0.InsertAfter Text:=vbCr & "================== " & vbCr \_

& myResult & vbCr

If stopAndShowTime = True Then

Application.Visible = True

MsgBox myResult

End If

Next j

StatusBar = " "

t = Timer - st0

If t > 600 Then

ttot = Int(t / 6) / 10

tText = Str(ttot) & " min"

Else

ttot = Int(t \* 10) / 10

tText = Str(ttot) & " sec"

End If

myResult = "Ave wds/sec: " & Int(10 \* i \* (numRuns + 1) / t) / 10 \_

& vbCr & vbCr

myResult = myResult & "Total time: " & tText

rng0.InsertAfter Text:=vbCr & "================== " & vbCr \_

& "================== " & vbCr & myResult & vbCr

Application.Visible = True

Beep

ActiveDocument.ActiveWindow.Caption = ""

MsgBox myResult

wordsDoc.Close SaveChanges:=False

resultsDoc.Activate

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "@@@@@"

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

Set rng = Selection.range

rng.End = ActiveDocument.Content.End

rng.HighlightColorIndex = wdNoHighlight

Selection.Collapse wdCollapseStart

Application.StatusBar = False

Beep

End Sub

**Sub StyleEffectDetector()**

' Paul Beverley - Version 10.09.19

' Reports the style and effects applied to the text

CR2 = vbCr & vbCr

Do

If Selection.Start = Selection.End Then Selection.MoveEnd , 1

myRes = ""

mixed = 9999999

noPattern = -16777216

Set sel = Selection.range

myTest = sel.Text

crPos = InStr(myTest, vbCr)

If crPos < Len(myTest) And crPos <> 0 Then

MsgBox "Please select some text WITHIN a paragraph and rerun the macro."

Exit Sub

End If

sty = sel.Style

If sty <> "Normal" Then \_

myRes = myRes & "Style: " & sel.Style & CR2

styBold = sel.Style.Font.Bold

styItalic = sel.Style.Font.Italic

stySize = sel.Style.Font.Size

styColour = sel.Style.Font.ColorIndex

col = sel.Font.ColorIndex

hicol = sel.HighlightColorIndex

If col = mixed Then

myRes = myRes & "Mixed font colours" & CR2

Else

If col > 0 Then

If col = styColour Then

myRes = myRes & "Font colour: " & col & \_

" (style)" & CR2

Else

myRes = myRes & "Font colour: " \_

& col & " (added)" & CR2

End If

End If

End If

If hicol = mixed Then

myRes = myRes & "Mixed highlight colours" & CR2

Else

If hicol > 0 Then myRes = myRes & "Highlight colour: " \_

& hicol & CR2

End If

If sel.Shading.Texture <> wdTextureNone Then

myRes = myRes & "Shading.Texture: " & sel.Shading.Texture & CR2

End If

fore = sel.Shading.ForegroundPatternColor

If fore <> noPattern Then

myRes = myRes & "Shading.ForegroundPatternColor: " & \_

Hex(fore) & CR2

End If

bck = sel.Shading.BackgroundPatternColor

If bck <> noPattern Then

myRes = myRes & "Shading.backgroundPatternColor: " & \_

Hex(bck) & CR2

End If

If sel.Bold = mixed Then myRes = myRes & "Mixed bold " & CR2

If sel.Bold = True Then

If styBold = True Then

myRes = myRes & "Bold (style)" & CR2

Else

myRes = myRes & "Bold (added)" & CR2

End If

Else

If sel.Bold = False And styBold Then

myRes = myRes & "Bold removed" & CR2

End If

End If

If sel.Italic = mixed Then myRes = myRes & \_

"Mixed italic " & CR2

If sel.Italic = True Then

If styItalic = True Then

myRes = myRes & "Italic (style)" & CR2

Else

myRes = myRes & "Italic (added)" & CR2

End If

Else

If sel.Italic = False And styItalic Then

myRes = myRes & "Italic removed" & CR2

End If

End If

If sel.Font.Size = mixed Then

myRes = myRes & "Mixed size " & CR2

Else

If sel.Font.Size <> sel.Style.Font.Size Then \_

myRes = myRes & "Size: " & sel.Font.Size & \_

" (changed from style size)" & CR2

End If

If sel.Font.Superscript = True Then myRes = myRes & \_

"Superscript " & CR2

If sel.Font.Superscript = mixed Then myRes = myRes & \_

"Mixed superscript " & CR2

If sel.Font.Subscript = True Then myRes = myRes & \_

"Subscript " & CR2

If sel.Font.Subscript = mixed Then myRes = myRes & \_

"Mixed subscript " & CR2

If myRes = "" Then myRes = "Pure Normal" & CR2

myResponse = MsgBox(myRes & vbCr & "Continue?", vbQuestion \_

+ vbYesNoCancel, "StyleEffectDetector")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then

Selection.MoveLeft , 2

Else

Selection.MoveRight , 1

End If

Loop Until 0

End Sub

**Sub IZtoIS()**

**'** Paul Beverley - Version 01.10.24

' Corrects text to give -is-, -ys- spellings

promptForConfirmation = True

nonoStyles = "DisplayQuote,ReferenceList"

' textColour = wdColorLightBlue

textColour = 0

highlightColour = wdYellow

' highlightColour = 0

bothTCandHighlight = False

closeExceptionsFile = True

' Address where the IZ exceptions file is held

' On Windows, it will need to be something like:

' myZFile = "C:\Documents and Settings\Paul\My Documents\IZ\_words.docx"

' ...but remove the apostrophe in front of myZFile

' On a Mac, it will need to be something like:

' myZFile = "/Users/Paul/My Documents/Macro stuff/IZ\_words.docx"

' ...but remove the apostrophe in front of myZFile' mySFile = "/Users/Paul/My Documents/Macro stuff/IS\_words.docx"

ExceptionZFile = "IZ\_words"

' Start of main program

nonoStyles = "," & nonoStyles & ","

Set mainDoc = ActiveDocument

myTrack = ActiveDocument.TrackRevisions

gottaDoc = False

For Each thisDoc In Application.Documents

thisName = thisDoc.Name

Debug.Print thisDoc

If InStr(thisName, ExceptionZFile) > 0 Then

gottaDoc = True

closeExceptionsFile = False

thisDoc.Activate

Exit For

End If

Next thisDoc

On Error Resume Next

If gottaDoc = False Then

Documents.Open myZFile

If Err.Number = 5174 Then

MsgBox ("Please open the IZ exceptions file")

Err.Clear

Exit Sub

Else

On Error GoTo 0

End If

End If

allWords = "!"

For Each wd In ActiveDocument.Words

thisWord = Trim(wd)

If Len(thisWord) > 2 Then

If Asc(thisWord) > 32 Then allWords = allWords & thisWord & "!"

End If

Next wd

allWords = LCase(allWords)

If closeExceptionsFile = True And gottaDoc = False Then

ActiveDocument.Close SaveChanges:=False

End If

mainDoc.Activate

Selection.HomeKey Unit:=wdStory

If promptForConfirmation = True Then

myResponse = MsgBox("IZ to IS: Edit the text?", vbQuestion + \_

vbYesNoCancel)

Else

myResponse = vbYes

End If

If myResponse = vbCancel Then Exit Sub

If myTrack = True And myResponse = vbYes Then

If bothTCandHighlight = False Then

textColour = 0

highlightColour = 0

End If

End If

If myResponse = vbNo Then ActiveDocument.TrackRevisions = False

totChanges = 0

For hit = 1 To 4

goes = 0

If hit = 1 Then

thisMany = ActiveDocument.Endnotes.Count

If thisMany > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

End If

End If

If hit = 2 Then

thisMany = ActiveDocument.Footnotes.Count

If thisMany > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

End If

End If

If hit = 3 Then

Set rng = ActiveDocument.Content

Set rng1 = ActiveDocument.Content

thisMany = 1

goes = 1

End If

goes = 1

someText = True

If hit = 4 Then

thisMany = ActiveDocument.Shapes.Count

goes = thisMany

End If

If goes > 0 And thisMany > 0 Then

For myGo = 1 To goes

If hit = 4 Then

Do

someText = False

If ActiveDocument.Shapes(myGo).Type <> 24 And \_

ActiveDocument.Shapes(myGo).Type <> 3 Then

someText = ActiveDocument.Shapes(myGo).TextFrame.HasText

If someText Then

Set rng = ActiveDocument.Shapes(myGo).TextFrame.TextRange

Else

myGo = myGo + 1

End If

End If

DoEvents

Loop Until someText Or myGo > goes

End If

theEnd = rng.End

If someText = True Then

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[iy]z[iea]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If rng.Find.Found = True Then

fnd = rng

opposite = Replace(fnd, "z", "s")

Set rng1 = rng.Duplicate

rng1.Expand wdWord

Do While InStr(ChrW(8217) & "' " & ChrW(160), \_

Right(rng1.Text, 1)) > 0

rng1.MoveEnd , -1

DoEvents

Loop

startWord = rng1.start

endWord = rng1.End

ChangeIt = True

' But don't make the change if...

thisStyle = "," & rng.Style & ","

If InStr(nonoStyles, thisStyle) > 0 Then ChangeIt = False

If rng.Font.StrikeThrough = True Then ChangeIt = False

' if it's not in the list of z's

If InStr(allWords, "!" & LCase(rng1) & "!") = 0 \_

And ChangeIt = True Then

' then definitely change it to an s

If myResponse = vbYes Then

rng.Text = opposite

If ActiveDocument.TrackRevisions = True Then

rng1.End = endWord + 3

Else

rng1.End = endWord

End If

rng1.End = endWord

End If

If highlightColour > 0 Then

rng1.HighlightColorIndex = highlightColour

End If

If textColour > 0 Then

rng1.Font.Color = textColour

End If

totChanges = totChanges + 1

End If

stopNow = False

DoEvents

Else

stopNow = True

End If

rng.start = rng.End + 2

Loop Until stopNow = True

End If

DoEvents

Next myGo

End If

DoEvents

Next hit

ActiveDocument.TrackRevisions = myTrack

If promptForConfirmation = True Then

If myResponse = vbYes Then

MsgBox ("IZ words changed: " & Str(totChanges) & " ")

Else

MsgBox ("IZ words needing to be changed: " & Str(totChanges) & " ")

End If

End If

End Sub

**Sub IStoIZ()**

**'** Paul Beverley - Version 01.10.24

' Corrects text to give -iz-, -yz- spellings

promptForConfirmation = False

doExtraWords = False

szExceptions = "analys,reanalys,overanalys,catalys,dialys,"

szExceptions = szExceptions & "electrolys,paralys,hydrolys"

' textColour = wdColorBlue

textColour = 0

highlightColour = wdYellow

highlightColour = 0

analysesColour = 0

analysesColour = wdYellow

bothTCandHighlight = True

nonoStyles = "DisplayQuote,ReferenceList"

closeExceptionsFile = True

doTextBoxes = True

' Address where the IS exceptions file is held

' On Windows, it will need to be something like:

' mySFile = "C:\Documents and Settings\Paul\My Documents\IS\_words.docx"

' ...but remove the apostrophe in front of mySFile

' On a Mac, it will need to be something like:

' mySFile = "/Users/Paul/My Documents/Macro stuff/IS\_words.docx"

' ...but remove the apostrophe in front of mySFile

ExceptionSFile = "IS\_words"

' Start of main program

If doTextBoxes = True Then maxHits = 4 Else maxHits = 3

nonoStyles = "," & nonoStyles & ","

Set mainDoc = ActiveDocument

myTrack = ActiveDocument.TrackRevisions

gottaDoc = False

For Each thisDoc In Application.Documents

thisName = thisDoc.Name

If InStr(thisName, ExceptionSFile) > 0 Then

gottaDoc = True

closeExceptionsFile = False

thisDoc.Activate

Exit For

End If

Next thisDoc

On Error Resume Next

If gottaDoc = False Then

Documents.Open mySFile

If Err.Number = 5174 Then

MsgBox ("Please open the IS exceptions file")

Err.Clear

Exit Sub

Else

On Error GoTo 0

End If

End If

allWords = "!"

For Each wd In ActiveDocument.Words

thisWord = Trim(wd)

If Asc(thisWord) > 32 Then allWords = allWords & thisWord & "!"

Next wd

If closeExceptionsFile = True And gottaDoc = False Then

ActiveDocument.Close SaveChanges:=False

End If

mainDoc.Activate

If promptForConfirmation = True Then

myResponse = MsgBox("IS to IZ: Edit the text?", \_

vbQuestion + vbYesNoCancel)

Else

myResponse = vbYes

End If

If myResponse = vbCancel Then Exit Sub

If myTrack = True And myResponse = vbYes Then

If bothTCandHighlight = False Then

textColour = 0

highlightColour = 0

End If

End If

If myResponse = vbNo Then ActiveDocument.TrackRevisions = False

totChanges = 0

Selection.HomeKey Unit:=wdStory

For hit = 1 To maxHits

goes = 0

If hit = 1 Then

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

goes = 1

End If

End If

If hit = 2 Then

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

goes = 1

End If

End If

If hit = 3 Then

Set rng = ActiveDocument.Range

goes = 1

End If

If hit = 4 Then

goes = ActiveDocument.Shapes.Count

End If

If goes > 0 Then

For myGo = 1 To goes

someText = True

If hit = 4 Then

Do

someText = False

If ActiveDocument.Shapes(myGo).Type <> 24 And \_

ActiveDocument.Shapes(myGo).Type <> 3 Then

someText = ActiveDocument.Shapes(myGo).TextFrame.HasText

End If

If someText Then

Set rng = ActiveDocument.Shapes(myGo).TextFrame.TextRange

Else

myGo = myGo + 1

End If

DoEvents

Loop Until someText Or myGo > goes

End If

theEnd = rng.End

If someText = True Then

rng.start = 0

rng.End = 0

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[iy]s[iea]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If rng.Find.Found = True Then

fnd = rng

opposite = Replace(fnd, "s", "z")

Set rng1 = rng.Duplicate

rng1.Expand wdWord

Do While InStr(ChrW(8217) & "' " & ChrW(160), \_

Right(rng1.Text, 1)) > 0

rng1.MoveEnd , -1

DoEvents

Loop

startWord = rng1.start

endWord = rng1.End

ChangeIt = True

' But don't make the change if...

thisStyle = "," & rng.Style & ","

If InStr(nonoStyles, thisStyle) > 0 Then ChangeIt = False

If rng.Font.StrikeThrough = True Then ChangeIt = False

' If -is- is near the beginning of the word...

If rng.start - rng1.start < 4 Then

' look for an -is- later in the word

rng.start = rng1.start + 4

rng.End = endWord

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "is[iea]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

opposite = Replace(rng, "s", "z")

If rng.Find.Found = False Or rng.start > endWord \_

Then ChangeIt = False

End If

' Check that it's not in the list of s's

If InStr(allWords, "!" & LCase(rng1) & "!") > 0 \_

Then ChangeIt = False

If InStr(szExceptions, Left(LCase(rng1), 6)) > 0 And \_

rng1.LanguageID = wdEnglishUK Then ChangeIt = False

If LCase(rng1.Text) = "analyses" Then \_

rng1.HighlightColorIndex = analysesColour

If ChangeIt = True Then

' then change it to a z

If myResponse = vbYes Then rng.Text = opposite

If rng1.End < rng.End Then rng1.End = rng.End

If highlightColour > 0 Then

rng1.HighlightColorIndex = highlightColour

End If

If textColour > 0 Then

rng1.Font.Color = textColour

End If

totChanges = totChanges + 1

End If

stopNow = False

Else

stopNow = True

End If

If rng.Text > "" Then

Set rng = rng1.Duplicate

rng.Collapse wdCollapseEnd

End If

i = theEnd - rng.End

If (i Mod 100) = 0 And hit = 3 Then StatusBar = \_

"To go: " & Str(i)

DoEvents

Loop Until stopNow = True

End If

DoEvents

Debug.Print rng.Text

Next myGo

End If

DoEvents

Next hit

ActiveDocument.TrackRevisions = myTrack

If promptForConfirmation = True Then

If myResponse = vbYes Then

MsgBox ("IS words changed: " & Str(totChanges) & " ")

Else

MsgBox ("IS words needing to be changed: " & \_

Str(totChanges) & " ")

End If

End If

End Sub

**Sub IZ\_words()**

' Paul Beverley - Version 02.08.21

' Exceptions for IZtoIS

assize

assizes

baize

baizes

Belize

bizarre

bizarrely

bizarreness

Bizet

Brize

byzant

Byzantine

Byzantines

Byzantinism

Byzantium

Byzants

capsize

capsized

capsizes

capsizing

citizen

citizenries

citizenry

citizens

citizen’s

citizens’

citizenship

citizenships

denizen

denizens

downsize

downsized

downsizes

downsizing

Eliza

Elizabeth

Elizabethan

Giza

Gizeh

liza

lizard

lizards

lizas

maize

midsize

midsized

midsizes

mizen

mizenmast

mizenmasts

mizens

outsize

outsized

outsizes

oversize

oversized

oversizes

pizazz

prize

prized

prizefighter

prizefighters

prizes

prizewinner

prizewinners

prizewinning

prizing

resize

resized

resizes

resizing

seize

seized

seizes

seizing

sizable

sizably

sizar

sizars

size

sizeable

sizeably

sized

sizer

sizers

sizes

sizes

Sizewell

sizing

sizism

sizist

undersize

undersized

unsized

vizier

vizierate

vizierates

vizierial

viziers

viziership

vizierships

whizes

wizard

wizardly

wizardries

wizardry

wizards

wizen

wizened

wizenedness

wizening

wizens

End Sub

Sub IS\_words()

' Paul Beverley - Version 09.06.22

' Exceptions for IStoIZ

acquisition

acquisitions

acquisitive

advertise

advertised

advertisement

advertisements

advertiser

advertisers

advertises

advertising

advise

advised

advisedly

adviser

advises

advising

analysis

anticlockwise

antivivisectionist

antivivisectionists

anywise

aphrodisia

aphrodisiac

aphrodisiacal

aphrodisiacs

appraisal

appraisals

appraise

appraised

appraiser

appraisers

appraises

appraising

appraisingly

acquisitive

acquisitively

acquisition

acquisitions

astrophysical

astrophysically

astrophysicist

astrophysicists

astrophysics

biophysical

biophysically

biophysicist

biophysicists

biophysics

bipartisan

bourgeoisie

catalysis

chastise

chastised

chastiser

chastisers

chastises

chastising

chastisement

chrysalis

chrysalises

circumcise

circumcised

circumcises

circumcising

circumcision

circumcisions

clockwise

collision

collisional

collisions

commiserate

commiserated

commiserates

commiserating

commiseration

commiserations

complaisance

complaisances

complaisant

complaisantly

complaisantness

comprise

comprised

compriser

comprisers

comprises

comprising

compromise

compromised

compromiser

compromisers

compromises

compromising

compromisingly

concise

concisely

conciseness

conciser

concisest

concision

concisions

contraclockwise

contrariwise

counterclockwise

counter-clockwise

counterpoise

counterpoised

counterpoises

counterpoising

countryside

countrysides

crosswise

cruise

cruised

cruises

cruising

cuisine

cuisines

decision

decisions

demise

demises

denoise

denoised

denoises

denoising

derision

despise

despised

despises

despising

devisable

devisably

devise

devised

devises

devising

dialysis

disease

diseases

disenfranchise

disguise

disguised

disguises

disguising

disquisition

electrolysis

electrophysiological

electrophysiology

elision

elisions

emphysema

enfranchise

enfranchised

enfranchisement

enfranchisements

enfranchises

enfranchising

enterprise

enterpriser

enterprises

enterprising

enterprisingly

enterprisingness

envision

envisioned

envisioning

envisions

equipoise

equipoised

equipoises

excision

excisional

excisions

exercise

exercised

exerciser

exercisers

exercises

exercising

expertise

exquisite

exquisitely

exquisiteness

franchise

franchised

franchisee

franchisees

franchiser

franchisers

franchises

franchising

fundraise

fundraised

fundraiser

fundraisers

fundraises

fundraising

geophysical

geophysically

geophysicist

geophysicists

geophysics

glycolysis

heaviside

hemidemisemiquaver

hemidemisemiquavers

hollandaise

hydrolysis

imprecise

imprecisely

impreciseness

imprecision

imprecisions

improvisation

improvisational

improvisatorial

improvisatory

improvise

improvised

improviser

improvises

improvising

inadvisabilities

inadvisability

inadvisable

inadvisableness

indecision

indecisions

indecisive

indecisively

indecisiveness

indivisibilities

indivisibility

indivisible

indivisibleness

indivisibly

inquisition

inquisitional

inquisitions

inquisitive

inquisitively

inquisitiveness

inquisitor

inquisitorial

inquisitorially

inquisitorialness

inquisitors

inquisitory

invisible

invisibly

lackadaisical

lackadaisically

lavoisier

likewise

listwise

louise

lyse

mademoiselle

malaise

malaysia

malaysian

malaysians

marquise

marquises

mayonnaise

mayonnaises

metaphysical

metaphysically

metaphysician

metaphysicians

metaphysics

microdialysis

millisecond

milliseconds

misprision

misprisions

moonrise

moonrises

mortise

mortised

mortises

mortising

multisite

necrolysis

necrolyse

neurophysiology

neurophysiologist

neurophysiologists

nicoise

nonphysical

otherwise

otherwiseness

overadvertise

overadvertised

overappraisal

overexercise

overpraise

overpraised

overpraises

overpraising

overprecise

overprecisely

overwise

overwisely

paradise

paradises

paradisaical

paradisiacal

paradisical

paralysis

partisan

partisans

partisanship

pelvises

perquisite

perquisites

pharisaic

pharisaical

pharisaically

pharisaicalness

pharisee

pharisees

photophysical

photophysics

phthisis

physiology

piecewise

polonaise

polonaises

porpoise

porpoises

portcullises

practise

practised

practises

practising

precise

precised

precisely

preciseness

precises

precising

precision

precisions

premise

premised

premises

prerequisite

prerequisites

prevision

previsions

promise

promised

promisee

promisees

promiser

promisers

promises

promising

promisingly

promisingness

provision

provisional

provisionally

provisionalness

provisionary

provisioned

provisioner

provisioners

provisioning

provisions

psychoanalysis

psychophysics

psychophysical

reacquisition

reacquisitions

reappraisal

reappraisals

reappraise

reappraised

reappraises

reappraising

recision

redivision

reprisal

reprisals

reprise

reprised

reprises

reprising

reprovision

requisite

requisitely

requisiteness

requisites

requisition

requisitionary

requisitioned

requisitioning

requisitions

retrovision

spanwise

stepwise

streamwise

subdivision

subdivisional

subdivisions

sunrise

sunrises

supervise

supervised

supervises

supervising

supervision

supervisionary

supervisions

surmise

surmised

surmises

surmising

surprise

surprised

surprises

surprising

surprisingly

tbilisi

televisable

televise

televised

televises

televising

television

televisionaries

televisionary

televisions

tortoise

tortoises

tortoiseshell

tortoiseshells

treatise

treatises

trellised

trellises

trellising

turquoise

unacquisitive

unacquisitively

unacquisitiveness

uncompromise

uncompromised

uncompromises

uncompromising

uncompromisingly

uncompromisingness

underexercise

undisguised

undisguisedly

undisguisedness

unenterprising

unenterprisingly

unenterprisingness

unphysical

unphysically

unpractised

unpromised

unpromising

unpromisingly

unpromisingness

unraised

unrevised

unsupervised

unsupervisedly

unsurprised

unsurprisedness

unsurprising

unsurprisingly

upraise

upraised

upraises

upraising

vichyssoise

vichyssoises

vivisectionist

vivisectionists

End Sub

**Sub IZIScount()**

' Paul Beverley - Version 06.02.21

' Counts IS/IZ spellings

szExceptions = "analys,reanalys,overanalys,catalys,dialys,"

szExceptions = szExceptions & "electrolys,paralys,hydrolys"

changeZColour = wdTurquoise

changeSColour = wdBrightGreen

nonoStyles = "DisplayQuote,ReferenceList"

ExceptionSFile = "IS\_words"

ExceptionZFile = "IZ\_words"

closeExceptionsFiles = True

' Address where the IS exceptions file is held

' On Windows, it will need to be something like:

mySFile = "C:\Documents and Settings\Paul\My Documents\IS\_words.docx"

' On a Mac, it will need to be something like:

mySFile = "/Users/Paul/My Documents/Macro stuff/IS\_words.docx"

' Address where the IZ exceptions file is held

' On Windows, it will need to be something like:

myZFile = "C:\Documents and Settings\Paul\My Documents\IZ\_words.docx"

' On a Mac, it will need to be something like:

myZFile = "/Users/Paul/My Documents/Macro stuff/IZ\_words.docx"

Set mainDoc = ActiveDocument

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Highlight the IS/IZ words?", \_

vbQuestion + vbYesNoCancel, "ISIZcount")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbYes Then doHighlight = True

gottaSdoc = False

For Each thisDoc In Application.Documents

thisName = thisDoc.Name

If InStr(thisName, ExceptionSFile) > 0 Then

gottaSdoc = True

thisDoc.Activate

Exit For

End If

Next thisDoc

On Error Resume Next

If gottaSdoc = False Then

Documents.Open mySFile

If Err.Number = 5174 Then

MsgBox ("Please open the IS exceptions file")

Err.Clear

Exit Sub

Else

On Error GoTo 0

End If

End If

allSWords = "!"

For Each wd In ActiveDocument.Words

thisWord = Trim(wd)

If Asc(thisWord) > 32 Then allSWords = allSWords & thisWord & "!"

Next wd

allSWords = LCase(allSWords)

If closeExceptionsFiles = True And gottaSdoc = False Then

ActiveDocument.Close SaveChanges:=False

End If

gottaZdoc = False

For Each thisDoc In Application.Documents

thisName = thisDoc.Name

If InStr(thisName, ExceptionZFile) > 0 Then

gottaZdoc = True

thisDoc.Activate

Exit For

End If

Next thisDoc

If gottaZdoc = False Then

Documents.Open myZFile

If Err.Number = 5174 Then

MsgBox ("Please open the IZ exceptions file")

Err.Clear

Exit Sub

Else

On Error GoTo 0

End If

End If

allZWords = "!"

For Each wd In ActiveDocument.Words

thisWord = Trim(wd)

If Asc(thisWord) > 32 Then allZWords = allZWords & thisWord & "!"

Next wd

allZWords = LCase(allZWords)

If closeExceptionsFiles = True And gottaZdoc = False Then

ActiveDocument.Close SaveChanges:=False

End If

mainDoc.Activate

totZwords = 0

For hit = 1 To 3

If hit = 1 Then

thisMany = ActiveDocument.Endnotes.Count

If thisMany > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Set rng1 = ActiveDocument.StoryRanges(wdEndnotesStory)

End If

End If

If hit = 2 Then

thisMany = ActiveDocument.Footnotes.Count

If thisMany > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Set rng1 = ActiveDocument.StoryRanges(wdFootnotesStory)

End If

End If

If hit = 3 Then

Set rng = ActiveDocument.Content

Set rng1 = ActiveDocument.Content

thisMany = 1

End If

If thisMany > 0 Then

theEnd = rng.End

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[iy]z[iea]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If rng.Find.Found = True Then

fnd = rng

opposite = Replace(fnd, "z", "s")

' Find end of word

rng1.Start = rng.End - 1

rng1.End = rng.End

Do

rng1.End = rng1.End + 1

rng1.Start = rng1.Start + 1

Loop Until UCase(rng1) = LCase(rng1)

wdEnd = rng1.Start

' find start of word

rng1.Start = rng.Start

rng1.End = rng.Start + 1

Do

rng1.End = rng1.End - 1

rng1.Start = rng1.Start - 1

Loop Until UCase(rng1) = LCase(rng1)

' set rng 1 to the whole word

rng1.Start = rng1.End

rng1.End = wdEnd

changeIt = True

' But don't make the change if...

thisStyle = rng.Style

If InStr(nonoStyles, thisStyle) > 0 Then changeIt = False

If rng.Font.StrikeThrough = True Then changeIt = False

' if it's not in the list of z's

If InStr(allZWords, "!" & LCase(rng1) & "!") = 0 And \_

changeIt = True Then

If doHighlight = True Then rng1.HighlightColorIndex = \_

changeZColour

totZwords = totZwords + 1

End If

stopNow = False

Else

stopNow = True

End If

rng.Start = rng.End

i = 2 \* theEnd - rng.End

If (i Mod 100) = 0 And hit = 3 Then StatusBar = "To go: " & Str(i)

Loop Until stopNow = True

End If

Next hit

totSwords = 0

For hit = 1 To 3

If hit = 1 Then

thisMany = ActiveDocument.Endnotes.Count

If thisMany > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Set rng1 = ActiveDocument.StoryRanges(wdEndnotesStory)

End If

End If

If hit = 2 Then

thisMany = ActiveDocument.Footnotes.Count

If thisMany > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Set rng1 = ActiveDocument.StoryRanges(wdFootnotesStory)

End If

End If

If hit = 3 Then

Set rng = ActiveDocument.Content

Set rng1 = ActiveDocument.Content

thisMany = 1

End If

If thisMany > 0 Then

theEnd = rng.End

xcvzxcv = rng.Start

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[iy]s[iea]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

If rng.Find.Found = True Then

fnd = rng

opposite = Replace(fnd, "s", "z")

' Find end of word

rng1.Start = rng.End - 1

rng1.End = rng.End

Do

rng1.End = rng1.End + 1

rng1.Start = rng1.Start + 1

Loop Until UCase(rng1) = LCase(rng1)

wdEnd = rng1.Start

' find start of word

rng1.Start = rng.Start

rng1.End = rng.Start + 1

Do

rng1.End = rng1.End - 1

rng1.Start = rng1.Start - 1

Loop Until UCase(rng1) = LCase(rng1)

' set rng 1 to the whole word

rng1.Start = rng1.End

rng1.End = wdEnd

startWord = rng1.Start

changeIt = True

' But don't make the change if...

thisStyle = rng.Style

If InStr(nonoStyles, thisStyle) > 0 Then changeIt = False

If rng.Font.StrikeThrough = True Then changeIt = False

' If -is- is near the beginning of the word...

If rng.Start - rng1.Start < 4 Then

' look for an -is- later in the word

rng.Start = rng1.Start + 4

rng.End = wdEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "is[iea]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

opposite = Replace(rng, "s", "z")

If rng.Find.Found = False Or rng.Start > wdEnd \_

Then changeIt = False

End If

' Check that it's not in the list of s's

If InStr(allSWords, "!" & LCase(rng1) & "!") > 0 \_

Then changeIt = False

If InStr(szExceptions, Left(LCase(rng1), 6)) > 0 \_

And rng1.LanguageID = wdEnglishUK Then changeIt = False

If changeIt = True Then

If doHighlight = True Then rng1.HighlightColorIndex \_

= changeSColour

totSwords = totSwords + 1

End If

stopNow = False

Else

stopNow = True

End If

rng.Start = wdEnd

rng.End = wdEnd

i = theEnd - rng.End

If (i Mod 100) = 0 And hit = 3 Then StatusBar = "To go: " & Str(i)

Loop Until stopNow = True

End If

Next hit

StatusBar = ""

MsgBox ("IZ words: " & Str(totZwords) & vbCr & vbCr & "IS words: " \_

& Str(totSwords))

End Sub

**Sub UKUScount()**

**'** Paul Beverley - Version 29.10.24

' Counts relative spellings between UK and US English

minLengthSpell = 5

doCount = True

timeStart = Timer

UKcount = 0

UScount = 0

CR = vbCr

UKwords = CR

USwords = CR

Selection.HomeKey Unit:=wdStory

i = ActiveDocument.Words.Count

iStart = i

StatusBar = "Spellchecking. To go: 100%"

On Error GoTo ReportIt

Application.ScreenUpdating = False

For Each wd In ActiveDocument.Words

If Len(myWrd) >= minLengthSpell And wd.Font.StrikeThrough = False Then

UKok = Application.CheckSpelling(myWrd, \_

MainDictionary:=Languages(wdEnglishUK).NameLocal)

USok = Application.CheckSpelling(myWrd, \_

MainDictionary:=Languages(wdEnglishUS).NameLocal)

If UKok <> USok Then

If UKok Then

UKcount = UKcount + 1

If InStr(UKwords, "!" & LCase(myWrd) & CR) = 0 Then \_

UKwords = UKwords & "!" & LCase(myWrd) & CR

Else

UScount = UScount + 1

If InStr(USwords, "$" & LCase(myWrd) & CR) = 0 Then \_

USwords = USwords & "$" & LCase(myWrd) & CR

End If

wd.Select

StatusBar = "Spellchecking. To go: " & Trim(Str(Int((i / iStart) \_

\* 100))) & "% UK: " & UKcount & \_

" US: " & UScount

Debug.Print "Spellchecking. To go: " & Trim(Str(Int((i / iStart) \_

\* 100))) & "% UK: " & UKcount & \_

" US: " & UScount

End If

End If

i = i - 1

If i Mod 1000 = 0 Then

StatusBar = "Spellchecking. To go: " & \_

Trim(Str(Int((i / iStart) \* 100))) & \_

"% UK: " & \_

UKcount & " US: " & UScount

DoEvents

End If

Next wd

Application.ScreenUpdating = True

endTime = Timer

MsgBox "Total words" & CR & CR & "UK: " & UKcount \_

& CR & CR & "US: " & UScount

totTime = endTime - timeStart

If doCount = True Then

If totTime > 60 Then MsgBox ((Int(10 \* totTime / 60) / 10) & \_

" minutes")

End If

Selection.HomeKey Unit:=wdStory

Beep

myResponse = MsgBox("List the different words found?", \_

vbQuestion + vbYesNo, "UKUScount")

If myResponse <> vbYes Then Exit Sub

Documents.Add

Selection.TypeText Text:=UKwords & CR & USwords

Set rng = ActiveDocument.Content

rng.Sort

rng.InsertAfter Text:=CR

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{1,}\!"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceOne

DoEvents

.Text = "$"

.Replacement.Text = "^p^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

Set rng2 = ActiveDocument.Content

lenOne = Len(rng2)

.Text = "$"

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

lenTwo = Len(rng2)

.Text = "!"

.Execute Replace:=wdReplaceAll

lenThree = Len(rng2)

End With

Selection.TypeText Text:=CR & "US: " & Trim(Str(lenOne - lenTwo)) & CR

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Distinct words" & CR & CR & "UK: " & \_

Trim(Str(lenTwo - lenThree)) & CR

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub UKUShighlight()**

**'** Paul Beverley - Version 02.10.23

' Marks US spellings within UK text and vice versa

myColour = wdBrightGreen

minLengthSpell = 3

' List here any words that Word erroneously thinks are correct spellings

' Highlight these in a UK text

UKexceptions = "practicing,licencing"

' Highlight these in a US text

USexceptions = "practicing,licencing"

If Selection.LanguageID = wdEnglishUK Then

mainLanguage = wdEnglishUK: altLanguage = wdEnglishUS

myList = UKexceptions

Else

mainLanguage = wdEnglishUS: altLanguage = wdEnglishUK

myList = USexceptions

End If

' Find words from prefix wordlist

myList = myList & ","

myList = Replace(myList, ",,", ",")

numExceptions = Len(myList) - Len(Replace(myList, ",", ""))

ReDim exWord(numExceptions) As String

For i = 1 To numExceptions

nextComma = InStr(myList, ",")

exWord(i) = Left(myList, nextComma - 1)

myList = Mid(myList, nextComma + 1)

DoEvents

Next i

' To measure the time taken

timeStart = Timer

' Check that tracking is off!

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

' Blank off all apostrophe-s

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8217) & "s"

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.MatchCase = True

.Replacement.Text = " zczc"

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Spellcheck the endnotes

myJump = 100

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8217) & "s"

.Replacement.Text = " zczc"

.Execute Replace:=wdReplaceAll

DoEvents

End With

countWds = rng.Words.Count

i = 0

For Each wd In rng.Words

If i Mod myJump = 0 Then StatusBar = "Checking words in endnotes: " \_

& Str(Int(i / myJump) \* myJump)

i = i + 1

If Len(wd) >= minLengthSpell And Application.CheckSpelling(wd, \_

MainDictionary:=Languages(mainLanguage).NameLocal) = False Then

If Application.CheckSpelling(wd, \_

MainDictionary:=Languages(altLanguage).NameLocal) \_

= True Then wd.HighlightColorIndex = myColour

End If

DoEvents

Next wd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " zczc"

.Replacement.Text = ChrW(8217) & "s"

.Execute Replace:=wdReplaceAll

DoEvents

End With

For i = 1 To numExceptions

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = exWord(i)

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

End If

' Spellcheck the footnotes

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8217) & "s"

.Replacement.Text = " zczc"

.Execute Replace:=wdReplaceAll

DoEvents

End With

countWds = rng.Words.Count

i = 0

For Each wd In rng.Words

If i Mod myJump = 0 Then StatusBar = "Checking words in footnotes: " \_

& i

i = i + 1

If Len(wd) >= minLengthSpell And Trim(wd) <> "zczc" And \_

Application.CheckSpelling(wd, \_

MainDictionary:=Languages(mainLanguage).NameLocal) = False Then

If Application.CheckSpelling(wd, \_

MainDictionary:=Languages(altLanguage).NameLocal) \_

= True Then wd.HighlightColorIndex = myColour

End If

DoEvents

Next wd

With rng.Find

.Text = " zczc"

.Replacement.Text = ChrW(8217) & "s"

.Execute Replace:=wdReplaceAll

End With

For i = 1 To numExceptions

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = exWord(i)

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

End If

' Spellcheck the main text

i = ActiveDocument.Words.Count

For Each wd In ActiveDocument.Words

If Len(wd) >= minLengthSpell And Trim(wd) <> "zczc" And \_

Application.CheckSpelling(wd, \_

MainDictionary:=Languages(mainLanguage).NameLocal) = False Then

If Application.CheckSpelling(wd, \_

MainDictionary:=Languages(altLanguage).NameLocal) = True \_

Then wd.HighlightColorIndex = myColour

End If

i = i - 1

If i Mod 100 = 0 Then StatusBar = "Spellchecking. To go: " & Str(i)

DoEvents

Next wd

' restore all apostrophe-s

Set rng = ActiveDocument.Range

With rng.Find

.Text = " zczc"

.Replacement.Text = ChrW(8217) & "s"

.Replacement.Highlight = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

For i = 1 To numExceptions

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = exWord(i)

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Next i

totTime = Timer - timeStart

If totTime > 60 Then MsgBox ((Int(10 \* totTime \_

/ 60) / 10) & " minutes")

ActiveDocument.TrackRevisions = nowTrack

StatusBar = ""

End Sub

**Sub AccentAlyse()**

**'** Paul Beverley - Version 24.10.24

' Analyses all the words that contain an accent

' For including 'Central European' characters

addExtraCharacters = False

' addExtraCharacters = True

fromUnicode = 256

toUnicode = 382

' Minimum word length

minLength = 3

' Create the list of accents to watch out for

allAccents = ""

For i = 192 To 214

allAccents = allAccents & ChrW(i)

Next i

For i = 216 To 246

allAccents = allAccents & ChrW(i)

Next i

For i = 248 To 255

allAccents = allAccents & ChrW(i)

Next i

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox(" AccentAlyse" & vbCr & vbCr & \_

"Analyse this document?", vbQuestion \_

+ vbYesNoCancel, "AccentAlyse")

If myResponse <> vbYes Then Exit Sub

End If

If addExtraCharacters = True Then

For i = fromUnicode To toUnicode

allAccents = allAccents & ChrW(i)

Next i

End If

myLead = " . . . "

Set rng = ActiveDocument.Content

Documents.Add

Set resultDoc = ActiveDocument

Set res = ActiveDocument.Content

Documents.Add

Set testDoc = ActiveDocument

Selection.Text = rng.Text

' Add any foot/endnotes

If FUT.Endnotes.Count > 0 Then

Set nts = FUT.StoryRanges(wdEndnotesStory)

Selection.InsertAfter Text:=nts.Text

End If

If FUT.Footnotes.Count > 0 Then

Set nts = FUT.StoryRanges(wdFootnotesStory)

Selection.InsertAfter Text:=nts.Text

End If

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Finding accented words..." & vbCr \_

& vbCr & vbCr

Selection.start = 0

Selection.Range.Style = ActiveDocument.Styles(wdStyleHeading1)

Selection.Font.Underline = True

Set rng = testDoc.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & allAccents & "]"

.Font.Underline = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

DoEvents

rng.Expand wdWord

myEnd = rng.End

If InStr(" " & ChrW(8217), Right(rng.Text, 1)) \_

> 0 Then rng.End = rng.End - 1

If InStr(" " & ChrW(8217), Right(rng.Text, 1)) \_

> 0 Then rng.End = rng.End - 1

myAccWord = rng.Text

If Len(myAccWord) >= minLength Then

Set rng2 = testDoc.Content

myTot = rng2.End

DoEvents

Debug.Print myAccWord & vbTab & "acc"

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myAccWord

.Wrap = wdFindContinue

.MatchWholeWord = True

.Replacement.Text = "^&!"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

myCount = testDoc.Content.End - myTot

WordBasic.EditUnDo

DoEvents

Selection.End = 0

With rng2.Find

.Replacement.Text = "^&"

.Replacement.Font.Underline = True

.MatchWholeWord = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

' Find an accent-different word

findWord = ""

myAltWord = ""

For i = 1 To Len(myAccWord)

myChar = Mid(myAccWord, i, 1)

If InStr(allAccents, myChar) > 0 Then

findWord = findWord & "^$"

Else

findWord = findWord & myChar

End If

Next i

testDoc.Content.Font.StrikeThrough = False

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = findWord

.Font.Underline = False

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng2.Find.Found = True

DoEvents

myAltWord = rng2.Text

Set rng3 = testDoc.Content

myTot = rng3.End

With rng3.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myAltWord

.Font.Underline = False

.Wrap = wdFindContinue

.MatchWholeWord = True

.Replacement.Text = "^&!"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Debug.Print myAltWord & vbTab & "alt"

DoEvents

myAltCount = testDoc.Content.End - myTot

WordBasic.EditUnDo

DoEvents

Selection.End = 0

DoEvents

With rng3.Find

.ClearFormatting

.Replacement.ClearFormatting

.Replacement.Text = "^&"

.Replacement.Font.StrikeThrough = True

.MatchWholeWord = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

If InStr(res.Text, myAltWord & myLead) = 0 Then

myLine = myAltWord & myLead & Trim(Str(myAltCount)) & vbCr

res.InsertAfter myLine

ActiveDocument.Paragraphs(2).Range.Text = myLine

ActiveDocument.Paragraphs(2).Range.Font.StrikeThrough = True

End If

rng2.start = myEnd

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = findWord

.Font.Underline = False

.Font.StrikeThrough = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Loop

If myAltWord > "" Then

myLine = myAccWord & myLead & Trim(Str(myCount)) & vbCr

res.InsertAfter myLine

ActiveDocument.Paragraphs(1).Range.Text = myLine

End If

End If

rng.start = myEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & allAccents & "]"

.Font.Underline = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Loop

testDoc.Close SaveChanges:=False

resultDoc.Activate

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Accent Use" & vbCr

ActiveDocument.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading1)

If doingSeveralMacros = False Then

Beep

Else

FUT.Activate

End If

End Sub

**Sub AccentedWordCollector()**

' Paul Beverley - Version 03.02.20

' Creates a list of all the accented words in a text

Set rng = ActiveDocument.Content

Documents.Add

Selection.TypeText vbCr

Selection.Text = rng.Text

Set rng = ActiveDocument.Content

' preserve hyphens

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "-"

.Wrap = wdFindContinue

.Replacement.Text = "zczc"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' preserve "single high reverse 9 quotation mark"!

With rng.Find

.Text = ChrW(8219)

.Replacement.Text = "cvcv"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' preserve apostrophes

With rng.Find

.Text = "([a-zA-Z])['" & ChrW(8217) & "]([a-zA-Z])"

.Replacement.Text = "\1qcqc\2"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

'Convert any non-word stuff to spaces

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[\$€£©°+~\#|\_\\*\@\\%^^=0-9\<\>\)\(\[\]\}\{`^0133" \_

& "\/^0160^t.,:;\&\!\?^34^39^=^+^2" \_

& ChrW(8216) & ChrW(8217) & ChrW(8220) & ChrW(8722) \_

& ChrW(8242) & ChrW(8243) & ChrW(8201) & ChrW(8221) & "]{1,}"

.Wrap = wdFindContinue

.Replacement.Text = " "

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

' ensure at least two spaces between words

Set rng = ActiveDocument.Content

With rng.Find

.Text = " "

.Wrap = wdFindContinue

.Replacement.Text = " "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

With rng.Find

.Text = "^p"

.Wrap = wdFindContinue

.Replacement.Text = "^p "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' remove any multiple spaces

Set rng = ActiveDocument.Content

With rng.Find

.Text = " {3,}"

.Wrap = wdFindContinue

.Replacement.Text = " "

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = wdGray25

' unhighlight any accent-less characters

Set rng = ActiveDocument.Content

With rng.Find

.Text = "[ abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ]{1,}"

.Wrap = wdFindContinue

.Replacement.Text = "^&"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set thisDoc = ActiveDocument

Documents.Add

Set newDoc = ActiveDocument

Set rng = ActiveDocument.Content

thisDoc.Activate

For Each myPara In ActiveDocument.Paragraphs

If myPara.range.HighlightColorIndex > 0 Then

For Each wd In myPara.range.Words

If wd.HighlightColorIndex > 0 Then

rng.InsertAfter Text:=wd & vbCr

DoEvents

End If

Next wd

End If

Next myPara

thisDoc.Close SaveChanges:=False

newDoc.Activate

'Restore hyphens

Set rng = ActiveDocument.Content

With rng.Find

.Text = "zczc"

.Wrap = wdFindContinue

.Replacement.Text = "-"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

'Restore apostrophes

Set rng = ActiveDocument.Content

With rng.Find

.Text = "qcqc"

.Wrap = wdFindContinue

.Replacement.Text = ChrW(8217)

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

'Restore funny things

Set rng = ActiveDocument.Content

With rng.Find

.Text = "cvcv"

.Wrap = wdFindContinue

.Replacement.Text = ChrW(8219)

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.Text = " "

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

With rng.Find

.Text = "^13{2,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

rng.Find.Execute Replace:=wdReplaceAll

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

DoEvents

Selection.EndKey Unit:=wdStory

Selection.TypeText vbCr

' Exchange rogue linefeed (ascii 10)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^10"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Selection.MoveEndWhile cset:=Chr(13), Count:=wdForward

Selection.TypeBackspace

Selection.TypeText "List of accented words" & vbCr & vbCr

ActiveDocument.Paragraphs(1).range.Font.Bold = True

ActiveDocument.Paragraphs(1).range.Font.Size = 14

Beep

myResponse = MsgBox("Remove duplicates?", \_

vbQuestion + vbYesNoCancel, "AccentedWordCollector")

If myResponse <> vbYes Then Exit Sub

For j = ActiveDocument.Paragraphs.Count To 2 Step -1

Set rng1 = ActiveDocument.Paragraphs(j).range

Set rng2 = ActiveDocument.Paragraphs(j - 1).range

If anyCase = True Then

r1 = LCase(rng1)

r2 = LCase(rng2)

Else

r1 = rng1

r2 = rng2

End If

If r1 = r2 Then rng1.Delete

DoEvents

Next j

End Sub

**Sub PDFpageNumberer()**

' Paul Beverley - Version 30.09.20

' Adds page numbers to each line of an exported PDF text

numbersInHeader = False

numbersInHeader = True

' allNumbersAtRight = True

allNumbersAtRight = False

' evenNumbersAreOnLeft = True

evenNumbersAreOnLeft = True

' myFontSize = 9

myFontSize = ActiveDocument.Styles(wdStyleNormal).Font.Size - 1

' How many lines to go on checking if the next page number is missing

searchLines = 100

ActiveDocument.TrackRevisions = False

Selection.Expand wdWord

myGuess = Val(Selection)

If myGuess > 0 Then

myDefault = Trim(Selection)

Else

myDefault = "1"

End If

If evenNumbersAreOnLeft = True Then

myOdd = 1

Else

myOdd = 0

End If

If numbersInHeader = True Then

addThis = 0

myPmpt = "HEADERS"

Else

addThis = 1

myPmpt = "FOOTERS"

End If

myNum = InputBox("Start number (in " & \_

myPmpt & ")?", "PDFpageNumberer", myDefault)

pNoNow = Val(myNum)

If pNoNow = 0 Then

Beep

Exit Sub

End If

myPause = MsgBox("Continue without stopping at each page?", \_

vbQuestion + vbYesNoCancel, "PDFpageNumberer")

If myPause = vbCancel Then Exit Sub

paraNum = ActiveDocument.Range(0, \_

Selection.Paragraphs(1).Range.End).Paragraphs.Count

pMax = ActiveDocument.Paragraphs.Count - 3

Set rng = ActiveDocument.Content

If InStr(rng.Text, ChrW(160) & ChrW(160) & vbTab) = 0 Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Wrap = wdFindContinue

.Replacement.Text = "^p^s^s^t"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[ ]{1,}^13"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

pNoNext = pNoNow + 1 - addThis

q = ""

pTextNow = q & Trim(Str(pNoNow))

pTextNext = Trim(Str(pNoNext))

lnNo = 0

lnNoLimit = searchLines

i = paraNum

iWas = i

Do

Do

lnNo = lnNo + 1

' Check paragraph for a new page number

Set pa = ActiveDocument.Paragraphs(i).Range.Duplicate

tabPos = InStr(pa.Text, vbTab)

rng.Start = pa.Start + tabPos

rng.End = pa.End - 1

If Len(rng.Text) > 0 Then

aa = Trim(rng.Words(1))

rWds = rng.Words.Count

zz = rng.Words(rWds)

Else

aa = ""

zz = ""

End If

If allNumbersAtRight = True Then

If zz = pTextNext Then

pNoNow = pNoNext + addThis

pTextNow = q & Trim(Str(pNoNow))

lnNo = 1

End If

Else

If pNoNext Mod 2 = myOdd Then

If zz = pTextNext Then

pNoNow = pNoNext + addThis

pTextNow = q & Trim(Str(pNoNow))

lnNo = 1

End If

Else

If aa = pTextNext Then

pNoNow = pNoNext + addThis

pTextNow = q & Trim(Str(pNoNow))

lnNo = 1

End If

End If

End If

myLine = pTextNow & " " & Trim(Str(lnNo - addThis))

If lnNo > addThis Then

With pa.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\*^t"

.Wrap = wdFindStop

.Replacement.Text = myLine & "^t"

.Replacement.Font.Size = myFontSize

.MatchWildcards = True

.Execute Replace:=wdReplaceOne

DoEvents

End With

End If

i = i + 1

Loop Until lnNo > lnNoLimit Or \_

pNoNow = pNoNext + addThis Or i > pMax

If myPause = vbNo Then

pa.Select

Selection.Collapse wdCollapseEnd

myResponse = MsgBox("Continue?", \_

vbQuestion + vbYesNoCancel, "PDFpageNumberer")

If myResponse <> vbYes Then Exit Sub

End If

If i > pMax Then

Beep

pa.Select

Selection.Collapse wdCollapseEnd

Exit Sub

End If

If lnNo > lnNoLimit Then

i = iWas

q = "??" & q

lnNoLimit = lnNoLimit + searchLines

If Len(q) > 4 Then

Beep

Exit Sub

End If

Else

iWas = i

q = ""

lnNoLimit = searchLines

End If

lnNo = 1

pNoNext = pNoNext + 1

pTextNext = Trim(Str(pNoNext))

pTextNow = q & Trim(Str(pNoNow))

DoEvents

pa.Select

Selection.Collapse wdCollapseStart

Loop Until i > pMax

pa.Select

Selection.Collapse wdCollapseEnd

Beep

End Sub

**Sub PhraseCount()**

' Paul Beverley - Version 22.12.22

' Counts each of a list of words/phrases

phraseLenMax = 4

ignoreDoc = "zzSwitchList"

Set testDoc = ActiveDocument

testName = testDoc.Name

allText = testDoc.Content.Text

allTextLC = LCase(allText)

' Find the word list

For Each myDoc In Documents

If InStr(myDoc.Name, ignoreDoc) = 0 \_

And InStr(myDoc.Name, testName) = 0 Then

numParas = myDoc.Paragraphs.Count + 1

numWords = myDoc.Content.ComputeStatistics(wdStatisticWords)

myProfile = numWords / numParas

If myProfile < phraseLenMax + 1 Then

Set listDoc = myDoc

listDoc.Activate

myResponse = MsgBox("Count items in this list?", vbQuestion \_

+ vbYesNoCancel, "PhraseCount")

If myResponse <> vbYes Then

If myResponse = vbNo Then

myResponse = MsgBox("Sorry! This looked list a phrase list." \_

& vbCr & vbCr & \_

"Please close this file and run the macro again.", \_

vbQuestion + vbOKCancel, "PhraseCount")

End If

Exit Sub

End If

Exit For

End If

End If

Next myDoc

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.Text = rngOld.Text

Set resultsDoc = ActiveDocument

myTot = Len(allText)

testDoc.Activate

For Each myPar In resultsDoc.Paragraphs

testText = Replace(myPar.Range.Text, vbCr, "")

If Len(testText) > 1 Then

Set rngTest = testDoc.Content

myTextTot = rngTest.End

With rngTest.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<" & testText & ">"

.Replacement.Text = "^&!"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

numWholeWds = rngTest.End - myTextTot

If numWholeWds > 0 Then WordBasic.EditUndo

itemCount = Len(Replace(allText, testText, testText & "!")) - myTot

testTextLC = LCase(testText)

itemCountLC = Len(Replace(allTextLC, testTextLC, \_

testTextLC & "!")) - myTot

Set rng = myPar.Range

rng.End = rng.End - 1

rng.InsertAfter Text:=vbTab & Trim(Str(itemCountLC)) \_

& vbTab & Trim(Str(itemCount)) & vbTab & \_

Trim(Str(numWholeWds))

End If

Next myPar

resultsDoc.Activate

Selection.TypeText Text:=vbTab & "Insens" & vbTab & "Sens" & \_

vbTab & "Whole" & vbCr

Selection.WholeStory

Selection.ConvertToTable Separator:=wdSeparateByTabs

Set tb = ActiveDocument.Tables(1)

tb.AutoFitBehavior (wdAutoFitContent)

tb.Borders(wdBorderTop).LineStyle = wdLineStyleNone

tb.Borders(wdBorderLeft).LineStyle = wdLineStyleNone

tb.Borders(wdBorderBottom).LineStyle = wdLineStyleNone

tb.Borders(wdBorderRight).LineStyle = wdLineStyleNone

tb.Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

tb.Borders(wdBorderVertical).LineStyle = wdLineStyleNone

tb.Rows(1).Range.Bold = True

For i = 2 To 4

tb.Columns(i).Select

Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter

Next i

Selection.HomeKey Unit:=wdStory

End Sub

**Sub WordGraph()**

' Paul Beverley - Version 16.01.21

' Gives a visual indication of the occurrences of a word or phrase

myFontSize = 120

myZoomSize = 10

myHighlight = wdBrightGreen

' myHighlight = wdNoHighlight

If Len(Selection) > 1 Then

thisWord = Trim(Selection)

Else

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

thisWord = InputBox("Word to show?", "WordGraph", Selection)

If thisWord = "" Then Exit Sub

End If

Selection.Collapse wdCollapseEnd

If LCase(thisWord) <> thisWord Then

myResponse = MsgBox("Case sensitive?", vbQuestion \_

+ vbYesNoCancel, "WordGraph")

If myResponse = vbCancel Then Exit Sub

caseSense = (myResponse = vbYes)

End If

normalSize = ActiveDocument.Styles(wdStyleNormal).Font.Size

If Left(ActiveDocument.Name, 8) = "Document" Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.MatchCase = False

.Font.Size = myFontSize

.Replacement.Text = ""

.Replacement.Font.Size = normalSize

.Replacement.Highlight = False

.Execute Replace:=wdReplaceAll

End With

Else

Set oldRng = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = oldRng.FormattedText

End If

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myHighlight

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = thisWord

If InStr(thisWord, "<") + InStr(thisWord, ">") + \_

InStr(thisWord, "^") = 0 Then

.MatchWildcards = False

Else

.MatchWildcards = True

End If

If myHighlight > 0 Then

.Replacement.Highlight = True

End If

.Replacement.Text = ""

.Replacement.Font.Size = myFontSize

.MatchCase = caseSense

.Execute Replace:=wdReplaceAll

End With

ActiveWindow.ActivePane.View.Zoom.Percentage = myZoomSize

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub HyphenationToStylesheet()**

' Paul Beverley - Version 27.01.16

' Create a word list from a HyphenAlyse file

CR = vbCr

CR2 = CR & CR

' First check if there's any underline

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

useUline = (rng.Find.Found = True)

' Avoid the title line

Selection.HomeKey Unit:=wdStory

Selection.MoveDown , 1

Set rng = ActiveDocument.Content

rng.Start = Selection.End

' Go and find the highlighted bits

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myText = ","

Do While rng.Find.Found = True

myBit = rng.Text

rng.Expand wdParagraph

myCell = rng

justWord = Trim(Left(myCell, InStr(myCell, ".") - 1))

If Not (Len(myBit) < Len(justWord)) Then

myText = myText & "!"

myBit = justWord

End If

myText = myText & myBit & ","

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

myWords = Split(myText, ",")

' Go and find all underlined (or struckthrough) bits

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

If useUline = True Then

.Font.Underline = True

Else

.Font.StrikeThrough = True

End If

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

allExcepts = ","

Do While rng.Find.Found = True

rng.Expand wdParagraph

myCell = rng.Text

justWord = Trim(Left(myCell, InStr(myCell, ".") - 1))

allExcepts = allExcepts & justWord & ","

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Selection.EndKey Unit:=wdStory

' Look at each of the highlighted items in turn

myExcepts = Split(allExcepts, ",")

For i = 1 To UBound(myWords) - 1

If Left(myWords(i), 1) = "!" Then

myText = Mid(myWords(i), 2) & CR

Else

If Right(myWords(i), 1) = "-" Then

myWord = Replace(myWords(i), "-", "")

myLink = "ALL"

Else

myLink = "NONE"

myWord = myWords(i)

End If

myText = myWord & "<word> " & ChrW(8211) & myLink & " are hyphenated"

addedExcept = False

For j = 1 To UBound(myExcepts) - 1

If Left(myExcepts(j), Len(myWords(i))) = myWords(i) Then

If addedExcept = False Then

addedExcept = True

myText = myText & " except ..." & CR

End If

myText = myText & ChrW(9) & myExcepts(j) & CR

End If

Next j

If addedExcept = False Then myText = myText & CR

End If

Selection.TypeText Text:=myText

Next i

End Sub

**Sub WordPairAlyse()**

' Paul Beverley - Version 20.01.22

' Creates a file of all the adjacent word pairs

' Ignore these words

nonWords = "a,as"

myScreenOff = True

Set FUT = ActiveDocument

At = LCase(FUT.Content.Text)

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox(" WordPairAlyse" & vbCr & vbCr & \_

"Find word pairs?", vbQuestion \_

+ vbYesNoCancel, "WordPairAlyse")

If myResponse <> vbYes Then Exit Sub

End If

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

startTime = Timer

chs = " , . ! : ; [ ] { } ( ) / \ + "

chs = chs & ChrW(8220) & " "

chs = chs & ChrW(8221) & " "

chs = chs & ChrW(8201) & " "

chs = chs & ChrW(8222) & " "

chs = chs & ChrW(8217) & " "

chs = chs & ChrW(8216) & " "

chs = chs & ChrW(8212) & " "

chs = chs & ChrW(8722) & " "

chs = chs & vbCr & " "

chs = chs & vbTab & " "

chs = " " & chs & " "

chs = Replace(chs, " ", " ")

chs = Left(chs, Len(chs) - 1)

chars = Split(chs, " ")

For i = 1 To UBound(chars)

At = Replace(At, chars(i), " " & chars(i) & " ")

Next i

' Remove all non-words

nonWords = "," & nonWords & ","

nonWords = Replace(nonWords, ",,", ",")

nonWords = Left(nonWords, Len(nonWords) - 1)

wd = Split(nonWords, ",")

Set rng = ActiveDocument.Content

For i = 1 To UBound(wd)

At = Replace(At, " " & wd(i) & " ", " ")

DoEvents

Next i

At = Replace(At, " ", " ")

Documents.Add

Selection.Text = " " & At

Set rng = ActiveDocument.Content

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

At = LCase(rng.Text)

myTot = Len(At)

If Asc(At) = 32 Then

ptr = 2

Else

ptr = 1

End If

ptrWas = ptr

Do

ch = Mid(At, ptr, 1)

' Debug.Print ch & "|"

ptr = ptr + 1

Loop Until ch = " "

w2 = Mid(At, ptrWas, ptr - ptrWas - 1)

' Debug.Print w2 & "|"

allChkd = " "

myResult = ""

Do

w1 = w2

ptrWas = ptr

Do

ch = Mid(At, ptr, 1)

ptr = ptr + 1

Loop Until ch = " "

w2 = Mid(At, ptrWas, ptr - ptrWas - 1)

If UCase(w1) <> w1 And UCase(w2) <> w2 Then

oneWd = w1 & w2

chk = " " & oneWd & " "

If InStr(allChkd, chk) = 0 Then

' Check it!

If InStr(At, chk) > 0 Then

numSingle = Len(Replace(At, chk, chk & "!")) - myTot

chk2 = " " & w1 & " " & w2 & " "

numPair = Len(Replace(At, chk2, chk2 & "!")) - myTot

myResult = myResult & w1 & " " & w2 & " . . " & \_

Trim(Str(numPair)) & vbCr

myResult = myResult & oneWd & " . . " & \_

Trim(Str(numSingle)) & vbCr & vbCr

Debug.Print Trim(Str(Int((myTot - ptr) / 6000))) \_

& ",000 to go" & " " & w1 & " " & w2

StatusBar = Trim(Str(Int((myTot - ptr) / 6000))) \_

& ",000 to go" & " " & w1 & " " & w2

End If

allChkd = allChkd & oneWd & " "

End If

End If

DoEvents

Loop Until InStr(Mid(At, ptr), " ") = 0

Selection.WholeStory

Selection.Delete

If Len(myResult) > 0 Then

Selection.Text = myResult

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p^p"

.Replacement.Text = "zczc"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = "^p"

.Replacement.Text = ":"

.Execute Replace:=wdReplaceAll

.Text = "zczc"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

With rng.Find

.Text = "^p"

.Replacement.Text = "^p^p"

.Execute Replace:=wdReplaceAll

.Text = ":"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

If Len(rng.Paragraphs(1)) < 3 Then rng.Paragraphs(1).Range.Delete

Else

Selection.TypeText "No word pairs found" & vbCr

End If

Selection.HomeKey Unit:=wdStory

Selection.TypeText vbCr & "Word pair inconsistencies" & vbCr

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

timNow = Timer

timGone = timNow - startTime

m = Int(timGone / 60)

s = Int(timGone) - m \* 60

Application.ScreenUpdating = True

If doingSeveralMacros = False Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.3

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.3

Beep

MsgBox "Total time:" & Str(m) & " m " & Str(s) & " s"

Else

FUT.Activate

End If

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub SentenceAlyse()**

' Paul Beverley - Version 28.09.19

' Analyse the lengths of sentences

myStep = 4

showReadability = True

Dim tot(5000) As Integer

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

' Remove multispaces

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Text = " {2,}"

.Replacement.Text = " "

.Forward = True

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

DoEvents

' and multi-returns

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "^13{2,}"

.Replacement.Text = "^p"

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

DoEvents

' Abbreviations that Word thinks are sentences!

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = ". ([a-z])"

.Replacement.Text = " \1"

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

DoEvents

' Spaces off paragraph ends

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = " ^p"

.Replacement.Text = "^p"

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

DoEvents

' Spaces off dashes & hyphens

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = " [-^0150] "

.Replacement.Text = " "

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

DoEvents

' Spaces off dashes & hyphens

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "[\?\!]"

.Replacement.Text = "."

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

DoEvents

' Beware multiple dots

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "[.]{2,}"

.Replacement.Text = "."

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

DoEvents

'Find average sentence length

totalWords = 0

maxLen = 0

maxCol = 0

For Each sn In ActiveDocument.Sentences

mySnt = sn

mySnt = Replace(mySnt, Chr(13), "")

If Right(mySnt, 1) = " " Then myCrrn = 0 Else myCrrn = 1

wds = Len(mySnt) - Len(Replace(mySnt, " ", "")) + myCrrn

totalWords = totalWords + wds

If wds > maxLen Then maxLen = wds

col = Int((wds - 1) / myStep)

tot(col) = tot(col) + 1

If col > maxCol Then maxCol = col

' This is for diagnosis

sn.Select

Selection.Start = Selection.End - 2

Selection.End = Selection.End - 1

Selection.range.HighlightColorIndex = wdYellow

DoEvents

Next sn

sentNum = ActiveDocument.Sentences.Count

meanLength = totalWords / sentNum

' Find standard deviation

Dsq = 0

sntCnt = 0

For Each sn In ActiveDocument.Sentences

sntCnt = sntCnt + 1

mySnt = sn

mySnt = Replace(mySnt, Chr(13), "")

If Right(mySnt, 1) = " " Then myCrrn = 0 Else myCrrn = 1

myWds = Len(mySnt) - Len(Replace(mySnt, " ", "")) + myCrrn

Dsq = Dsq + (meanLength - myWds) ^ 2

Next sn

sd = Sqr(Dsq / sentNum)

' Prepare text of results printout

Blank = vbCr & vbCr

myAnswer = Blank & "zczcComplete textczcz" & vbCr

myAnswer = myAnswer & "Words = " & Str(totalWords) & vbCr

myAnswer = myAnswer & "Sentences = " & Str(sntCnt) & vbCr

myAnswer = myAnswer & "Average sentence length = " & Str(Int(meanLength \* 10) / 10) & vbCr

myAnswer = myAnswer & "Standard deviation = " & Str(Int(sd \* 10) / 10) & Blank

' Now create the set of frequencies

myFreq = Blank & "zczcWith headingsczcz" & vbCr

For col = 0 To Int((maxLen - 1) / myStep)

myFreq = myFreq & Str((col \* myStep) + 1) & \_

" to " & Str((col + 1) \* myStep) & " = " & \_

tot(col) & vbCr

Next

' Delete headings = any line with no full stop

' and then do the stats again

For Each pa In ActiveDocument.Paragraphs

myText = pa

If Len(myText) > 3 Then

ch = Mid(myText, Len(myText) - 1, 1)

If InStr("!.?:", ch) = 0 Then pa.range.Font.Underline = True

End If

i = i + 1: If i Mod 100 = 0 Then DoEvents

Next pa

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = ""

.Font.Underline = True

.Replacement.Text = ""

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

DoEvents

'Find average sentence length

totalWords = 0

' Zero the counts ready for next set

For i = 0 To maxCol

tot(i) = 0

Next

maxLen = 0

maxCol = 0

For Each sn In ActiveDocument.Sentences

mySnt = sn

mySnt = Replace(mySnt, Chr(13), "")

If Right(mySnt, 1) = " " Then myCrrn = 0 Else myCrrn = 1

wds = Len(mySnt) - Len(Replace(mySnt, " ", "")) + myCrrn

totalWords = totalWords + wds

If wds > maxLen Then maxLen = wds

col = Int((wds - 1) / myStep)

tot(col) = tot(col) + 1

If col > maxCol Then maxCol = col

Next sn

sentNum = ActiveDocument.Sentences.Count

meanLength = totalWords / sentNum

' Find standard deviation

Dsq = 0

sntCnt = 0

For Each sn In ActiveDocument.Sentences

sntCnt = sntCnt + 1

mySnt = sn

mySnt = Replace(mySnt, Chr(13), "")

If Right(mySnt, 1) = " " Then myCrrn = 0 Else myCrrn = 1

myWds = Len(mySnt) - Len(Replace(mySnt, " ", "")) + myCrrn

Dsq = Dsq + (meanLength - myWds) ^ 2

Next sn

sd = Sqr(Dsq / sentNum)

' Prepare text of results printout

myAnswer = myAnswer & Blank & "zczcWithout headingsczcz" & vbCr

myAnswer = myAnswer & "Words = " & Str(totalWords) & vbCr

myAnswer = myAnswer & "Sentences = " & Str(sntCnt) & vbCr

myAnswer = myAnswer & "Average sentence length = " & Str(Int(meanLength \* 10) / 10) & vbCr

myAnswer = myAnswer & "Standard deviation = " & Str(Int(sd \* 10) / 10) & Blank

' Now create the set of frequencies

myFreq = myFreq & Blank & "zczcWithout headingsczcz" & vbCr

For col = 0 To Int((maxLen - 1) / myStep)

myFreq = myFreq & Str((col \* myStep) + 1) & \_

" to " & Str((col + 1) \* myStep) & " = " & \_

tot(col) & vbCr

Next

ActiveDocument.TrackRevisions = False

' ActiveDocument.Close SaveChanges:=False

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=myAnswer

Selection.TypeText Text:=myFreq

Set rng = ActiveDocument.Content

rng.LanguageID = wdEnglishUK

rng.NoProofing = False

If showReadability = True Then

DocStats = Blank & "zczcWord's Readability Statistics:czcz" & Blank

With ActiveDocument.Content

For i = 1 To 10

If i = 1 Or i = 6 Then DocStats = DocStats & "zczc"

DocStats = DocStats & .ReadabilityStatistics(i) & ": " \_

& .ReadabilityStatistics(i).Value & vbCr

If i = 1 Or i = 6 Then DocStats = DocStats & "czcz"

Next i

End With

Selection.TypeText Text:=DocStats

End If

' Embolden headings

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "zczc(\*)czcz"

.Replacement.Text = "\1"

.Wrap = wdFindStop

.Replacement.Font.Bold = True

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub LongSentenceHighlighter()**

' Paul Beverley - Version 19.05.12

' Highlight all sentences more than a certain length

mediumLength = 80

megaLength = 120

For Each mySent In ActiveDocument.Sentences

If mySent.Words.Count > megaLength Then

mySent.HighlightColorIndex = wdRed

mySent.Select

Else

If mySent.Words.Count > mediumLength Then \_

mySent.HighlightColorIndex = wdYellow

mySent.Select

End If

Next

End Sub

**Sub LongParagraphHighlighter()**

' Paul Beverley - Version 25.03.20

' Highlights all paragraphs more than a certain length

checkWordsCount = True

mediumWordsLength = 100

myWordsColourMed = wdYellow

megaWordsLength = 150

myWordsColourMega = wdBrightGreen

checkSentsCount = True

mediumSentsLength = 6

mySentsColourMed = wdColorBlue

megaSentsLength = 9

mySentsColourMega = wdColorRed

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

For Each myPara In ActiveDocument.Paragraphs

If checkWordsCount = True Then

If myPara.range.Words.Count > megaWordsLength Then

myPara.range.HighlightColorIndex = myWordsColourMega

myPara.range.Select

Else

If myPara.range.Words.Count > mediumWordsLength Then

myPara.range.HighlightColorIndex = myWordsColourMed

myPara.range.Select

End If

End If

End If

DoEvents

If checkSentsCount = True Then

If myPara.range.Sentences.Count > megaSentsLength Then

myPara.range.Font.Color = mySentsColourMega

myPara.range.Select

Selection.Collapse wdCollapseEnd

Else

If myPara.range.Sentences.Count > mediumSentsLength Then

myPara.range.Font.Color = mySentsColourMed

myPara.range.Select

Selection.Collapse wdCollapseEnd

End If

End If

End If

DoEvents

Next

ActiveDocument.TrackRevisions = nowTrack

Beep

End Sub

**Sub ParagraphLineLengthHighlighter()**

' Paul Beverley - Version 11.01.22

' Highlights any paragraphs longer than a number of lines

maxLines = 15

myColour = wdYellow

Set rng = ActiveDocument.Content

rng.Collapse wdCollapseEnd

rng.Start = rng.End - 3

If Left(rng.Text, 1) <> vbCr Then rng.InsertAfter Text:=vbCr

For Each myPar In ActiveDocument.Paragraphs

myPar.Range.Select

Selection.Collapse wdCollapseStart

Selection.MoveDown , maxLines

If myPar.Range.End > Selection.Start Then \_

myPar.Range.HighlightColorIndex = myColour

DoEvents

Next myPar

End Sub

**Sub FrequencySentenceLength()**

' Paul Beverley - Version 15.04.20

' Creates a histogram of sentence length

maxNumColumns = 30

maxBlocks = 60

bigDoc = 250000

myResponse = MsgBox("Allow macro to choose the column width?", vbQuestion \_

+ vbYesNoCancel, "Sentence Length Distribution")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then

myText = InputBox("New column width?", "Frequency Sentence Length")

myWidth = Val(myText)

If myWidth = 0 Then Exit Sub

End If

Set rng = ActiveDocument.Content

n = InStr(ActiveDocument.Content, vbCr & "The end" & vbCr)

If n > 0 Then

rng.End = n + 1

Else

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=vbCr & "The end" & vbCr

DoEvents

Set rng = ActiveDocument.Content

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

' Remove full stops from initials

.Text = "(<[A-Z]>)."

.Replacement.Text = "\1"

.MatchWildcards = True

.Wrap = wdFindStop

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[\)\(]"

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[/^=^+\(\)\<\>\[\]^0133^0160^t%+" & \_

ChrW(8220) & ChrW(8722) & ChrW(8221) & "]"

.Replacement.Text = " "

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "!."

.Replacement.Text = "!"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[^11^12^14]"

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "\?[! ^13]"

.Replacement.Text = " "

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "? "

.Replacement.Text = "?^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[.]{2,}"

.Replacement.Text = " "

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[\!]{2,}"

.Replacement.Text = "!"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[:,;]"

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "'"

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

' Remove multiple spaces

.Text = "[ ]{2,}"

.Replacement.Text = " "

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = " ([.\?\!])"

.Replacement.Text = "\1"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^p "

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "p[0-9\-]{1,}"

.Replacement.Text = "pnnnn"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[0-9][0-9.a-zA-Z]{1,}"

.Replacement.Text = "nnnn"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^pnnnn^p"

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[" & ChrW(912) & "-" & ChrW(8164) & "]{1,}"

.Replacement.Text = "z"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

' All sentences inside paras to separate paras

.Text = "([a-zA-Z0-9])[.\?\!][ ^13]"

.Replacement.Text = "\1^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

' Remove multiple newlines

.Text = "[^13]{2,}"

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Execute Replace:=wdReplaceAll

DoEvents

End With

For Each sn In ActiveDocument.Sentences

wdCount = sn.Words.Count - 1

If wdCount > numMax And wdCount > 10 Then

numMax = wdCount

End If

Next sn

If myWidth = 0 Then myWidth = Int(numMax / maxNumColumns) + 1

heads = Int(numMax / myWidth) + 1

ReDim h(heads) As Integer

For Each sn In ActiveDocument.Sentences

If InStr(sn, "The end") > 0 Then Exit For

wdCount = sn.Words.Count - 1

If wdCount > 1 Then

cat = Int((wdCount + myWidth - 1) / myWidth)

h(cat) = h(cat) + 1

End If

Next sn

maxCount = 0

For i = 1 To heads

If h(i) > maxCount Then maxCount = h(i)

Next i

oneBlock = maxCount / maxBlocks

If oneBlock < 1 Then oneBlock = 1

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr & vbCr

myResult = ""

For i = 1 To heads

fm = myWidth \* (i - 1) + 1

too = fm + myWidth - 1

If fm = 1 Then fm = 2

myRange = Trim(Str(fm)) & "-" & Trim(Str(too))

numLeft = 0

For j = i To heads

numLeft = numLeft + h(j)

Next j

If numLeft = 0 Then Exit For

myResult = myResult & myRange & vbTab

For j = 1 To Int(h(i) / oneBlock)

myResult = myResult & ChrW(9632)

Next j

myResult = myResult & " " & Trim(Str(h(i))) & vbCr

Next i

st = Selection.Start

Selection.TypeText Text:=myResult

Selection.Start = st

Selection.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(1.5)

Selection.Collapse wdCollapseEnd

Beep

myResponse = MsgBox("Create sorted list?", vbQuestion \_

+ vbYesNoCancel, "Frequency Sentence Length")

If myResponse <> vbYes Then Exit Sub

If ActiveDocument.Words.Count > bigDoc Then

Beep

myResponse = MsgBox("This may take some time. OK?", vbQuestion \_

+ vbYesNoCancel, "Frequency Sentence Length")

If myResponse <> vbYes Then Exit Sub

End If

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

Selection.Collapse wdCollapseEnd

Set tempDoc = ActiveDocument

n = InStr(ActiveDocument.Content, "The end" & vbCr)

If n > 0 Then

Set rng = ActiveDocument.Range(n - 1, ActiveDocument.Content.End)

rng.Delete

End If

For Each sn In ActiveDocument.Sentences

wdCount = sn.Words.Count - 1

If wdCount > 1 Then

myData = "[" & Trim(Str(wdCount + 1000)) & "] "

Else

myData = "x000x"

End If

sn.InsertBefore Text:=myData

Next sn

' Sort by straight searching & typing into new doc

If ActiveDocument.Words.Count < bigDoc Then

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Set rng = ActiveDocument.Content

n = InStr(rng.Text, "x000x")

If n > 0 Then

rng.Start = n - 1

rng.Delete

End If

rng.InsertAfter Text:=vbCr

Else

Set tempDoc = ActiveDocument

Documents.Add

Set listDoc = ActiveDocument

For i = 1002 To 1000 + numMax

Set rng = tempDoc.Content

myText = ""

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[" & Trim(Str(i)) & "\]\*^13"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

myText = myText & rng.Text

Selection.Collapse wdCollapseEnd

rng.Find.Execute

Loop

ss = Selection.Start

Selection.TypeText Text:=myText

Selection.Start = ss

Selection.Sort SortOrder:=wdSortOrderAscending

Selection.Collapse wdCollapseEnd

Next i

tempDoc.Close SaveChanges:=False

listDoc.Activate

Selection.EndKey Unit:=wdStory

End If

' Now strip leading zeros

Set rng = ActiveDocument.Content

With rng.Find

' Remove leading 1000's

.Text = "\[1[0]{1,}"

.Replacement.Text = "["

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub FrequencyWordLength()**

' Paul Beverley - Version 15.04.20

' Creates a histogram of word lengths

maxBlocks = 60

longWord = 12

doFrequency = True

CR = vbCr: CR2 = CR & CR

If doFrequency = True Then

myResponse = MsgBox("Include frequency of long words?" \_

& CR2 & "(A slow process on big files)", vbQuestion \_

+ vbYesNoCancel, "Frequency Word Length")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then doFrequency = False

End If

maxLen = 80

ReDim h(maxLen) As Integer

Set rng = ActiveDocument.Content

n = InStr(ActiveDocument.Content, vbCr & "The end" & vbCr)

If n > 0 Then

rng.Start = n + 7

rng.Delete

Else

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8217) & "s"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = "[/\-]"

.Replacement.Text = " "

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Text = "([a-zA-Z0-9]).([a-zA-Z0-9])"

.Replacement.Text = "\1 \2"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Text = "[!a-zA-Z0-9 ^13]"

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Selection.InsertAfter Text:=vbCr & "The end" & vbCr

End If

maxCount = 0

sps = " "

loadsSpaces = sps & sps & sps & sps & sps

allLong = ""

Set rng = ActiveDocument.Content

For Each myPara In ActiveDocument.Paragraphs

If myPara.Range.Text = "The end" & vbCr Then Exit For

For Each wd In myPara.Range.Words

w = Trim(wd)

cat = Len(w)

h(cat) = h(cat) + 1

If cat > longWord - 1 And doFrequency = True Then

wPlus = vbTab & LCase(w) & vbCr

If InStr(allLong, wPlus) = 0 Then

allLong = allLong & Trim(Str(100 + cat)) & wPlus

End If

End If

DoEvents

rng.Start = myPara.Range.End

toGo = rng.Paragraphs.Count

Next wd

StatusBar = loadsSpaces & "Paragraphs to go: " & Str(Int(toGo))

Next myPara

gotLast = False

For i = maxLen To 1 Step -1

If gotLast = False And h(i) > 0 Then

lastItem = i

gotLast = True

End If

If h(i) > maxCount Then maxCount = h(i)

Next i

oneBlock = maxCount / maxBlocks

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr & vbCr

myResult = ""

For i = 1 To lastItem

myResult = myResult & Trim(Str(i)) & vbTab

For j = 1 To Int(h(i) / oneBlock)

myResult = myResult & ChrW(9632)

Next j

myResult = myResult & " " & Trim(Str(h(i))) & vbCr

Next i

Selection.TypeText Text:=myResult & CR2

Beep

If doFrequency = True Then

Set rng = ActiveDocument.Range(Selection.End, ActiveDocument.Content.End)

rng.InsertAfter Text:=allLong

rng.Sort SortOrder:=wdSortOrderAscending

ss = rng.Start

wds = Split(rng, vbCr)

totWds = UBound(wds)

myResult = ""

nowWordLen = 0

For i = 1 To totWds - 1

myText = wds(i)

wd = Mid(wds(i), 5)

ln = Val(Left(wds(i), 3)) - 100

Set rng = ActiveDocument.Content

lenWas = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = wd

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWholeWord = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

lenNow = rng.End

WordBasic.EditUndo

freq = (lenWas - lenNow) / ln - 1

If ln > nowWordLen Then

myResult = myResult & CR & "(" & Trim(Str(ln)) & ")" & CR

nowWordLen = ln

End If

myResult = myResult & wd & vbTab & Trim(Str(freq)) & CR

StatusBar = loadsSpaces & "Words to go: " & \_

Str(Int(totWds - i))

Next i

Selection.Start = ss

Selection.TypeText Text:=myResult

Beep

End If

End Sub

**Sub FrequencyParagraphLength()**

' Paul Beverley - Version 15.04.20

' Creates a histogram of paragraph length

maxNumColumns = 30

maxBlocks = 60

bigDoc = 250000

CR = vbCr: CR2 = CR & CR

myResponse = MsgBox("Allow macro to choose the column width?", vbQuestion \_

+ vbYesNoCancel, "Frequency Paragraph Length")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then

myText = InputBox("New column width?", "Frequency Paragraph Length")

myWidth = Val(myText)

If myWidth = 0 Then Exit Sub

End If

Set rng = ActiveDocument.Content

n = InStr(ActiveDocument.Content, vbCr & "The end" & vbCr)

If n > 0 Then

rng.End = n + 1

Else

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=vbCr & "The end" & vbCr

DoEvents

Set rng = ActiveDocument.Content

End If

For Each pa In ActiveDocument.Paragraphs

wdCount = pa.Range.Words.Count - 1

If wdCount > numMax And wdCount > 10 Then

numMax = wdCount

End If

Next pa

If myWidth = 0 Then

myWidth = Int(numMax / maxNumColumns) + 1

myWidth = Int((myWidth + 3) / 5) \* 5

asdgfsdf = 0

End If

heads = Int(numMax / myWidth) + 1

ReDim h(heads) As Integer

For Each pa In ActiveDocument.Paragraphs

If InStr(pa, "The end") > 0 Then Exit For

wdCount = pa.Range.Words.Count - 1

lastChar = Left(Right(pa.Range.Text, 2), 1)

If wdCount > 1 And LCase(lastChar) = UCase(lastChar) Then

cat = Int((wdCount + myWidth - 1) / myWidth)

h(cat) = h(cat) + 1

End If

Next pa

maxCount = 0

For i = 1 To heads

If h(i) > maxCount Then maxCount = h(i)

Next i

oneBlock = maxCount / maxBlocks

If oneBlock < 1 Then oneBlock = 1

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr & vbCr

myResult = ""

For i = 1 To heads

fm = myWidth \* (i - 1) + 1

too = fm + myWidth - 1

If fm = 1 Then fm = 2

myRange = Trim(Str(fm)) & "-" & Trim(Str(too))

numLeft = 0

For j = i To heads

numLeft = numLeft + h(j)

Next j

If numLeft = 0 Then Exit For

myResult = myResult & myRange & vbTab

For j = 1 To Int(h(i) / oneBlock)

myResult = myResult & ChrW(9632)

Next j

myResult = myResult & " " & Trim(Str(h(i))) & vbCr

Next i

st = Selection.Start

Selection.TypeText Text:=myResult

Selection.Start = st

Selection.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(1.5)

Selection.Collapse wdCollapseEnd

Beep

myResponse = MsgBox("Create sorted list?", vbQuestion \_

+ vbYesNoCancel, "Sentence Length Distribution")

If myResponse <> vbYes Then Exit Sub

If ActiveDocument.Words.Count > bigDoc Then

Beep

myResponse = MsgBox("This may take some time. OK?", vbQuestion \_

+ vbYesNoCancel, "Frequency Paragraph Length")

If myResponse <> vbYes Then Exit Sub

End If

Set rng = ActiveDocument.Content

Documents.Add

Selection.TypeText Text:=CR

Selection.Text = rng.Text

Selection.Collapse wdCollapseEnd

Set tempDoc = ActiveDocument

n = InStr(ActiveDocument.Content, "The end" & vbCr)

If n > 0 Then

Set rng = ActiveDocument.Range(n - 1, ActiveDocument.Content.End)

rng.Delete

End If

For Each pa In ActiveDocument.Paragraphs

wdCount = pa.Range.Words.Count - 1

If wdCount > 1 Then

myData = "[" & Trim(Str(wdCount + 1000)) & "] "

Else

myData = "x000x"

End If

pa.Range.InsertBefore Text:=myData

Next pa

' Sort by straight searching & typing into new doc

If ActiveDocument.Words.Count < bigDoc Then

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Set rng = ActiveDocument.Content

n = InStr(rng.Text, "x000x")

If n > 0 Then

rng.Start = n - 1

rng.Delete

End If

rng.InsertAfter Text:=vbCr

Else

Set tempDoc = ActiveDocument

Documents.Add

Set listDoc = ActiveDocument

For i = 1002 To 1000 + numMax

Set rng = tempDoc.Content

myText = ""

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[" & Trim(Str(i)) & "\]\*^13"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

myText = myText & rng.Text

Selection.Collapse wdCollapseEnd

rng.Find.Execute

Loop

ss = Selection.Start

Selection.TypeText Text:=myText

Selection.Start = ss

Selection.Sort SortOrder:=wdSortOrderAscending

Selection.Collapse wdCollapseEnd

Next i

tempDoc.Close SaveChanges:=False

listDoc.Activate

Selection.EndKey Unit:=wdStory

End If

' Now strip leading zeros

Set rng = ActiveDocument.Content

With rng.Find

' Remove leading 1000's

.Text = "[1"

.Replacement.Text = "^p["

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "\[[0]{1,}"

.Replacement.Text = "["

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub CountCase()**

**'** Paul Beverley - Version 23.11.11

' Count the numbers of upper and lowercase characters

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

myTot = ActiveDocument.Range.End

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-z]"

.Replacement.Text = "^&!"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totLC = ActiveDocument.Range.End - myTot

If totLC > 0 Then WordBasic.editunDo

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z]"

.Replacement.Text = "^&!"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

totUC = ActiveDocument.Range.End - myTot

If totUC > 0 Then WordBasic.editunDo

MsgBox "Uppercase: " & totUC & vbCrLf & "Lowercase: " & \_

totLC & vbCrLf & "Percentage: " & Int(1000 \* \_

totUC / totLC) / 10 & "%"

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub CountWordsInHighlightColour()**

' Paul Beverley - Version 10.02.16

' Count the number of words in a given highlight colour

selColour = Selection.range.HighlightColorIndex

If selColour = 0 Then

myResponse = MsgBox("Count words in ANY colour?", vbQuestion \_

+ vbYesNoCancel, "CountWordsInHighlightColour")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then

MsgBox ("Place cursor in area of colour to be counted")

Exit Sub

End If

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Documents.Add

Do While rng.Find.Found = True

If selColour = 0 Then

Selection.TypeText Text:=rng.Text & vbCr

Selection.EndKey Unit:=wdStory

Else

If rng.HighlightColorIndex = selColour Then

Selection.TypeText Text:=rng.Text & vbCr

Selection.EndKey Unit:=wdStory

End If

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.LanguageID = wdEnglishUK

rng.NoProofing = False

myComment = Str(ActiveDocument.Content.ComputeStatistics(wdStatisticWords))

myComment = "Word count: " & myComment

MsgBox (myComment)

End Sub

**Sub** **CountSectionWords()**

' Paul Beverley - Version 03.04.21

' Counts words in sections of text between headings

myStyle1 = "Heading 1"

myStyle2 = "Heading 2"

doBold = True

Dim myTitle(200) As String

Dim myStart(200) As Long

i = 0

For Each myPar In ActiveDocument.Paragraphs

If (myPar.Style = myStyle1 Or myPar.Style = myStyle2) And \_

Len(myPar.Range.Text) > 2 Then

i = i + 1

myTitle(i) = Replace(myPar.Range.Text, vbCr, "")

If doBold = True And myPar.Style = myStyle1 Then

myTitle(i) = "XXX" & myTitle(i)

End If

myStart(i) = myPar.Range.Start

End If

Next myPar

iMax = i

i = i + 1

myTitle(0) = "Before first heading"

myStart(i) = ActiveDocument.Content.End

myStart(0) = 1

iMax = i

Set rng = ActiveDocument.Content

Documents.Add

For i = 0 To iMax - 1

rng.Start = myStart(i)

rng.End = myStart(i + 1)

myStatWords = rng.ComputeStatistics(wdStatisticWords)

myText = Trim(myTitle(i)) & vbTab & Trim(Str(myStatWords)) & vbCr

Selection.TypeText Text:=myText

Next i

If doBold = True Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "XXX(\*)^13"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "\1^p"

.Replacement.Font.Bold = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

Set rng = ActiveDocument.Content

rng.ConvertToTable Separator:=wdSeparateByTabs

Set tb = ActiveDocument.Tables(1)

tb.AutoFitBehavior (wdAutoFitContent)

tb.Borders(wdBorderTop).LineStyle = wdLineStyleNone

tb.Borders(wdBorderLeft).LineStyle = wdLineStyleNone

tb.Borders(wdBorderBottom).LineStyle = wdLineStyleNone

tb.Borders(wdBorderRight).LineStyle = wdLineStyleNone

tb.Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

tb.Borders(wdBorderVertical).LineStyle = wdLineStyleNone

End Sub

**Sub CountChapterPages()**

' Paul Beverley - Version 03.04.21

' Counts words in sections of text between headings

'Chapter heading style:

myStyle = "Heading 1"

Dim myTitle(100) As String

Dim myPage(100) As Long

i = 0

For Each myPar In ActiveDocument.Paragraphs

If myPar.Style = myStyle And Len(myPar.Range.Text) > 2 Then

i = i + 1

myTitle(i) = Replace(myPar.Range.Text, vbCr, "")

myPar.Range.Collapse wdCollapseStart

myPage(i) = myPar.Range.Information(wdActiveEndPageNumber)

Debug.Print myTitle(i), myPage(i)

End If

Next myPar

iMax = i

Set rng = ActiveDocument.Content

myPage(i + 1) = rng.Information(wdActiveEndPageNumber) + 1

Documents.Add

For i = 1 To iMax

chStart = myPage(i)

chEnd = myPage(i + 1)

myText = Trim(myTitle(i))

numPages = chEnd - chStart

If numPages > 0 Then

myText = myText & vbTab & Trim(numPages) & vbCr

Else

myText = myText & vbTab & "" & vbCr

End If

Selection.TypeText Text:=myText

Next i

Set rng = ActiveDocument.Content

rng.ConvertToTable Separator:=wdSeparateByTabs

Set tb = ActiveDocument.Tables(1)

tb.AutoFitBehavior (wdAutoFitContent)

tb.Borders(wdBorderTop).LineStyle = wdLineStyleNone

tb.Borders(wdBorderLeft).LineStyle = wdLineStyleNone

tb.Borders(wdBorderBottom).LineStyle = wdLineStyleNone

tb.Borders(wdBorderRight).LineStyle = wdLineStyleNone

tb.Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

tb.Borders(wdBorderVertical).LineStyle = wdLineStyleNone

End Sub

**Sub CopyHighlightedTextParagraphs()**

' Paul Beverley - Version 06.01.18

' Copies paragraphs containing highlighted text into a new file

doColouredText = True

doubleSpace = True

Set rng = ActiveDocument.Content

Documents.Add

For Each myPara In rng.Paragraphs

copyIt = (myPara.range.HighlightColorIndex > 0)

If doColouredText = True Then

If myPara.range.Font.Color > 0 Then copyIt = True

End If

If copyIt = True Then

myPara.range.Copy

Selection.Paste

Selection.Collapse wdCollapseEnd

If doubleSpace = True Then Selection.TypeText Text:=vbCr

End If

Next myPara

End Sub

**Sub** **CopyHighlightedTextSentences()**

' Paul Beverley - Version 09.06.20

' Copies sentences containing highlighted text into a new file

doColouredText = True

doubleSpace = True

Set rng = ActiveDocument.Content

Documents.Add

For Each sn In rng.Sentences

copyIt = (sn.HighlightColorIndex > 0)

If doColouredText = True Then

If sn.Font.Color > 0 Then copyIt = True

End If

If copyIt = True Then

sn.Copy

Selection.Paste

Selection.Collapse wdCollapseEnd

If doubleSpace = True Then Selection.TypeText vbCr

End If

Next sn

End Sub

**Sub TooDifficultWordHighlighter()**

**'** Paul Beverley - Version 15.02.12

' Highlights any words not included in word list

ignoreLength = 3

easyWords = "this that these those" & " "

easyWords = easyWords & "your yours their theirs some" & " "

easyWords = easyWords & "will would could were have give take" & " "

' How many files are open. Warn if wrong number.

numDocs = Word.Documents.Count

If numDocs < 2 Then

MsgBox "Please open a text file and a word list file."

Exit Sub

End If

If numDocs > 2 Then

MsgBox "Too many files open."

Exit Sub

End If

' Work out which file is which

For Each myDoc In Documents

myDoc.Activate

Set rng = ActiveDocument.Content

rng.End = rng.Start + 50

myText = LCase(rng.Text)

If InStr(LCase(rng.Text), "word list") > 0 Then

Set TheList = myDoc

Else

Set TheText = myDoc

End If

Next myDoc

' Pick up all the words in the word list, and add the easy words

TheList.Activate

Set rng = ActiveDocument.Content

allWords = Replace(rng.Text & " " & easyWords, Chr(13), " ")

' Highlight any words longer than ignoreLength

' and that are not in the word list or the easy words

TheText.Activate

For Each myWord In ActiveDocument.Words

thisWord = Trim(myWord.Text) & " "

longer = Len(thisWord) > ignoreLength + 1

notInList = InStr(allWords, thisWord) = 0

If longer And notInList Then myWord.HighlightColorIndex = wdYellow

Next myWord

End Sub

**Sub FindRepeatedWords()**

' Paul Beverley - Version 03.11.17

' Finds words that are repeated in a given range

ignoreWords = "their,these,what,that,which"

rangeWords = 20

minLength = 4

If Len(Selection) > 1 Then

Selection.Collapse wdCollapseEnd

Selection.MoveLeft wdWord, 2

End If

Set rng = ActiveDocument.range(Selection.End, \_

ActiveDocument.Content.End)

ignoreWords = "," & ignoreWords & ","

wordsLeft = rng.Words.Count

For i = 1 To wordsLeft

nowWord = LCase(Trim(rng.Words(i)))

If Len(nowWord) >= minLength And InStr(ignoreWords, \_

"," & nowWord & ",") = 0 Then

For j = 1 To rangeWords

If (i + j) < wordsLeft Then

newWord = Trim(LCase(rng.Words(i + j)))

foundOne = False

If nowWord = newWord Then foundOne = True

nowStem = nowWord & "!"

nowStem = Replace(nowStem, "ing!", "")

nowStem = Replace(nowStem, "ed!", "")

nowStem = Replace(nowStem, "es!", "")

nowStem = Replace(nowStem, "s!", "")

nowStem = Replace(nowStem, "!", "")

newStem = newWord & "!"

newStem = Replace(newStem, "ing!", "")

newStem = Replace(newStem, "ed!", "")

newStem = Replace(newStem, "es!", "")

newStem = Replace(newStem, "s!", "")

newStem = Replace(newStem, "!", "")

Debug.Print nowStem, newStem

If nowStem = newStem Then foundOne = True

If nowStem & "e" = newStem Then foundOne = True

If nowStem = newStem & "e" Then foundOne = True

nowEnd = Right(nowStem, 1)

newEnd = Right(newStem, 1)

If nowEnd = newEnd Then

If Len(nowStem) - Len(newStem) = 1 Then \_

checkEnd = Replace(nowStem, newStem, "")

If Len(newStem) - Len(nowStem) = 1 Then \_

checkEnd = Replace(newStem, nowStem, "")

If checkEnd = newEnd Then foundOne = True

End If

If foundOne = True Then

rng.Words(i).Select

startRange = Selection.Start

rng.Words(i + j).Select

Selection.Start = startRange

Exit Sub

End If

End If

Next j

End If

Next i

Selection.EndKey Unit:=wdStory

Beep

End Sub

**Sub RepeatedWordsInSentences()**

' Paul Beverley - Version 12.09.19

' Highlights any words duplicated within a given sentence

wdsIgnore = ",about,been,from,have,here,some,into,"

wdsIgnore = wdsIgnore & ",that,their,them,then,there,these,they,"

wdsIgnore = wdsIgnore & ",this,those,very,were,will,what,with,"

wdsIgnore = wdsIgnore & ",it" & ChrW(8217) & "s," ' This is for "it's"

useHighlight = True

useColour = False

useManyColours = True

Dim myCol(20)

myCol(1) = wdYellow

myCol(2) = wdBrightGreen

myCol(3) = wdTurquoise

myCol(4) = wdPink

myCol(11) = wdColorPink

myCol(12) = wdColorBlue

myCol(13) = wdColorRed

myCol(14) = wdColorBrightGreen

myColTotal = 4

myCount = 0

col = 1

senNum = ActiveDocument.range(0, Selection.Sentences(1).End).Sentences.Count

For sen = senNum To ActiveDocument.Sentences.Count

Set se = ActiveDocument.Sentences(sen)

txt = LCase(se.Text)

lensen = Len(txt)

wds = ","

For i = 1 To se.Words.Count

wd = LCase(Trim(se.Words(i)))

lnw = Len(wd)

If lnw > 3 And InStr(wdsIgnore, "," & wd & ",") = 0 Then

Set rng = se.Duplicate

rs = rng.Start

newtxt = Replace(txt, wd, "")

If (lensen - Len(newtxt)) > lnw And \_

InStr(wds, "," & wd & ",") = 0 Then

wds = wds & wd & ","

For j = 1 To lensen - lnw

If Mid(txt, j, lnw) = wd Then

rng.Start = rs + j - 1

rng.End = rng.Start + lnw

If useHighlight Then

rng.HighlightColorIndex = myCol(col)

End If

If useColour Then

rng.Font.Color = myCol(col + 10)

End If

j = j + lnw

End If

Next j

If useManyColours Then col = col Mod myColTotal + 1

DoEvents

End If

End If

Next i

myCount = myCount + 1

If myCount = 10 Then

se.Words(1).Select

myCount = 0

End If

Next sen

End Sub

**Sub RepeatedWordsInParagraphs()**

' Paul Beverley - Version 10.09.23

' Highlights any words duplicated within given paragraphs

wdsIgnore = ",about,been,from,have,here,some,into,"

wdsIgnore = wdsIgnore & ",that,their,them,then,there,these,they,"

wdsIgnore = wdsIgnore & ",this,those,very,were,will,what,with,"

wdsIgnore = wdsIgnore & ",it" & ChrW(8217) & "s," ' This is for "it's"

useHighlight = True

useColour = False

useManyColours = True

Dim myCol(20)

myCol(1) = wdYellow

myCol(2) = wdBrightGreen

myCol(3) = wdTurquoise

myCol(4) = wdPink

myCol(5) = wdRed

myCol(6) = wdGray25

myCol(7) = wdGray50

myCol(8) = wdDarkYellow

myCol(11) = wdColorPink

myCol(12) = wdColorBlue

myCol(13) = wdColorRed

myCol(14) = wdColorBrightGreen

myCol(15) = wdColorPink

myCol(16) = wdColorBlue

myCol(17) = wdColorRed

myCol(18) = wdColorBrightGreen

myColTotal = 8

myCount = 0

col = 1

parNum = ActiveDocument.Range(0, Selection.Paragraphs(1).Range.End).Paragraphs.Count

For par = parNum To ActiveDocument.Paragraphs.Count

Set pa = ActiveDocument.Paragraphs(par)

txt = LCase(pa.Range.Text)

lenpar = Len(txt)

wds = ","

For i = 1 To pa.Range.Words.Count

wd = LCase(Trim(pa.Range.Words(i)))

lnw = Len(wd)

If lnw > 3 And InStr(wdsIgnore, "," & wd & ",") = 0 Then

Set rng = pa.Range.Duplicate

rs = rng.Start

newtxt = Replace(txt, wd, "")

If (lenpar - Len(newtxt)) > lnw And \_

InStr(wds, "," & wd & ",") = 0 Then

wds = wds & wd & ","

For j = 1 To lenpar - lnw

If Mid(txt, j, lnw) = wd Then

rng.Start = rs + j - 1

rng.End = rng.Start + lnw

If useHighlight Then

rng.HighlightColorIndex = myCol(col)

End If

If useColour Then

rng.Font.Color = myCol(col + 10)

End If

j = j + lnw

End If

Next j

If useManyColours Then col = col Mod myColTotal + 1

DoEvents

End If

End If

DoEvents

Next i

myCount = myCount + 1

If myCount = 20 Then

pa.Range.Words(1).Select

myCount = 0

End If

Next par

End Sub

**Sub MacroRunNoTC()**

' Paul Beverley - Version 01.01.22

' Runs an analysis on track-accepted text

myMacro = "RepeatedWordsInParagraphs"

myScreenOff = True

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

Selection.HomeKey Unit:=wdStory

Set myDoc = ActiveDocument

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.Text = rngOld.Text

Application.Run MacroName:=myMacro

Set rng = ActiveDocument.Content

If rng.HighlightColorIndex > 0 Then

For ip = 1 To rng.Paragraphs.Count

Set myPar = rng.Paragraphs(ip)

If myPar.Range.HighlightColorIndex > 0 Then

For iw = 1 To myPar.Range.Words.Count

Set wd = myPar.Range.Words(iw)

If wd.HighlightColorIndex > 9999 Then

For ic = 1 To wd.Characters.Count

Set ch = wd.Characters(ic)

If ch.HighlightColorIndex > 0 Then

rngOld.Paragraphs(ip).Range.Words(iw).Characters(ic).HighlightColorIndex \_

= ch.HighlightColorIndex

End If

DoEvents

Next ic

Else

If wd.HighlightColorIndex > 0 Then

rngOld.Paragraphs(ip).Range.Words(iw).HighlightColorIndex \_

= wd.HighlightColorIndex

End If

End If

DoEvents

Next iw

End If

DoEvents

Next ip

End If

If rng.Font.ColorIndex > 0 Then

For ip = 1 To rng.Paragraphs.Count

Set myPar = rng.Paragraphs(ip)

If myPar.Range.Font.ColorIndex > 0 Then

For iw = 1 To myPar.Range.Words.Count

Set wd = myPar.Range.Words(iw)

If wd.Font.ColorIndex > 9999 Then

For ic = 1 To wd.Characters.Count

Set ch = wd.Characters(ic)

If ch.Font.ColorIndex > 0 Then

rngOld.Paragraphs(ip).Range.Words(iw).Characters(ic).Font.ColorIndex \_

= ch.Font.ColorIndex

End If

DoEvents

Next ic

Else

If wd.Font.ColorIndex > 0 Then

rngOld.Paragraphs(ip).Range.Words(iw).Font.ColorIndex \_

= wd.Font.ColorIndex

End If

End If

DoEvents

Next iw

End If

DoEvents

Next ip

End If

ActiveDocument.Close SaveChanges:=False

myDoc.Activate

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub SpecialSortsLister()**

' Paul Beverley - Version 18.01.23

' Collect all special sorts in a file

' useWithinFRedit = True

useWithinFRedit = False

listAccentedChars = True

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0) Or \_

useWithinFRedit = True

Set rng = ActiveDocument.Range

rng.Copy

Documents.Add

Selection.Paste

Selection.WholeStory

Selection.Range.Style = ActiveDocument.Styles(wdStyleNormal)

Selection.Font.Reset

numberCmnts = ActiveDocument.Comments.Count

If numberCmnts > 0 Then ActiveDocument.DeleteAllComments

Set rng = ActiveDocument.Content

CR = vbCr

If listAccentedChars = True Then

mainChars = "[abcdefghijklmnopqrstuvwxyz" & \_

"ABCDEFGHIJKLMNOPQRSTUVWXYZ^+ ]{1,}"

Else

mainChars = "[a-zA-Z^+ ]{1,}"

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mainChars

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Beep

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^m0-9^13^t,.:;\!\?^=^+\-\(\)£]{1,}"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Beep

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & ChrW(8216) & ChrW(8217) & ChrW(8221) & ChrW(8220) \_

& ChrW(8230) & ChrW(174) & ChrW(176) & "]{1,}"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Beep

allSorts = ""

For Each myChar In ActiveDocument.Characters

uCode = 0

If myChar > "" Then uCode = AscW(myChar)

If InStr(allSorts, myChar) = 0 And uCode > 128 Then

allSorts = allSorts & myChar & vbTab

Select Case uCode

Case 160: ExtraBit = "non-breaking space"

Case 176: ExtraBit = "degree symbol"

Case 178: ExtraBit = "dodgy squared symbol"

Case 179: ExtraBit = "dodgy cubed symbol"

Case 184: ExtraBit = "cedilla"

Case 186: ExtraBit = "masculine ordinal"

Case 215: ExtraBit = "proper multiply symbol"

Case 8194: ExtraBit = "en space"

Case 8195: ExtraBit = "em space"

Case 8201: ExtraBit = "thin space"

Case 8222: ExtraBit = "German open curly quote"

Case 8226: ExtraBit = "ordinary bullet"

Case 8242: ExtraBit = "unicode: single prime"

Case 8243: ExtraBit = "unicode: double prime"

Case 8249: ExtraBit = "French open quote"

Case 8250: ExtraBit = "French close quote"

Case 8722: ExtraBit = "minus sign"

Case Else: ExtraBit = "zczc"

End Select

allSorts = allSorts & ExtraBit & CR

DoEvents

End If

Next myChar

Selection.WholeStory

If allSorts = "" Then allSorts = CR & "No special sorts used"

Selection.TypeText allSorts

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^tzczc"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Selection.EndKey Unit:=wdStory

Selection.TypeText CR

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Special sorts used" & CR

ActiveDocument.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading1)

If doingSeveralMacros = False Then

Beep

Else

FUT.Activate

End If

End Sub

**Sub ListAllColouredWords()**

' Paul Beverley - Version 02.12.16

' Creates an alphabetic list of all words in the selected font colour.

selColour = Selection.range.Font.Color

Set rng = ActiveDocument.Content

rng.Copy

Documents.Add

Selection.Paste

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = wdNoHighlight

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = selColour

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = False

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchCase = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

rng.Copy

Selection.WholeStory

Selection.PasteSpecial DataType:=wdPasteText

Set rng = ActiveDocument.Content

Selection.Sort SortOrder:=wdSortOrderAscending

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^$"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.Execute

End With

Selection.Collapse wdCollapseStart

Selection.Start = 0

Selection.Delete

End Sub

**Sub ListHighlightedText()**

' Paul Beverley - Version 08.09.17

' Lists alphabetically any text that is highlighted

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Highlight = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr

Selection.HomeKey Unit:=wdStory

Selection.Expand wdParagraph

Do While Len(Selection) = 1

Selection.Delete

Selection.Expand wdParagraph

Loop

Selection.HomeKey Unit:=wdStory

End Sub

**Sub ListAllLinks()**

' Paul Beverley - Version 01.09.16

' Creates a table of all URLs in the file

Set thisDoc = ActiveDocument

Documents.Add

Set listDoc = ActiveDocument

Set rng = ActiveDocument.Content

thisDoc.Activate

For Each lnk In ActiveDocument.Fields

includeLink = True

If lnk.Kind <> 2 Then

lnk.Select

includeLink = False

' MsgBox "Different link kind"

End If

If lnk.Type <> 88 Then

lnk.Select

includeLink = False

' MsgBox "Different link type"

End If

If includeLink = True Then

linkCode = lnk.Code

myURL = Replace(linkCode, "HYPERLINK", "")

myURL = Trim(Replace(myURL, """", ""))

myVisibleText = lnk.Result

' lnk.ShowCodes = True

rng.InsertAfter Text:=myVisibleText & vbTab

startLink = rng.End

rng.InsertAfter Text:=myURL & vbCr

rng.End = rng.End - 1

rng.Start = startLink - 1

rng.Font.Color = wdColorBlue

rng.Font.Italic = True

rng.Start = rng.End + 1

End If

Next lnk

listDoc.Activate

Selection.WholeStory

Selection.ConvertToTable Separator:=wdSeparateByTabs

Selection.Tables(1).AutoFitBehavior (wdAutoFitContent)

If deleteTableBorders = True Then

Selection.Tables(1).Borders(wdBorderTop).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderLeft).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderBottom).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderRight).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderVertical).LineStyle = wdLineStyleNone

End If

Selection.HomeKey Unit:=wdStory

End Sub

**Sub ListOfParas()**

' Paul Beverley - Version 16.01.21

' Lists all paragraphs (formatted text) starting with certain text

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

' In the main text, find the search word(s) then create the list

findString = InputBox("Search for?", "ListOfParas", Trim(Selection))

If findString = "" Then Exit Sub

If startOfLine Then findString = "^13" & findString

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13" & findString

.Wrap = wdFindStop

.MatchCase = False

.Forward = True

.MatchWildcards = False

.Execute

End With

Documents.Add

Set rng2 = ActiveDocument.Content

Do While rng.Find.Found = True

rng.MoveStart , 1

rng.Expand wdParagraph

rng.Copy

rng.Collapse wdCollapseEnd

rng2.Paste

rng2.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Beep

Selection.EndKey Unit:=wdStory

End Sub

**Sub ListOfTextParas()**

' Paul Beverley - Version 16.01.21

' Lists all paragraphs (pure text) starting with certain text

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

' In the main text, find the search word(s) then create the list

findString = InputBox("Search for?", "ListOfTextParas", Trim(Selection))

If findString = "" Then Exit Sub

If startOfLine Then findString = "^13" & findString

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13" & findString

.Wrap = wdFindStop

.MatchCase = False

.Forward = True

.MatchWildcards = False

.Execute

End With

Documents.Add

Set resultsDoc = ActiveDocument

Set rng2 = resultsDoc.Content

Do While rng.Find.Found = True

rng.MoveStart , 1

rng.Expand wdParagraph

myText = rng.Text

rng.Collapse wdCollapseEnd

rng2.InsertAfter Text:=myText

rng.Find.Execute

Loop

Beep

Selection.EndKey Unit:=wdStory

resultsDoc.Activate

End Sub

**Sub** **FontColourDocumentSplit()**

' Paul Beverley - Version 25.06.20

' Splits a document into coloured and not coloured

removeColour = True

' Create a first copy of the document

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = wdYellow

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = 0

rng.Shading.BackgroundPatternColor = wdColorAutomatic

' Highlight any text in black

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.ColorIndex = 0

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

' Including the other 'black'

.Font.ColorIndex = 1

.Replacement.Text = "^&"

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Make a second copy

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

' In the first copy, remove unhighlighted = coloured text

' i.e. leave English behind

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = False

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Clear the highlighting

rng.HighlightColorIndex = 0

' In the second copy, unhighlight the CRs

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Replacement.Text = ""

.Replacement.Highlight = False

.Execute Replace:=wdReplaceAll

DoEvents

' then remove the highlighted text = black

Set rng = ActiveDocument.Content

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

End With

If removeColour = True Then rng.Font.ColorIndex = 0

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub ListHighlightedOrColoured()**

' Paul Beverley - Version 09.09.17

' Lists alphabetically any text that is highlighted

removeColouration = True

myColour = Selection.range.Font.ColorIndex

myHighlight = Selection.range.HighlightColorIndex

styleColour = Selection.range.Style.Font.ColorIndex

If myColour + myHighlight = 0 Then

MsgBox "Please place the cursor in the colour to be listed."

Exit Sub

End If

If myColour > 0 And myColour = styleColour Then

MsgBox "Sorry, this macro doesn't work with colours in styles."

Exit Sub

End If

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

rng.Font.StrikeThrough = True

' Wait for repagination to complete

Selection.EndKey Unit:=wdStory

Selection.HomeKey Unit:=wdStory

mixedColour = 9999999

If myHighlight > 0 Then

For Each myPara In rng.Paragraphs

col = myPara.range.HighlightColorIndex

If col <> mixedColour Then

If col = myHighlight Then myPara.range.Font.StrikeThrough = False

Else

For Each wd In myPara.range.Words

col = wd.HighlightColorIndex

If col <> mixedColour Then

If col = myHighlight Then wd.Font.StrikeThrough = False

Else

For Each ch In wd.Characters

col = ch.HighlightColorIndex

If col <> mixedColour Then

If col = myHighlight Then ch.Font.StrikeThrough = False

End If

DoEvents

Next ch

End If

DoEvents

Next wd

End If

DoEvents

j = j + 1

If j Mod 25 = 0 Then myPara.range.Select

Next myPara

Else

For Each myPara In rng.Paragraphs

col = myPara.range.Font.ColorIndex

If col <> mixedColour Then

If col = myColour Then myPara.range.Font.StrikeThrough = False

Else

For Each wd In myPara.range.Words

col = wd.Font.ColorIndex

If col <> mixedColour Then

If col = myColour Then wd.Font.StrikeThrough = False

Else

For Each ch In wd.Characters

col = ch.Font.ColorIndex

If col <> mixedColour Then

If col = myColour Then ch.Font.StrikeThrough = False

End If

DoEvents

Next ch

End If

DoEvents

Next wd

End If

DoEvents

Next myPara

End If

Set rng = ActiveDocument.Content

If removeColouration = True Then

rng.Font.ColorIndex = 0

rng.HighlightColorIndex = 0

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Font.StrikeThrough = True

.Execute Replace:=wdReplaceAll

End With

myResponse = MsgBox("Sort?", vbQuestion \_

+ vbYesNoCancel, "ListHighlightedOrColoured")

Set rng = ActiveDocument.Content

If myResponse = vbYes Then

rng.Sort SortOrder:=wdSortOrderAscending

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr

Selection.HomeKey Unit:=wdStory

Selection.Expand wdParagraph

Do While Len(Selection) = 1

Selection.Delete

Selection.Expand wdParagraph

Loop

Selection.HomeKey Unit:=wdStory

Beep

Else

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13{1,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Paragraphs(1).range

If Asc(rng) = 13 Then rng.Delete

End If

End Sub

**Sub ListOfList()**

' Paul Beverley - Version 16.01.21

' Lists all items in a list that contain a particular text

CaseSensitive = True

addBlankLine = False

Set doc = ActiveDocument

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(" ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

myWord = Selection

Selection.Collapse wdCollapseStart

myText = InputBox("Find?", "ListOfList", myWord)

If myText = "" Then Exit Sub

Documents.Add

For Each myPara In doc.Paragraphs

DoEvents

myParaText = myPara.Range.Text

If CaseSensitive = False Then

myParaText = LCase(myParaText)

myText = LCase(myText)

End If

If InStr(myParaText, myText) > 0 Then

Selection.Range.FormattedText = myPara.Range.FormattedText

Selection.EndKey Unit:=wdStory

If addBlankLine = True Then Selection.TypeText vbCr

End If

Next myPara

Beep

End Sub

**Sub HideShowText()**

' Paul Beverley - Version 07.09.17

' Makes body text invisible

If Selection.Start = Selection.End Then

doAll = True

Set rng = ActiveDocument.Content

Else

doAll = False

Set rng = Selection.range.Duplicate

End If

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Application.ScreenUpdating = False

For Each ch In rng.Characters

myCol = ch.Font.Color

If myCol = wdColorWhite Then

myColour = wdColorAutomatic

Exit For

End If

If myCol = 0 Or myCol = wdColorAutomatic \_

Or myCol = wdBlack Then

myColour = wdColorWhite

Exit For

End If

Next ch

mixedColour = 9999999

If doAll = True Then

For Each myPara In rng.Paragraphs

If myPara.range.Font.Color <> mixedColour Then

myPara.range.Font.Color = myColour

Else

For Each wd In myPara.range.Words

myCol = wd.Font.Color

If myCol <> mixedColour Then

If myCol = 0 Or myCol = wdColorAutomatic Or \_

myCol = wdColorWhite Or myCol = wdBlack \_

Then wd.Font.Color = myColour

Else

For Each ch In wd.Characters

myCol = ch.Font.Color

If myCol = 0 Or myCol = wdColorAutomatic Or \_

myCol = wdColorWhite Or myCol = wdBlack \_

Then ch.Font.Color = myColour

Next ch

End If

Next wd

End If

Next myPara

Else

For Each ch In rng.Characters

myCol = ch.Font.Color

If myCol = 0 Or myCol = wdColorAutomatic Or \_

myCol = wdColorWhite Or myCol = wdBlack \_

Then ch.Font.Color = myColour

Next ch

End If

ActiveDocument.TrackRevisions = myTrack

Application.ScreenUpdating = True

Beep

End Sub

**Sub ChronologyChecker()**

' Paul Beverley - Version 15.04.22

' Copies paragraphs containing date references into a new file

' Case sensitive

myColour\_1 = wdYellow

myWords\_1 = "Monday, Tuesday, Wednesday, Thursday, Friday,"

myWords\_1 = myWords\_1 & "Saturday, Sunday,"

myColour\_2 = wdBrightGreen

myWords\_2 = "January, February, April, June, July, August,"

myWords\_2 = myWords\_2 & "September, October, November, December"

' Case insensitive

myColour\_3 = wdYellow

myWords\_3 = "years old, tomorrow, next day, morning, evening, week, month"

' Case insensitive + whole word

myColour\_4 = wdYellow

myWords\_4 = "age, aged"

' Case sensitive AND whole word

myColour\_5 = wdBrightGreen

myWords\_5 = "May, March, Mon, Tue, Tues, Wed, Weds, Thu, Thurs, Fri, Sat, Sun"

' For years

myColour\_6 = wdTurquoise

multiSpace = 4

myWords\_1 = Replace(myWords\_1, " ", "")

myWords\_1 = Replace("," & myWords\_1 & ",", ",,", ",")

myWords\_2 = Replace(myWords\_2, " ", "")

myWords\_2 = Replace("," & myWords\_2 & ",", ",,", ",")

myWords\_3 = Replace(myWords\_3, " ", "")

myWords\_3 = Replace("," & myWords\_3 & ",", ",,", ",")

myWords\_4 = Replace(myWords\_4, " ", "")

myWords\_4 = Replace("," & myWords\_4 & ",", ",,", ",")

myWords\_5 = Replace(myWords\_5, " ", "")

myWords\_5 = Replace("," & myWords\_5 & ",", ",,", ",")

allWords = Replace(myWords\_1 & myWords\_2 & myWords\_3 & myWords\_4 \_

& myWords\_5, ",,", ",")

For i = 1 To multiSpace

SP = SP & vbCr

Next i

Set rng = ActiveDocument.Content

Documents.Add

For Each myPar In rng.Paragraphs

copyIt = False

For Each wd In myPar.Range.Words

DoEvents

mywd = Trim(wd.Text)

myTest = "," & LCase(mywd) & ","

If InStr(LCase(allWords), myTest) > 0 Then

copyIt = True

Exit For

End If

If Len(mywd) = 4 And LCase(mywd) = UCase(mywd) Then

' Is the first character 1 or 2?

isYear = (InStr("12", Left(mywd, 1)) > 0)

' Are the other three characters digits 0-9?

For i = 2 To 4

j = Asc(Mid(mywd, i)) - 48

If j < 0 Or j > 9 Then isYear = False

Next i

If isYear = True Then

copyIt = True

Exit For

End If

End If

DoEvents

Next wd

If copyIt Then

myPar.Range.Copy

Selection.Paste

Selection.Collapse wdCollapseEnd

Selection.TypeText SP

DoEvents

End If

Next myPar

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Dates context" & vbCr & vbCr

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

Selection.MoveLeft , 2

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour\_1

ActiveDocument.Content.HighlightColorIndex = wdNoHighlight

mywd = Split(myWords\_1, ",")

For i = 1 To UBound(mywd) - 1

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mywd(i)

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

mywd = Split(myWords\_2, ",")

Options.DefaultHighlightColorIndex = myColour\_2

For i = 1 To UBound(mywd) - 1

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mywd(i)

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

mywd = Split(myWords\_3, ",")

Options.DefaultHighlightColorIndex = myColour\_3

For i = 1 To UBound(mywd) - 1

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mywd(i)

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

mywd = Split(myWords\_4, ",")

Options.DefaultHighlightColorIndex = myColour\_4

For i = 1 To UBound(mywd) - 1

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mywd(i)

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = False

.MatchWholeWord = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

mywd = Split(myWords\_5, ",")

Options.DefaultHighlightColorIndex = myColour\_5

For i = 1 To UBound(mywd) - 1

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mywd(i)

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = True

.MatchWholeWord = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

Options.DefaultHighlightColorIndex = myColour\_6

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[12][0-9]{3}>"

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = True

.MatchWholeWord = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

For i = ActiveDocument.Paragraphs.Count To 2 Step -1

Set myPar = ActiveDocument.Paragraphs(i).Range

If Len(myPar.Text) > 1 And myPar.HighlightColorIndex = wdNoHighlight Then

myPar.Select

Selection.MoveEnd , multiSpace

Selection.Delete

End If

DoEvents

Next i

Beep

End Sub

**Sub WordsPhrasesInContext()**

' Paul Beverley - Version 19.02.18

' Copies paragraphs containing specific names into a new file

myListName = "zzSwitchList"

' myListName = "zzFReditList"

' findWords = "Brown | Jones | Green"

findWords = ""

myBasicColour = wdBrightGreen

returnToText = False

maxWds = 10

CaseSensitive = True

multiSpace = 4

CR = vbCr

For i = 1 To multiSpace

sp = sp & vbCr

Next i

myWords = ""

Set rng = ActiveDocument.Range(Selection.Start, ActiveDocument.Content.End)

parasToEnd = rng.Paragraphs.Count

totParas = ActiveDocument.Paragraphs.Count

If (totParas / parasToEnd) > 10 Then

Selection.Expand wdParagraph

myResponse = MsgBox("Start with this line?", vbQuestion \_

+ vbYesNoCancel, "WordsPhrasesInContext")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbYes Then

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.End = ActiveDocument.Content.End

myWords = rng.Text

myWords = Replace(CR & myWords & CR, CR & CR, CR)

myWords = Replace(myWords, CR & CR, CR)

myWords = Replace(myWords, CR & CR, CR)

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

myWd = Split(myWords, CR)

totWords = UBound(myWd) - 1

ReDim myCol(totWords) As Integer

If rng.HighlightColorIndex = wdNoHighlight Then

For i = 1 To totWords

myCol(i) = myBasicColour

Next i

Else

For i = 1 To totWords

myCol(i) = rng.HighlightColorIndex

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

Next i

End If

Else

Selection.HomeKey Unit:=wdStory

End If

End If

Set mainDoc = ActiveDocument

If myWords = "" Then

gotExternalList = False

' Does a FRedit/Switch list have a "Context words:" line?

For Each myWnd In Application.Windows

thisName = myWnd.Document.Name

If InStr(thisName, myListName) > 0 Then

myWnd.Document.Activate

gotExternalList = True

Exit For

End If

Next myWnd

If gotExternalList = False Then mainDoc.Activate

' If so, load the word and colour arrays from it

DoEvents

Set rng = ActiveDocument.Content

DoEvents

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Context words:"

.Replacement.Text = ""

.MatchCase = True

.MatchWildcards = False

.Execute

DoEvents

End With

If rng.Find.Found = False Then

mainDoc.Activate

DoEvents

Set rng = ActiveDocument.Content

DoEvents

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Context words:"

.Replacement.Text = ""

.MatchCase = True

.MatchWildcards = False

.Execute

End With

DoEvents

End If

If rng.Find.Found Then

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

rng.End = ActiveDocument.Content.End

myWords = rng.Text

myWords = Replace(CR & myWords & CR, CR & CR, CR)

myWords = Replace(myWords, CR & CR, CR)

myWords = Replace(myWords, CR & CR, CR)

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

myWd = Split(myWords, CR)

totWords = UBound(myWd) - 1

ReDim myCol(totWords) As Integer

For i = 1 To totWords

myCol(i) = rng.HighlightColorIndex

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

Next i

End If

End If

If myWords = "" Then

If Selection.Start = Selection.End Then Selection.Expand wdWord

If findWords = "" Then

myWords = InputBox("Names to find?", "WordsInContext", \_

Trim(Selection))

If myWords = "" Then Exit Sub

Else

myWords = findWords

End If

myWords = Replace(myWords, "| ", "|")

myWords = Replace(myWords, " |", "|")

myWords = Replace(myWords, "|", CR)

myWords = Replace(CR & myWords & CR, CR & CR, CR)

myWords = Replace(myWords, CR & CR, CR)

myWords = Replace(myWords, CR & CR, CR)

myWd = Split(myWords, CR)

totWords = UBound(myWd) - 1

ReDim myCol(totWords) As Integer

For i = 1 To totWords

myCol(i) = myBasicColour

Next i

End If

' In case we're in the external list...

mainDoc.Activate

Set rng = ActiveDocument.Content

Documents.Add

myTestWords = Replace(myWords, ChrW(172), "")

For Each myPara In rng.Paragraphs

parText = myPara.Range.Text

StatusBar = parText

copyIt = False

If Left(parText, 13) = "Context words" Then Exit For

For Each wd In myPara.Range.Words

DoEvents

Set myrange = wd.Duplicate

For i = 1 To maxWds

theseWds = Trim(myrange.Text)

myTest = CR & theseWds & CR

If InStr(LCase(myTestWords), LCase(myTest)) > 0 Then

copyIt = True

Exit For: Exit For

End If

myrange.MoveEnd wdWord, 1

Next i

Next wd

If copyIt Then

myPara.Range.Copy

Selection.Paste

Selection.Collapse wdCollapseEnd

Selection.TypeText sp

DoEvents

End If

Next myPara

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Words/phrases in context" & vbCr & vbCr

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

Selection.MoveLeft , 2

ActiveDocument.Content.HighlightColorIndex = wdNoHighlight

oldColour = Options.DefaultHighlightColorIndex

myWd = Split(myWords, CR)

For i = 1 To totWords

If Asc(myWd(i)) = 172 Then

myWd(i) = Mid(myWd(i), 2)

CaseSensitive = False

Else

CaseSensitive = True

End If

Options.DefaultHighlightColorIndex = myCol(i)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myWd(i)

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = CaseSensitive

.MatchWildcards = False

.MatchWholeWord = False

.Execute Replace:=wdReplaceAll

End With

Next i

Options.DefaultHighlightColorIndex = oldColour

For i = ActiveDocument.Paragraphs.Count To 2 Step -1

Set myPara = ActiveDocument.Paragraphs(i).Range

If Len(myPara.Text) > 1 And myPara.HighlightColorIndex = wdNoHighlight Then

myPara.Select

Selection.MoveEnd , multiSpace

Selection.Delete

End If

Next i

If returnToText = True Then mainDoc.Activate

Beep

End Sub

**Sub FieldAlyse()**

' Paul Beverley - Version 06.09.19

' Counts all fields of different types

Dim n(100) As Integer

knownTypes = " 13 33 58 72 81 88 "

stopAtUnknown = True

Set myFile = ActiveDocument

Set rng = ActiveDocument.Content

For i = 1 To myFile.Fields.Count

Set fld = myFile.Fields(i)

myType = fld.Type

n(myType) = n(myType) + 1

If stopAtUnknown = True And InStr(knownTypes, Str(myType)) = 0 Then

fld.Select

myResponse = MsgBox("Type: " & Str(myType) & " Continue?", \_

vbQuestion + vbYesNoCancel)

If myResponse <> vbYes Then Exit Sub

knownTypes = knownTypes & " " & Str(myType)

End If

Next i

myResult = ""

For i = 1 To 100

If n(i) > 0 Then

myResult = myResult & Str(n(i)) & " " & \_

"field type" & Str(i) & " ("

Select Case i

Case 13: myType = "table of contents"

Case 33: myType = "page"

Case 58: myType = "equation"

Case 72: myType = "note reference"

Case 81: myType = "citation"

Case 88: myType = "hyperlink"

Case Else: myType = "type unknown by this macro)" & vbCr & \_

"[https://docs.microsoft.com/en-us/dotnet/api/microsoft." & \_](https://docs.microsoft.com/en-us/dotnet/api/microsoft.%22%20&%20_)

"sharepoint.spfieldtype?view=sharepoint-server"

End Select

myResult = myResult & myType & ")" & vbCr & vbCr

End If

Next i

Selection.EndKey Unit:=wdStory

If myResult = "" Then

Beep

Else

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.TypeText Text:=vbCr & vbCr & myResult

ActiveDocument.TrackRevisions = myTrack

End If

End Sub

**Sub FRedit()**

**'** Paul Beverley - Version 07.12.23

' Scripted find and replace

' ©2009-2023 Paul Beverley

promptForSelectedText = True

doFinalBeep = True

showTime = True

useReverseStrikeFeature = False

debugging = False

' debugging = True

funnyCode = "Blank"

' used as the code to mean "Leave the Find/Replace box blank"

isMacro = "DoMacro"

' Used as the code to mean "Do the following macro"

caseCode = ChrW(172)

' The 'bent pipe' character (horizontal line with a bent end)

maxLines = 2000

' the maximum number of F&R lines in your list

myScreenOff = True

oldColour = Options.DefaultHighlightColorIndex

CR = vbCr: CR2 = CR & CR

Set rng = ActiveDocument.Content

justEditStrikeText = False

editAll = vbNo

If useReverseStrikeFeature = True And \_

rng.Font.StrikeThrough = 9999999 Then

editAll = MsgBox("Edit all text?", \_

vbQuestion + vbYesNoCancel, "FRedit")

If editAll = vbCancel Then Exit Sub

If editAll = vbNo Then

myPrompt = "Edit ONLY strikethrough text (= Yes)" & CR2

myPrompt = myPrompt & "DON'T edit strikethrough text (= No)" & CR2

myPrompt = myPrompt & "Panic! (= Cancel)"

myResponse = MsgBox(myPrompt, vbQuestion + vbYesNoCancel, "FRedit")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbYes Then justEditStrikeText = True

End If

End If

ReDim findText(maxLines) As String, ReplaceText(maxLines) As String

ReDim fHlight(maxLines) As Integer, rHlight(maxLines) As Integer

ReDim fTxtCol(maxLines) As Long, rTxtCol(maxLines) As Long

ReDim fFontSize(maxLines) As Integer, rFontSize(maxLines) As Integer

ReDim styleArray(maxLines, 4) As String, funct(maxLines, 18) As Boolean

timeStart = Timer

For i = 1 To 30

myspaces = myspaces & "> "

Next i

myBaseStyle = ActiveDocument.Styles(wdStyleNormal)

' In case there's an error

Set thisRng = Selection.Range.Duplicate

' FRedit the selected text only?

Set rng = ActiveDocument.Content

If Len(Selection) = Len(rng) Then Selection.HomeKey Unit:=wdStory

Set rngSel = Selection.Range.Duplicate

If Selection.End <> Selection.Start Then

If promptForSelectedText = True Then

myResponse = MsgBox("Work on selected text only?", \_

vbQuestion + vbYesNoCancel, "FRedit")

If myResponse = vbCancel Then Exit Sub

Else

myResponse = vbYes

End If

wasSelectedText = (myResponse = vbYes)

Else

wasSelectedText = False

End If

If debugging = False Then On Error GoTo ReportIt

If myScreenOff = True Then Application.ScreenUpdating = False

' Assume cursor is in the file to be edited

Set workFile = ActiveDocument

myTrack = workFile.TrackRevisions

' Find the FRedit list file

gottaList = 0

For Each myDoc In Application.Documents

DoEvents

pNum = myDoc.Paragraphs.Count

myNum = 5

If pNum < 5 Then myNum = pNum

Set rng = myDoc.Paragraphs(myNum).Range

rng.Start = 0

myTest = Replace(LCase(rng), " ", "")

If InStr(myTest, ChrW(124) & "fredit") Then

gottaList = gottaList + 1

If gottaList = 1 Then

myDoc.Activate

Set theList = myDoc

End If

End If

Next myDoc

myWarning = "Please ensure that your FRedit list starts with: | FRedit" \_

& CR & CR2 & "Place the cursor in the text to be edited, and rerun FRedit."

If gottaList = 0 Then

Beep

myResponse = MsgBox("Can't find a FRedit LIST." & CR2 & \_

myWarning, vbExclamation + vbOKOnly, "FRedit")

Exit Sub

End If

If gottaList > 1 Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

myResponse = MsgBox("There are TWO or more FRedit LISTs open!" & CR2 & \_

"Is this the list you want to use?", vbQuestion + vbYesNo, "FRedit")

If myResponse <> vbYes Then

myResponse = MsgBox("Please close the list(s) you don't want to use," & \_

CR & "and then rerun FRedit", vbExclamation + vbOKOnly, "FRedit")

Exit Sub

End If

End If

If workFile = theList Then

Beep

myResponse = MsgBox("The CURSOR seems to be in the FRedit LIST." & CR2 & \_

"Place the cursor in the TEXT TO BE EDITED, and rerun FRedit.", \_

vbExclamation + vbOKOnly, "FRedit")

Exit Sub

End If

' Check for rogue manual line breaks in the list

Set rng = ActiveDocument.Content

mlb = InStr(rng.Text, Chr(11))

If mlb > 0 Then

Selection.Start = mlb - 1

Selection.MoveLeft , 1

Selection.End = mlb + 1

Beep

MsgBox "Beware: FRedit list contains manual line breaks!"

Application.Run macroName:="ShowFormatting"

MsgBox "I've turned Show Formatting on, so you can see."

Application.ScreenUpdating = True

Exit Sub

End If

' Create the list of F&Rs

Selection.HomeKey Unit:=wdStory

FRitem = 0

fNotes = False: eNotes = False: Etext = True: BoxText = False

' FRitem is used to count the actual lines that are F&R commands

lastLine = ActiveDocument.Paragraphs.Count

For i = 1 To lastLine

' Look through the list for things that aren't actual F&Rs

Do

Do

Set rng = ActiveDocument.Paragraphs(i).Range

StatusBar = myspaces & "Reading line: " & Str(FRitem)

DoEvents

rng.End = rng.End - 1

allLine = rng

i = i + 1

' Keep going until you find a non-blank line

Loop Until Len(allLine) > 0 Or i > lastLine

' Check if it's a comment line, i.e. starting with a pad

firstChar = Left(allLine, 1)

If firstChar = "|" Then

If InStr(allLine, "rack =") > 0 Then

' Check for | Track command

Select Case LCase(Right(allLine, 3))

Case " no"

If myTrack Then

MsgBox "Please switch track changes off!", \_

vbOKOnly + vbExclamation, "FRedit"

workFile.Activate

Application.ScreenUpdating = True

Exit Sub

End If

Case "yes"

If myTrack = False Then

MsgBox "Please switch track changes on!", \_

vbOKOnly + vbExclamation, "FRedit"

workFile.Activate

Application.ScreenUpdating = True

Exit Sub

End If

Case Else

myError = 4: GoTo myErrorReport

End Select

End If

If InStr(allLine, "ootnotes =") > 0 Then

' Check for | Footnote command

If InStr(LCase(allLine), "yes") > 0 Then

fNotes = True

Else

fNotes = False

End If

End If

If InStr(allLine, "ndnotes =") > 0 Then

' Check for | Endnote command

If InStr(LCase(allLine), "yes") > 0 Then

eNotes = True

Else

eNotes = False

End If

End If

If InStr(allLine, "ext =") > 0 Then

' Check for | Text command

If InStr(LCase(allLine), "yes") > 0 Then

Etext = True

Else

Etext = False

End If

End If

If InStr(allLine, "box =") > 0 Or InStr(allLine, \_

"boxes =") > 0 Then

' Check for | Textboxes command

If InStr(LCase(allLine), "yes") > 0 Then

BoxText = True

Else

BoxText = False

End If

End If

End If

Loop Until firstChar <> "|" Or i > lastLine

i = i - 1

' If you find hashes, stop looking for F&R lines

If Left(allLine, 1) = "#" Then Exit For

' It's an F&R line, so check for highlighting

If Len(allLine) > 1 Then

lineStart = rng.Start

lineEnd = rng.End

' Has it got a vertical bar in it?

padPosition = InStr(allLine, "|")

' If not, it's a two-line F&R

If padPosition = 0 Then

' We've got two lines

fText = rng

' Clip off MatchCase and Wildcard indicators

mchWild = False

mchCase = True

If Left(fText, 1) = caseCode Then

fText = Right(fText, Len(fText) - 1)

mchCase = False

rng.Start = rng.Start + 1

End If

If Left(fText, 1) = "~" Then

fText = Right(fText, Len(fText) - 1)

mchWild = True

rng.Start = rng.Start + 1

End If

' What style is the Find in?

fStyle = rng.Style

If fStyle = myBaseStyle Then fStyle = ""

rng.End = lineStart + 1

' Check format & type colour of first char of Find

fItalic = rng.Italic

fBold = rng.Bold

fSuper = rng.Font.Superscript

fSub = rng.Font.Subscript

fUline = rng.Underline

fSmall = rng.Font.SmallCaps

fAllcaps = rng.Font.AllCaps

fDstrike = rng.Font.DoubleStrikeThrough

fFont = rng.Font.Name

fSize = rng.Font.Size

fHiColour = rng.HighlightColorIndex

fTxtColour = rng.Font.Color

i = i + 1

Set rng = ActiveDocument.Paragraphs(i).Range

rng.End = rng.End - 1

lineEnd = rng.End

rText = rng

padPosition = InStr(rText, "|")

If padPosition > 0 Or Len(rng) = 0 Then myError = 2: GoTo myErrorReport

' What style is the Replace in?

rstyle = rng.Style

If rstyle = myBaseStyle Then rstyle = ""

rng.End = rng.Start + 1

' Check format & type colour of first char of Replace

rItalic = rng.Italic

rBold = rng.Bold

rSuper = rng.Font.Superscript

rSub = rng.Font.Subscript

rUline = rng.Underline

rSmall = rng.Font.SmallCaps

rAllcaps = rng.Font.AllCaps

rDstrike = rng.Font.DoubleStrikeThrough

rFont = rng.Font.Name

rSize = rng.Font.Size

rHiColour = rng.HighlightColorIndex

rTxtColour = rng.Font.Color

Else

' It's all on one line, so no style change

' unless the style is different from Normal

' What style is the Find in?

fStyle = ""

rstyle = rng.Style

If rstyle = myBaseStyle Then rstyle = ""

' Chop up the line into F and R

fText = Left(allLine, padPosition - 1)

rText = Right(allLine, Len(allLine) - padPosition)

' Clip off MatchCase and Wildcard indicators

mchWild = False

mchCase = True

If Left(fText, 1) = caseCode Then

fText = Right(fText, Len(fText) - 1)

mchCase = False

rng.Start = rng.Start + 1

End If

If Left(fText, 1) = "~" Then

fText = Right(fText, Len(fText) - 1)

mchWild = True

rng.Start = rng.Start + 1

End If

rng.End = lineStart + 1

' Check format & type colour of first char of Find

If rng.Text = " " Then

rng.Start = rng.End

rng.End = rng.End + 1

End If

fItalic = rng.Italic

fBold = rng.Bold

fSuper = rng.Font.Superscript

fSub = rng.Font.Subscript

fUline = rng.Underline

fSmall = rng.Font.SmallCaps

fAllcaps = rng.Font.AllCaps

fDstrike = rng.Font.DoubleStrikeThrough

fFont = rng.Font.Name

fSize = rng.Font.Size

fHiColour = rng.HighlightColorIndex

fTxtColour = rng.Font.Color

rng.End = lineStart + padPosition + 1

rng.Start = lineStart + padPosition

' Check format & type colour of first char of Replace

rItalic = rng.Italic

rBold = rng.Bold

rSuper = rng.Font.Superscript

rSub = rng.Font.Subscript

rUline = rng.Underline

rSmall = rng.Font.SmallCaps

rAllcaps = rng.Font.AllCaps

rDstrike = rng.Font.DoubleStrikeThrough

rFont = rng.Font.Name

rSize = rng.Font.Size

rHiColour = rng.HighlightColorIndex

rTxtColour = rng.Font.Color

trackIt = Not (rng.Font.StrikeThrough)

End If

FRitem = FRitem + 1

' Save all the F&R info in arrays

findText(FRitem) = fText

ReplaceText(FRitem) = rText

styleArray(FRitem, 1) = fStyle

styleArray(FRitem, 2) = rstyle

styleArray(FRitem, 3) = fFont

styleArray(FRitem, 4) = rFont

fHlight(FRitem) = fHiColour

rHlight(FRitem) = rHiColour

fTxtCol(FRitem) = fTxtColour

rTxtCol(FRitem) = rTxtColour

fFontSize(FRitem) = fSize

rFontSize(FRitem) = rSize

funct(FRitem, 1) = mchWild

funct(FRitem, 2) = mchCase

funct(FRitem, 3) = fBold

funct(FRitem, 4) = rBold

funct(FRitem, 5) = fItalic

funct(FRitem, 6) = rItalic

funct(FRitem, 7) = fSuper

funct(FRitem, 8) = rSuper

funct(FRitem, 9) = fSub

funct(FRitem, 10) = rSub

funct(FRitem, 11) = fUline

funct(FRitem, 12) = rUline

funct(FRitem, 13) = fSmall

funct(FRitem, 14) = rSmall

funct(FRitem, 15) = fAllcaps

funct(FRitem, 16) = rAllcaps

funct(FRitem, 17) = trackIt

funct(FRitem, 18) = fDstrike

' ^p is not allowed in wildcard searches!

If mchWild And InStr(fText, "^" & "p") > 0 Then

myError = 5: GoTo myErrorReport

End If

If InStr(fText, "^" & "{") > 0 Then

myError = 95: GoTo myErrorReport

End If

' You can't do case insensitive AND wildcard

If mchWild And mchCase = False Then

myError = 6: GoTo myErrorReport

End If

End If

Next i

lastItem = FRitem

' Check the Normal font

normalSize = ActiveDocument.Styles(myBaseStyle).Font.Size

normalFont = ActiveDocument.Styles(myBaseStyle).Font.Name

workFile.Activate

' Remember if TC is on or off

trackNow = ActiveDocument.TrackRevisions

' Remember current cursor position and move cursor to top, for speed

Set thisRng = Selection.Range.Duplicate

Selection.HomeKey Unit:=wdStory

' Define the ranges

' Get the data out of the arrays

For FRitem = 1 To lastItem

fText = findText(FRitem)

rText = ReplaceText(FRitem)

fStyle = styleArray(FRitem, 1)

rstyle = styleArray(FRitem, 2)

fFont = styleArray(FRitem, 3)

rFont = styleArray(FRitem, 4)

fHiColour = fHlight(FRitem)

rHiColour = rHlight(FRitem)

fSize = fFontSize(FRitem)

rSize = rFontSize(FRitem)

fTxtColour = fTxtCol(FRitem)

rTxtColour = rTxtCol(FRitem)

mchWild = funct(FRitem, 1)

mchCase = funct(FRitem, 2)

fBold = funct(FRitem, 3)

rBold = funct(FRitem, 4)

fItalic = funct(FRitem, 5)

rItalic = funct(FRitem, 6)

fSuper = funct(FRitem, 7)

rSuper = funct(FRitem, 8)

fSub = funct(FRitem, 9)

rSub = funct(FRitem, 10)

fUline = funct(FRitem, 11)

rUline = funct(FRitem, 12)

fSmall = funct(FRitem, 13)

rSmall = funct(FRitem, 14)

fAllcaps = funct(FRitem, 15)

rAllcaps = funct(FRitem, 16)

trackIt = funct(FRitem, 17)

fDstrike = funct(FRitem, 18)

If trackNow = True Then ActiveDocument.TrackRevisions = trackIt

If fText = isMacro Then

Application.Run macroName:=rText

Else

' funnyCode means fText should be blank

If InStr(fText, funnyCode) > 0 Then fText = ""

BlankIt = False

If InStr(rText, funnyCode) > 0 Then rText = "": BlankIt = True

If fText = "<Symbol>" Then

fText = "": BlankIt = True

fFont = "Symbol"

rFont = normalFont

End If

' Replace hex code strings with codes

codePos = InStr(fText, "<&H")

Do While codePos > 0

codeLen = InStr(fText, ">") - codePos

uText = Mid(fText, codePos, codeLen)

uCode = Val(Right(uText, codeLen - 1))

uChar = ChrW(uCode)

fText = Replace(fText, uText & ">", uChar)

codePos = InStr(fText, "<&H")

Loop

codePos = InStr(rText, "<&H")

Do While codePos > 0

codeLen = InStr(rText, ">") - codePos

uText = Mid(rText, codePos, codeLen)

uCode = Val(Right(uText, codeLen - 1))

uChar = ChrW(uCode)

rText = Replace(rText, uText & ">", uChar)

codePos = InStr(rText, "<&H")

Loop

For hit = 1 To 4

If hit = 1 Then

If Not (fNotes = True And ActiveDocument.Footnotes.Count \_

> 0) Then hit = 2

End If

If hit = 2 Then

If Not (eNotes = True And ActiveDocument.Endnotes.Count \_

> 0) Then hit = 3

End If

If hit = 3 Then

If Etext = False Then hit = 4

End If

goes = 1

If hit = 4 Then

If BoxText = True Then

goes = ActiveDocument.Shapes.Count

Else

hit = 5

End If

End If

If hit < 5 Then

For myGo = 1 To goes

If hit = 1 Then Set rng = \_

ActiveDocument.StoryRanges(wdFootnotesStory)

If hit = 2 Then Set rng = \_

ActiveDocument.StoryRanges(wdEndnotesStory)

If hit = 3 Then

If wasSelectedText Then

Set rng = rngSel.Duplicate

Else

Set rng = ActiveDocument.Content

End If

End If

someText = True

If hit = 4 Then

Do

someText = False

If ActiveDocument.Shapes(myGo).Type <> 24 \_

And ActiveDocument.Shapes(myGo).Type <> 3 Then

someText = ActiveDocument.Shapes(myGo).TextFrame.HasText

End If

If someText Then

Set rng = ActiveDocument.Shapes(myGo).TextFrame.TextRange

Else

myGo = myGo + 1

End If

Loop Until someText Or myGo > goes

End If

If someText = True Then

' Now do the F&R with the appropriate conditions set

Options.DefaultHighlightColorIndex = rHiColour

If (rHiColour <> fHiColour) And (fHiColour <> 0) Then

' But first emboss all text in fHiColour

Set rngNow = rng.Duplicate

For Each myPar In rngNow.Paragraphs

If myPar.Range.HighlightColorIndex > 9999 Then

For Each wd In myPar.Range.Words

If wd.HighlightColorIndex > 9999 Then

For Each ch In wd.Characters

If ch.HighlightColorIndex = fHiColour Then

ch.Font.Emboss = True

End If

Next ch

Else

If wd.HighlightColorIndex = fHiColour Then

wd.Font.Emboss = True

End If

End If

Next wd

Else

If myPar.Range.HighlightColorIndex = fHiColour Then

If Len(myPar.Range.Text) > 1 Then myPar.Range.Font.Emboss = True

End If

End If

DoEvents

Next myPar

End If

StatusBar = myspaces & "F&Ring line: " & Str(FRitem) & \_

" of " & Str(lastItem) & " > > > " & fText & \_

"|" & rText

Debug.Print Str(FRitem) & " of " & Str(lastItem) & \_

" > " & fText & " | " & rText

' Now do the F&R

Set rngNow = rng.Duplicate

With rngNow.Find

.ClearFormatting

.Replacement.ClearFormatting

.Format = False

If wasSelectedText Then

.Wrap = False

Else

.Wrap = wdFindContinue

End If

.Text = fText

.Replacement.Text = rText

.MatchWildcards = mchWild

.MatchCase = mchCase

DoEvents

If fStyle > "" Or rstyle > "" Then

If rstyle = "" Then rstyle = myBaseStyle

If fStyle > "" Then .Style = fStyle

If rstyle > "" Then .Replacement.Style = rstyle

Else

' N.B. If changing styles, don't try to

' change bold, italic, etc, etc.

If fBold <> rBold Then

.Font.Bold = fBold

.Replacement.Font.Bold = rBold

End If

If rBold Then .Replacement.Font.Bold = True

If fItalic <> rItalic Then

.Font.Italic = fItalic

.Replacement.Font.Italic = rItalic

End If

If rItalic Then .Replacement.Font.Italic = True

If fSuper <> rSuper Then

.Font.Superscript = fSuper

.Replacement.Font.Superscript = rSuper

End If

If rSuper Then .Replacement.Font.Superscript = True

If fSub <> rSub Then

.Font.Subscript = fSub

.Replacement.Font.Subscript = rSub

End If

If rSub Then .Replacement.Font.Subscript = True

If fUline <> rUline Then

.Font.Underline = fUline

.Replacement.Font.Underline = rUline

End If

If rUline Then .Replacement.Font.Underline = True

If fSmall <> rSmall Then

.Font.SmallCaps = fSmall

.Replacement.Font.SmallCaps = rSmall

End If

If rSmall Then .Replacement.Font.SmallCaps = True

If fAllcaps <> rAllcaps Then

.MatchCase = fAllcaps

.Font.AllCaps = rAllcaps

.Replacement.Font.AllCaps = False

End If

If rAllcaps Then

.MatchCase = False

.Replacement.Font.AllCaps = True

End If

If rDstrike Then .Replacement.Font.StrikeThrough = True

If fStyle = rstyle Then

If fFont = rFont And fFont <> normalFont Then

.Replacement.Font.Name = rFont

End If

If fFont <> rFont Then

.Font.Name = fFont

.Replacement.Font.Name = rFont

End If

If fSize = rSize And fSize <> normalSize Then

.Replacement.Font.Size = rSize

End If

If fSize <> rSize Then

.Font.Size = fSize

.Replacement.Font.Size = rSize

End If

End If

End If

If rHiColour <> fHiColour And fHiColour <> 0 And rText > "" And \_

fHiColour > 0 Then .Font.Emboss = True

If rText > "" And (rHiColour > 0 Or fHiColour > 0) Then

.Replacement.Highlight = True

If rHiColour = 0 Then .Replacement.Highlight = False

End If

If rText > "" Then

If rTxtColour = fTxtColour Then

If fTxtColour > 0 Then \_

.Replacement.Font.Color = fTxtColour

Else

If rTxtColour > 0 Or fTxtColour > 0 Then

.Font.Color = fTxtColour

If BlankIt = False Then .Replacement.Font.Color = \_

rTxtColour

End If

End If

End If

If justEditStrikeText = True Then

.Font.StrikeThrough = True

Else

If editAll = vbNo Then .Font.StrikeThrough = False

End If

.Font.DoubleStrikeThrough = False

.Font.Hidden = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

If rHiColour <> fHiColour Then rng.Font.Emboss = False

End If

Next myGo

End If

Next hit

End If

Next FRitem

StatusBar = ""

' Restore highlight colour to normal

Options.DefaultHighlightColorIndex = oldColour

If myError < 7 Or myError > 16 Then thisRng.Select

If wasSelectedText Then rngSel.Select

totTime = Timer - timeStart

If showTime = True And totTime > 60 Then

MsgBox ((Int(10 \* totTime / 60) / 10) & \_

" minutes")

Else

If doFinalBeep = True Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

End If

End If

ActiveDocument.TrackRevisions = myTrack

Application.ScreenUpdating = True

Exit Sub

' Warn the user about problems that the macro has detected

myErrorReport:

If myError <= 6 Or (myError > 11 And myError < 17) \_

Or myError = 95 Then

rng.Select

Selection.Expand wdParagraph

End If

Select Case myError

Case 2: myPrompt = "No matching replace text"

Selection.MoveStart Unit:=wdParagraph, Count:=-1

' Selection.Expand wdParagraph

Case 4: myPrompt = "A 'Count =' line should say 'yes' or 'no'."

Case 5: myPrompt = "Sorry, Word can't use ^p in a wildcard search." \_

& CR2 & "On Word for Mac, try [^13]." & CR2 \_

& "On Word for Windows, try ^13."

Case 6: myPrompt = "Sorry, Word can't do case insensitive " & \_

"searches with wildcards."

Case 13: myPrompt = "A 'Footnotes =' line should say 'yes' or 'no'."

Case 14: myPrompt = "An 'Endnotes =' line should say 'yes' or 'no'."

Case 15: myPrompt = "A 'Text =' line should say 'yes' or 'no'."

Case 16: myPrompt = "A 'Textboxes =' line should say 'yes' or 'no'."

Case 17: myPrompt = "You have used an unacceptable ^(something) in a search."

Case 19: myPrompt = "Unacceptable pattern match in this F&R line."

Case 21: myPrompt = "Can't find a macro called: " & rText

Case 22: myPrompt = "More Replace groups than Find groups in wildcard F&R."

Case 23: myPrompt = "^? is not a valid special character in wildcard F&R."

Case 24: myPrompt = "^ on its own is not a valid character in Replace."

Case 25: myPrompt = "The style you are trying to apply does not exist." & CR2 \_

& fText & ChrW(124) & rText

Case 95: myPrompt = "Sorry, Word can't use ^{ in a search." \_

& CR2 & "Use ^94{ instead."

Case Else: myPrompt = "Progam error; please inform Paul Bev."

End Select

thisRng.Select

MsgBox myPrompt, vbOKOnly + vbExclamation, "FRedit"

Options.DefaultHighlightColorIndex = oldColour

ActiveDocument.TrackRevisions = myTrack

theList.Activate

Application.ScreenUpdating = True

Exit Sub

' Errors that Word generates end up here

ReportIt:

Application.ScreenUpdating = True

' DoMacro call to unknown macro

If Err.Number < 0 Then myError = 21: GoTo myErrorReport

' Can't find the files it needs

' If Err.Number = 4248 Then myError = 11: GoTo myErrorReport

' Trying to run FRedit from the Zip file

' If Err.Number = 5941 Then myError = 12: GoTo myErrorReport

' Non-existent style

If Err.Number = 5834 Then myError = 25: GoTo myErrorReport

errNow = Err.Number

' If we've found the list, select it...

theList.Activate

Set rng = ActiveDocument.Content

' ... and look for the current line in the list

' which is probably where the problem lies.

gottit = False

Dim myLine As Single

myLine = 1

For j = 1 To ActiveDocument.Paragraphs.Count

myTxt = ActiveDocument.Paragraphs(j)

If Asc(myTxt) <> Asc("|") And Len(myTxt) > 2 Then

If InStr(myTxt, "|") > 0 Then

myLine = myLine + 1

Else

myLine = myLine + 0.5

End If

End If

If myLine > FRitem Then Exit For

Next j

ActiveDocument.Paragraphs(j).Range.Select

If Err.Number = 5625 Or Err.Number = 5692 Then

myError = 17

GoTo myErrorReport

End If

If errNow = 5560 Then

myError = 19

GoTo myErrorReport

End If

If errNow = 5560 Or errNow = 5590 Then

myError = 19

GoTo myErrorReport

End If

' Wildcard error - too many groups

If errNow = 5623 Then myError = 22: GoTo myErrorReport

If errNow = 5692 Then myError = 23: GoTo myErrorReport

If errNow = 5624 Then myError = 24: GoTo myErrorReport

' Display Word's error message

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub FReditSelect()**

' Paul Beverley - Version 21.10.20

' Runs FRedit, but only on multiply selected text

Set rng = ActiveDocument.Content

rng.Font.StrikeThrough = True

Selection.Font.StrikeThrough = False

Call FRedit

rng.Font.StrikeThrough = False

End Sub

**Sub FReditHeadingsOnly()**

' Paul Beverley - Version 30.01.21

' Run FRedit, but only on similarly formatted text

Set rng = ActiveDocument.Content

rng.Font.StrikeThrough = True

Application.Run "SelectSimilarFormatting"

Selection.Font.StrikeThrough = False

Selection.Collapse wdCollapseStart

Call FRedit

rng.Font.StrikeThrough = False

End Sub

**Sub FReditListChecker()**

' Paul Beverley - Version 18.02.21

' Checks for possible anomalies in a FRedit list

CR = vbCr

nmlStyle = ActiveDocument.Styles(wdStyleNormal)

nmlFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

nmlSize = ActiveDocument.Styles(wdStyleNormal).Font.Size

Set rng = ActiveDocument.Range(0, Selection.End)

parNow = rng.Paragraphs.Count

For i = parNow To ActiveDocument.Paragraphs.Count

foundError = False

Set rng = ActiveDocument.Paragraphs(i).Range.Duplicate

Set rng2 = rng.Duplicate

myCmnt = ""

If rng.Style <> nmlStyle Then

myCmnt = myCmnt & CR & "Style: " & rng.Style

Else

fntName = rng.Font.Name

If fntName <> nmlFont Then

For j = 1 To rng.Characters.Count

funnyFont = rng.Characters(j).Font.Name

If funnyFont <> nmlFont Then

rng2.Start = rng.Start + j - 1

rng2.End = rng.End

Exit For

End If

Next j

myCmnt = myCmnt & CR & "Font name? (" & funnyFont & "))"

End If

fntSize = rng.Font.Size

If fntSize <> nmlSize Then

myCmnt = myCmnt & CR & "Font size? (" & fntSize & ")"

End If

End If

myLine = rng.Text

If Len(myLine) > 1 And InStr(myLine, "|") = 0 And \_

Left(myLine, 1) <> "#" Then

myCmnt = myCmnt & CR & "Missing pad character?"

End If

' Check if find & replace are different colour of highlight

If Len(myLine) > 3 And InStr(myLine, "|") > 0 Then

colourLeft = rng.Characters(2).HighlightColorIndex

colourRight = rng.Characters(Len(myLine) - 2).HighlightColorIndex

If colourLeft <> colourRight Then

myCmnt = myCmnt & CR & \_

"Did you really mean the CHANGE highlight colour?"

End If

End If

If myCmnt > "" Then

If Len(myCmnt) - Len(Replace(myCmnt, CR, "")) = 1 Then

myCmnt = Mid(myCmnt, 2)

End If

rng2.Select

ActiveDocument.ActiveWindow.LargeScroll down:=2

rng2.Select

foundError = True

Beep

myResponse = MsgBox(myCmnt & CR & CR & "Continue?", vbQuestion \_

+ vbYesNoCancel, "FReditListChecker")

If myResponse <> vbYes Then Exit Sub

End If

Next i

Selection.EndKey Unit:=wdStory

Beep

If foundError = False Then

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

myResponse = MsgBox("No more problems", \_

, "FReditListChecker")

End If

End Sub

**Sub FReditSimple()**

' Paul Beverley - Version 10.12.16

' Performs a list of F&Rs

Set rng = ActiveDocument.Content

nowTrack = ActiveDocument.TrackRevisions

If InStr(Left(rng, 100), "|") > 0 Then

MsgBox "Place the cursor in the file to be edited."

Exit Sub

End If

gottaList = False

For Each myWnd In Application.Windows

If InStr(Left(myWnd.Document.Content.Text, 100), "|") > 0 Then

gottaList = True

Exit For

End If

Next myWnd

If gottaList = False Then MsgBox "Can't find a changes list.": Exit Sub

For Each myPara In myWnd.Document.Paragraphs

myText = Replace(myPara.range.Text, vbCr, "")

If Left(myText, 1) = "#" Then Beep: Exit Sub

If Len(myText) > 2 And Left(myText, 1) <> "|" Then

padPos = InStr(myText, "|")

If padPos = 0 Then myPara.range.Select: MsgBox "No vertical bar!": Exit Sub

thisColour = myPara.range.HighlightColorIndex

If thisColour > 999 Then myPara.range.Select: \_

MsgBox "Please highlight the whole line.": Exit Sub

Options.DefaultHighlightColorIndex = thisColour

myFind = Left(myText, padPos - 1)

myRepl = Mid(myText, padPos + 1)

doWC = (Asc(myFind) = Asc("~"))

doAllCase = (Asc(myFind) = 172)

If doWC Or doAllCase Then myFind = Mid(myFind, 2)

ActiveDocument.TrackRevisions = nowTrack And \_

Not myPara.range.Font.StrikeThrough

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = myFind

.Replacement.Text = myRepl

.Replacement.Highlight = True

.MatchCase = Not doAllCase

.MatchWildcards = doWC

.Execute Replace:=wdReplaceAll

End With

End If

Next myPara

Beep

End Sub

**Sub MiniFRedit()**

' Paul Beverley - Version 15.10.18

' Adds attributes to certain words

doSpeedup = False

' Find #

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "#^p"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = False

.MatchCase = False

.MatchWildcards = False

.Execute

End With

rng.Collapse wdCollapseEnd

If rng.Find.Found = False Then

Beep

MsgBox ("Can't find the list!")

Exit Sub

End If

' Remember cursor position

Set rngOld = Selection.range.Duplicate

' Locate the list

rng.End = ActiveDocument.Content.End

listText = rng

numLines = Len(listText) - Len(Replace(listText, vbCr, ""))

rng.Collapse wdCollapseStart

startList = rng.Start

oldColour = Options.DefaultHighlightColorIndex

fnNum = ActiveDocument.Footnotes.Count

enNum = ActiveDocument.Endnotes.Count

nmlFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

nmlSize = ActiveDocument.Styles(wdStyleNormal).Font.Size

ActiveDocument.TrackRevisions = False

' To speed up search

If doSpeedup Then Selection.HomeKey Unit:=wdStory

For i = 1 To numLines

rng.Expand wdParagraph

If Len(rng) > 1 Then

Set tst = rng.Duplicate

tst.MoveEnd , -1

myFind = tst.Text

If Left(myFind, 1) = "!" Then

myFind = Mid(myFind, 2)

doUndo = True

Else

doUndo = False

End If

If Right(myFind, 1) = "-" Then

myFind = Left(myFind, Len(myFind) - 1)

Else

myFind = myFind & ">"

End If

If Left(myFind, 1) = "-" Then

myFind = Mid(myFind, 2)

Else

myFind = "<" & myFind

End If

tst.MoveStart , 1

tst.End = tst.Start + 1

' Check highlight and font colours

hiColor = tst.HighlightColorIndex

Options.DefaultHighlightColorIndex = hiColor

fontColour = tst.Font.Color

' Check the attributes on this item

myBold = tst.Font.Bold

myItal = tst.Font.Italic

mySize = tst.Font.Size

fntName = tst.Font.Name

myStrike = tst.Font.StrikeThrough

mySuper = tst.Font.Superscript

mySub = tst.Font.Subscript

myUline = tst.Font.Underline

' Now do the F&Rs

For j = 1 To 3

If j = 1 And fnNum = 0 Then j = 2

If j = 2 And enNum = 0 Then j = 3

Select Case j

Case 1: Set rng2 = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng2 = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng2 = ActiveDocument.Content: rng2.End = startList

End Select

DoEvents

If Len(myFind) > 1 Then

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Replacement.Text = ""

.Wrap = False

' Apply (or remove, for unDo) the attribute

If hiColor <> wdNoHighlight Then

.Replacement.Highlight = True

If doUndo Then .Replacement.Highlight = False

End If

If fontColour <> wdColorAutomatic Then

.Replacement.Font.Color = fontColour

If doUndo Then .Replacement.Font.Color = wdColorAutomatic

End If

If myBold Then

.Replacement.Font.Bold = True

If doUndo Then .Replacement.Font.Bold = False

End If

If myItal Then

.Replacement.Font.Italic = True

If doUndo Then .Replacement.Font.Italic = False

End If

If myStrike Then

.Replacement.Font.StrikeThrough = True

If doUndo Then .Replacement.Font.StrikeThrough = False

End If

If rng.Font.Name <> nmlFont Then

.Replacement.Font.Name = fntName

If doUndo Then .Replacement.Font.Name = nmlFont

End If

If rng.Font.Size <> nmlSize Then

.Replacement.Font.Size = mySize

If doUndo Then .Replacement.Font.Size = nmlSize

End If

If mySuper Then

.Replacement.Font.Superscript = True

If doUndo Then .Replacement.Font.Superscript = False

End If

If mySub Then

.Replacement.Font.Subscript = True

If doUndo Then .Replacement.Font.Subscript = False

End If

If myUline Then

.Replacement.Font.Underline = True

If doUndo Then .Replacement.Font.Underline = False

End If

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

Next j

End If

rng.Collapse wdCollapseEnd

Next i

Options.DefaultHighlightColorIndex = oldColour

If doSpeedup Then rngOld.Select

Beep

End Sub

**Sub QuotationMarker()**

' Paul Beverley - Version 21.05.20

' Marks all quotes + displayed text

' Do you want displayed text marked?

markDisplayed = True

' Add coding to existing displayed quotes?

addCodes = True

addCodes = False

preCode = "<DIS>"

postCode = "</DIS>"

codeOnNextLine = False

' Minimum length of quotes (words)

minLength = 3

' Minimum indent of quotes (cm)

minIndent = 1.05

' Colour the font of quotations

colourFont = False

colourFont = True

myColour = wdColorBlue

' Colour of the possible plural possessive problems

possessiveColour = wdColorRed

' Possible possessive errors only shown if within

' this many paragraphs

maxParasPossessive = 5

' Add a highlight

highlightText = False

myHighlight = wdYellow

' Strike it through

strSingle = True

strDouble = False

' What kind of quote?

doDoubleQuotes = True

doSingleQuotes = True

' Do you want the notes checked (if there are any)?

doFootnotes = True

doEndnotes = True

myResponse = MsgBox("Strike through quotations?", vbQuestion \_

+ vbYesNoCancel, "QuotationMarker")

If myResponse <> vbYes Then Exit Sub

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

nowColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myHighlight

soCalledWords = "---,.;:!?…" & ChrW(8220) & ChrW(8221)

' Clear old codes, in case you're running it a second time

If addCodes = True Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = preCode

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = postCode

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

If markDisplayed = True Then

For Each myPara In ActiveDocument.Paragraphs

If myPara.Range.ParagraphFormat.LeftIndent > \_

CentimetersToPoints(minIndent) Then

myPara.Range.Font.Emboss = True

If addCodes = True Then

myPara.Range.InsertBefore Text:=preCode

myPara.Range.InsertAfter Text:=postCode

End If

End If

Next

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = preCode & postCode

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

If codeOnNextLine = False Then

With rng.Find

.Text = "^p" & postCode

.Replacement.Text = postCode & "^p"

.Execute Replace:=wdReplaceAll

End With

End If

End If

gottaFunny = False

funnyCount = 0

For i = 1 To 3

If i = 1 And (ActiveDocument.Footnotes.Count = 0 Or \_

doFootnotes = False) Then i = 2

If i = 2 And (ActiveDocument.Endnotes.Count = 0 Or \_

doEndnotes = False) Then i = 3

Select Case i

Case 1: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

' Single quotes

If doSingleQuotes = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8216) & "\*" & ChrW(8217)

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Emboss = True

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

' Check for any apostrophe-letter that is half struck-through

If doSingleQuotes = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8217) & "^$"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

' If this is half struckthrough

If rng.Font.Emboss > 999 Then

' Extend to the following close single quote

rng.MoveEndUntil cset:=ChrW(8217), Count:=wdForward

rng.MoveEnd wdCharacter, 1

rng.Font.Emboss = True

rng.Start = rng.End - 1

rng.Collapse wdCollapseStart

Else

rng.Collapse wdCollapseEnd

End If

rng.Find.Execute

DoEvents

Loop

' Now check for plural possessives: s'

Select Case i

Case 1: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

' Add a dummy open quote at the end of the text

rng.InsertAfter ChrW(8216)

theVeryEnd = rng.End

' Check from each of the s-apostrophes to the next open quote

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "s" & ChrW(8217) & "\*" & ChrW(8216)

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.End = rng.Start + 1

isQuote = rng.Font.Emboss

rng.End = endNow

rng.MoveStart wdCharacter, 2

endQuoteWas = rng.Start

myText = rng.Text

If InStr(myText, ChrW(8217)) > 0 And isQuote = True Then

' How many apostrophe/close quotes are there?

aposNum = Len(myText) - Len(Replace(myText, ChrW(8217), ""))

myPointer = endQuoteWas

endQuoteNow = myPointer

For j = 1 To aposNum

aposPos = InStr(myText, ChrW(8217))

nextchar = Mid(myText, aposPos + 1, 1)

myPointer = myPointer + aposPos

' Is nextChar non-alpha? If so, this is the \*real\* end of the quote

If LCase(nextchar) = UCase(nextchar) Then

endQuoteNow = myPointer

End If

myText = Mid(myText, aposPos + 1)

Next

If endQuoteNow > endQuoteWas Then

endNow = rng.End

rng.End = endQuoteNow

rng.Font.Emboss = True

myChar = Right(rng.Text, 2)

thisLot = rng.Text

numParas = Len(thisLot) - Len(Replace(thisLot, Chr(13), ""))

If Asc(myChar) = Asc("s") And numParas < maxParasPossessive Then

gottaFunny = True

rng.Font.Emboss = False

rng.Font.Color = possessiveColour

funnyCount = funnyCount + 1

End If

End If

rng.End = myPointer

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

rng.Start = theVeryEnd - 2

rng.End = theVeryEnd - 1

rng.Delete

DoEvents

End If

' Double quotes

If doDoubleQuotes = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8220) & "\*" & ChrW(8221)

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Emboss = True

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

' Remove embossing from short quotes

If minLength > 0 Then

Select Case i

Case 1: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Emboss = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

' Find the true word count

myText = rng.Text

wdsOff = 1

For j = 1 To Len(myText)

If InStr(soCalledWords, Mid(myText, j, 1)) > 0 Then \_

wdsOff = wdsOff + 1

Next j

If j > 3 Then

' To correct Word's weird wordcount'hello' = count OK

' 'hello?' error by one word, and 'hello??' = count OK

charOne = Mid(myText, j - 3, 1)

charTwo = Mid(myText, j - 2, 1)

oneIsAlpha = (LCase(charOne) <> UCase(charOne))

twoIsNotAlpha = (LCase(charTwo) = UCase(charTwo))

If oneIsAlpha And twoIsNotAlpha Then wdsOff = wdsOff + 1

If rng.Words.Count - wdsOff < minLength - 1 Then

rng.Font.Emboss = False

rng.Collapse wdCollapseEnd

End If

End If

rng.Find.Execute

DoEvents

Loop

End If

' Apply whatever effect is wanted on the embossed text

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Font.Emboss = True

.Text = ""

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Emboss = False

If highlightText = True Then .Replacement.Highlight = True

If strSingle = True Then .Replacement.Font.StrikeThrough = True

If strDouble = True Then .Replacement.Font.DoubleStrikeThrough = True

If colourFont = True Then .Replacement.Font.Color = myColour

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

' Warn the user of possible plural possessives

If gottaFunny = True Then

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Font.Color = possessiveColour

.MatchWildcards = False

.Execute

End With

Selection.End = Selection.Start - 1

Selection.Find.Execute

MsgBox "Please check possible plural possessives (" \_

& Trim(Str(funnyCount)) & ")"

End If

Options.DefaultHighlightColorIndex = nowColour

ActiveDocument.TrackRevisions = myTrack

Beep

End Sub

**Sub GermanQuotes()**

' Paul Beverley - Version 14.02.13

' Switch UK quotes to German quotes

Options.AutoFormatAsYouTypeReplaceQuotes = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8220)

.Wrap = wdFindContinue

.Replacement.Text = ChrW(8222)

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8221)

.Wrap = wdFindContinue

.Replacement.Text = ChrW(8220)

.Execute Replace:=wdReplaceAll

End With

Options.AutoFormatAsYouTypeReplaceQuotes = True

End Sub

**Sub FrenchQuotes()**

' Paul Beverley - Version 14.02.13

' Switch UK quotes to French quotes

Options.AutoFormatAsYouTypeReplaceQuotes = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8220)

.Wrap = wdFindContinue

.Replacement.Text = ChrW(171)

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8221)

.Wrap = wdFindContinue

.Replacement.Text = ChrW(187)

.Execute Replace:=wdReplaceAll

End With

Options.AutoFormatAsYouTypeReplaceQuotes = True

End Sub

**Sub AutoCurlyQuotesOFF()**

' Paul Beverley - Version 14.02.13

' Switch off auto curly quotes

Options.AutoFormatAsYouTypeReplaceQuotes = False

End Sub

**Sub AutoCurlyQuotesON()**

' Paul Beverley - Version 14.02.13

' Switch on auto curly quotes

Options.AutoFormatAsYouTypeReplaceQuotes = True

End Sub

**Sub StrikeSingle()**

' Paul Beverley - Version 15.08.17

' Adds or removes strike-through

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Dim v As Variable

varsExist = False

For Each v In ActiveDocument.Variables

If v.Name = "colNum" Then varsExist = True: Exit For

Next v

If varsExist = True And Selection.Start = Selection.End Then

wasStart = ActiveDocument.Variables("selStart")

wasEnd = ActiveDocument.Variables("selEnd")

' If the cursor is inside the area, strike it

If Selection.Start >= wasStart Or Selection.End \_

<= wasEnd - 1 Then

Selection.Start = wasStart

Selection.End = wasEnd

Selection.range.Font.StrikeThrough = True

End If

Else

' If all selected, remove strikethrough

If Selection = ActiveDocument.Content Then

Selection.Font.StrikeThrough = False

Else

If Selection.Start = Selection.End Then Selection.Expand wdParagraph

stateNow = Selection.Characters(1).Font.StrikeThrough

Set rng = ActiveDocument.Content

If Selection.Start = 0 And Selection.End = rng.End Then

Selection.Font.StrikeThrough = False

Else

Selection.Font.StrikeThrough = Not (stateNow)

End If

Selection.Collapse wdCollapseEnd

End If

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub StrikeAndColour()**

' Paul Beverley - Version 01.09.16

' Adds strikethrough and colour

If Selection.Start = Selection.End Then Selection.Expand wdParagraph

Set rng = Selection.range.Duplicate

Call ColourPlus

rng.Select

Call StrikeSingle

End Sub

**Sub StrikeThroughAllURLs()**

**'** Paul Beverley - Version 18.04.24

' Strikes through all URLs to protect them from changes

highlightToo = True

myColour = wdBrightGreen

charsInURLs = "[%./:a-zA-Z0-9\_\-+\?=&,]"

myFind = "[wh][wt][wt][ps.]" & charsInURLs & "{1,}"

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

For myArea = 1 To 3

doThisArea = False

' Main text area

If myArea = 1 Then

If Selection.Start = Selection.End Then

myResponse = MsgBox("Scan the whole document?!", \_

vbQuestion + vbYesNo, "StrikeThroughAllURLs")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

Else

Set rng = Selection.Range.Duplicate

End If

doThisArea = True

End If

' Footnotes, if any

If ActiveDocument.Footnotes.Count > 0 And myArea = 2 Then

doThisArea = True

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

StatusBar = "Scanning footnotes"

End If

' Endnotes, if any

If ActiveDocument.Endnotes.Count > 0 And myArea = 3 Then

doThisArea = True

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

StatusBar = "Scanning endnotes"

End If

If doThisArea = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.StrikeThrough = True

If highlightToo Then .Replacement.Highlight = True

.Forward = True

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

Next myArea

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub EquationsStrikeThroughAll()**

' Paul Beverley - Version 31.08.16

' Ensure that no equations are edited by FRedit etc.

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

numEqs = ActiveDocument.OMaths.Count

For i = numEqs To 1 Step -1

ActiveDocument.OMaths(i).range.Font.StrikeThrough = True

DoEvents

StatusBar = " " & i

Next i

ActiveDocument.TrackRevisions = myTrack

Beep

StatusBar = ""

End Sub

**Sub CodeSegmentProtect()**

' Paul Beverley - Version 06.01.18

' Applies strike-through to computer code sections

ActiveDocument.TrackRevisions = False

styleToHide = "computer\_code"

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Replacement.Text = ""

.MatchWildcards = False

.Style = styleToHide

.Wrap = wdFindContinue

.Forward = True

.Replacement.Font.StrikeThrough = True

.Execute

End With

If rng.Find.Found = False Then

MsgBox "No such style found: " & styleToHide

Exit Sub

Else

rng.Select

myResponse = MsgBox("Hide this style?", \_

vbQuestion + vbYesNoCancel)

If myResponse <> vbYes Then Exit Sub

End If

rng.Find.Execute Replace:=wdReplaceAll

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\/\/[!^13]{1,}"

.MatchWildcards = True

.Replacement.Highlight = True

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.Replacement.Font.StrikeThrough = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Forward = True

.Text = "^p^33"

.MatchWildcards = False

.Replacement.Text = "^pzczc!"

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Forward = True

.Text = "zczc\![!^13]{1,}"

.MatchWildcards = True

.Replacement.Highlight = True

.Replacement.Text = ""

.Replacement.Font.StrikeThrough = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc"

.MatchWildcards = False

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

End With

ActiveDocument.TrackRevisions = True

End Sub

**Sub HighlightCertainCharacters()**

' Paul Beverley - Version 21.09.18

' Highlights certain characters with attributes

superscriptZeros = wdBrightGreen

italicCommas = wdBrightGreen

boldColons = wdPink

notBoldColons = 0

subSuperscriptSpace = wdGray50

' Just super and subscript numbers in italic

subSuperscriptNumberItalic = wdYellow

' All numbers in italic

allNumberItalic = wdGray25

' Various symbols - your choice

variousSymbols1 = wdPink

mySymbols1 = "[áÁàÀâÂäÄÃãÅåçÇéÉèÈêÊëËíÍìÌîÎñÑóÓòÒôÔöÖõÕøØßúÚùÙûÛüÜýÝÿŸ/–]"

variousSymbols2 = wdYellow

mySymbols2 = "[=\\*\>\<+" & ChrW(8722) & "]"

' Various symbols in italic - your choice

' (e.g. parentheses, exclamation mark and ½ symbol = 189)

specificSymbolsInItalic = wdTurquoise

mySymbols3 = "[\(\)\!" & ChrW(189) & "]"

oldColour = Options.DefaultHighlightColorIndex

If superscriptZeros > 0 Then

Options.DefaultHighlightColorIndex = superscriptZeros

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "0"

.Font.Superscript = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

If italicCommas > 0 Then

Options.DefaultHighlightColorIndex = italicCommas

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ","

.Font.Italic = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

If boldColons > 0 Then

Options.DefaultHighlightColorIndex = boldColons

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ":"

.Font.Bold = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

If notBoldColons > 0 Then

Options.DefaultHighlightColorIndex = notBoldColons

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ":"

.Font.Bold = False

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

If subSuperscriptSpace > 0 Then

Options.DefaultHighlightColorIndex = subSuperscriptSpace

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " "

.Font.Superscript = True

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Font.Subscript = True

.Text = " "

.Execute Replace:=wdReplaceAll

End With

End If

' Super and subscript numbers that are italic

If subSuperscriptNumberItalic > 0 Then

Options.DefaultHighlightColorIndex = subSuperscriptNumberItalic

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]"

.Font.Superscript = True

.Font.Italic = True

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Font.Subscript = True

.Font.Italic = True

.Text = "[0-9]"

.Execute Replace:=wdReplaceAll

End With

End If

' All numbers that are italic

If allNumberItalic > 0 Then

Options.DefaultHighlightColorIndex = allNumberItalic

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]"

.Font.Italic = True

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

If variousSymbols1 > 0 Then

Options.DefaultHighlightColorIndex = variousSymbols1

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySymbols1

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

If variousSymbols2 > 0 Then

Options.DefaultHighlightColorIndex = variousSymbols2

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySymbols2

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

If specificSymbolsInItalic > 0 Then

Options.DefaultHighlightColorIndex = specificSymbolsInItalic

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySymbols3

.Font.Italic = True

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub HighlightAllItalic()**

' Paul Beverley - Version 07.05.23

' Highlights all text in italic (including in notes)

myColour = wdYellow

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindContinue

.Format = True

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

If ActiveDocument.Footnotes.count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

With rng.Find

.Text = ""

.Font.Italic = True

.Replacement.Text = ""

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

End If

If ActiveDocument.Endnotes.count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.Text = ""

.Font.Italic = True

.Replacement.Text = ""

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

End If

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub HighlightNonRomanPunctuation()**

**'** Paul Beverley - Version 05.11.24

' Highlights non-roman punctuation

myItalic = wdGray25

myBold = wdGray25

mySymbols = "[,.:\!\?]"

oldColour = Options.DefaultHighlightColorIndex

fNotes = ActiveDocument.Footnotes.Count

eNotes = ActiveDocument.Endnotes.Count

For j = 1 To 3

If j = 1 And fNotes = 0 Then j = 2

If j = 2 And eNotes = 0 Then j = 3

Select Case j

Case 1: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

Options.DefaultHighlightColorIndex = myItalic

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySymbols

.Font.Italic = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

Options.DefaultHighlightColorIndex = myBold

.ClearFormatting

.Font.Bold = True

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub EquationsHighlightAll()**

' Paul Beverley - Version 10.10.20

' Highlights all maths items

doMathTypeOnes = True

myColour1 = wdWhite

myColour1 = wdTurquoise

myColour2 = wdYellow

myColour2 = wdGray25

doEqEditorOnes = True

myColour3 = wdGray25

myColour3 = wdBrightGreen

doSymbolFont = True

myColour4 = wdPink

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

numInlineShapes = ActiveDocument.InlineShapes.Count

numMaths1 = 0

numMaths2 = 0

If numInlineShapes > 0 And doMathTypeOnes = True Then

For Each myMath In ActiveDocument.InlineShapes

Select Case myMath.Type

Case 1

myMath.Range.HighlightColorIndex = myColour1

numMaths1 = numMaths1 + 1

Case 3

myMath.Range.HighlightColorIndex = myColour2

numMaths2 = numMaths2 + 1

Case Else

myMath.Range.HighlightColorIndex = myColour4

End Select

DoEvents

Next myMath

MsgBox "Possible MathType items found" & vbCr & "Editable: " \_

& numMaths1 & vbCr & "Non-editable: " & numMaths2

End If

numMaths = ActiveDocument.OMaths.Count

If numMaths > 0 And doEqEditorOnes = True Then

For Each myMath In ActiveDocument.OMaths

myMath.Range.HighlightColorIndex = myColour3

DoEvents

Next myMath

MsgBox "Equation Editor items found: " & numMaths

End If

If doSymbolFont = True Then

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour4

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Name = "Symbol"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

End If

Beep

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub EquationsConvertAll()**

' Paul Beverley - Version 24.10.16

' Converts all Equation Editor items to their text equivalent.

highlightConversion = True

myColour = wdYellow

numMaths = ActiveDocument.OMaths.Count

If numMaths > 0 Then

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

For Each myMath In ActiveDocument.OMaths

myMath.range.Select

Selection.MoveStart , -1

Selection.Cut

startHere = Selection.Start

Selection.PasteSpecial DataType:=wdPasteText

Selection.Start = startHere

If highlightConversion = True Then

Selection.range.HighlightColorIndex = myColour

End If

Next myMath

ActiveDocument.TrackRevisions = myTrack

MsgBox numMaths & " equation editor items converted"

End If

End Sub

**Sub EquationSpacer()**

' Paul Beverley - Version 06.12.22

' Adds spaces either side of equations that butt up to some text

myEEColour = wdYellow

doHighlight = True

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

numMaths = ActiveDocument.InlineShapes.Count

If numMaths > 0 = True Then

For Each myMath In ActiveDocument.InlineShapes

Set rng = myMath.Range.Duplicate

myColour = myMath.Range.HighlightColorIndex

myEnd = rng.End

myStart = rng.Start

rng.Start = myEnd

rng.End = myEnd + 1

myText = rng.Text

' Look at what follows the equation

If LCase(myText) <> UCase(myText) Then

rng.InsertBefore Text:=ChrW(32)

rng.Font.Color = wdColorPink

rng.Start = rng.Start - 1

rng.HighlightColorIndex = wdNoHighlight

myMath.Range.HighlightColorIndex = myColour

spacesAdded = spacesAdded + 1

End If

rng.Start = myStart - 1

rng.End = myStart

myText = rng.Text

' Look at what precedes the equation

If LCase(myText) <> UCase(myText) Or myText = "." Or myText = "," Then

rng.InsertAfter Text:=ChrW(32)

rng.Font.Color = wdColorPink

spacesAdded = spacesAdded + 1

End If

Next myMath

End If

numEqns = ActiveDocument.OMaths.Count

eeSpacesAdded = 0

If numEqns > 0 Then

Set rng = ActiveDocument.Content

For Each myMath In ActiveDocument.OMaths

Set rngMath = myMath.Range.Duplicate

rng.End = rngMath.Start

rng.Start = rng.End - 1

If LCase(rng) <> UCase(rng) Then

rng.Text = rng.Text & " "

If doHighlight = True Then rng.HighlightColorIndex = myEEColour

eeSpacesAdded = eeSpacesAdded + 1

End If

rng.End = rngMath.End + 1

rng.Start = rng.End - 1

If LCase(rng) <> UCase(rng) Then

rng.Text = " " & rng.Text

If doHighlight = True Then rng.HighlightColorIndex = myEEColour

eeSpacesAdded = eeSpacesAdded + 1

End If

DoEvents

Next myMath

End If

ActiveDocument.TrackRevisions = myTrack

myMsg = vbCr

If spacesAdded > 0 Then myMsg = spacesAdded & " spaces added to MathType equations" & vbCr & myMsg

If eeSpacesAdded > 0 Then myMsg = eeSpacesAdded & " spaces added to Equation Editor equations" & vbCr & myMsg

MsgBox myMsg

End Sub

**Sub FullNameType()**

**'** Paul Beverley - Version 31.03.11

' Types out the full filename of the open file

Selection.TypeText Text:=vbCrLf & ActiveDocument.FullName & vbCrLf

End Sub

**Sub FReditListRun()**

**'** Paul Beverley - Version 31.03.11

' Load, run and close a specific FRedit list

listFile = "C:\Documents and Settings\Paul\My Documents\myList.doc"

Set thisFile = ActiveDocument

Documents.Open listFile

Set FRlist = ActiveDocument

thisFile.Activate

Call FRedit

FRlist.Activate

ActiveDocument.Close SaveChanges:=False

' Beep to let you know it has finished

Beep

End Sub

**Sub FReditListMenu()**

' Paul Beverley - Version 06.02.21

' Provides a menu to run different FRedit lists

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\MyLists"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/MyLists"

Dim myList(30) As String

Dim myLetter(30) As String

n = 0

s = ";"

allLetters = s

myName = Dir(myFolder)

Do While myName <> ""

n = n + 1

myList(n) = myName

myLetter(n) = LCase(Left(myName, 1))

If InStr(allLetters, s & myLetter(n) & s) > 0 Then

myLetter(n) = myLetter(n) & myLetter(n)

If InStr(allLetters, s & myLetter(n) & s) > 0 Then

myResponse = MsgBox("Please don't have three items" & \_

" with the same initial letter.", \_

vbOKCancel, "FReditListMenu")

Exit Sub

End If

End If

allLetters = allLetters & myLetter(n) & s

myName = Dir()

Loop

For i = 1 To n

thisName = Replace(myList(i), ".docx", "")

thisName = Replace(thisName, ".doc", "")

myMenu = myMenu & myLetter(i) & \_

" – " & thisName & vbCr

Next i

myMenu = myMenu & vbCr & "Which list (letter)?"

myChoice = InputBox(myMenu, "FReditListMenu")

For i = 1 To n

If myLetter(i) = LCase(myChoice) Then

goodChoice = True

doList = myList(i)

Exit For

End If

Next i

myChoice = myList(i)

If myChoice = "" Then Exit Sub

Set myDoc = ActiveDocument

Documents.Open FileName:=myFolder & myChoice

Set listDoc = ActiveDocument

myDoc.Activate

Call FRedit

listDoc.Activate

ActiveDocument.Close SaveChanges:=False

End Sub

**Sub MultiFileFRedit()**

' Paul Beverley - Version 10.07.21

' Multi-file version of FRedit

acceptAllRevs = True

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Run FRedit on ALL the files in this folder:" & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "Multifile FRedit")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Run FRedit on just the files listed here?", \_

vbQuestion + vbYesNoCancel, "Multifile FRedit")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

myResponse = MsgBox("Are you SURE you want to run FRedit on all" & Str(numFiles) \_

& " files?" & vbCr & vbCr & "There's no going back, you know!", \_

vbQuestion + vbYesNoCancel, "WARNING! WARNING! WARNING!")

If myResponse <> vbYes Then Exit Sub

For i = 1 To numFiles

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set myDoc = Application.Documents.Open(FileName:=thisFile)

StatusBar = allMyFiles(i)

DoEvents

If acceptAllRevs = True Then ActiveDocument.AcceptAllRevisions

Application.Run MacroName:="FRedit"

myDoc.Save

ActiveWindow.Close

Next

Selection.HomeKey Unit:=wdStory

MsgBox ("FRedited " + Str(numFiles) + " files")

End Sub

**Sub SpellingSuggest()**

' Paul Beverley - Version 12.02.24

' Checks/corrects spellings or adds/subtracts FRedit suggested change

If LCase(Selection) = UCase(Selection) Then Selection.MoveLeft , 1

Set rngWd = Selection.Range.Duplicate

rngWd.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rngWd.Text, 1)) > 0

rngWd.MoveEnd , -1

DoEvents

Loop

a = rngWd.Characters(1)

b = rngWd.Characters(2)

If UCase(a) = a And UCase(b) = b Then \_

rngWd.Characters(2) = LCase(b)

myWord = rngWd

newWord = myWord

' Check spelling language

Set rng = ActiveDocument.Content

' (but only first character, in case of split language)

rng.End = rng.Start + 1

langName = Languages(rng.LanguageID).NameLocal

rng.End = ActiveDocument.Content.End

rng.Start = ActiveDocument.Content.End - 2

langName1 = Languages(rng.LanguageID).NameLocal

langName2 = Languages(Selection.LanguageID).NameLocal

StatusBar = langName & " " & langName1 & " " & langName2

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

wdsInPara = rng.Words.Count

myPara = rng

initChar = Left(myWord, 1)

lastChar = Right(myWord, 1)

spellOK = Application.CheckSpelling(myWord, MainDictionary:=langName)

Set suggList = Application.GetSpellingSuggestions(myWord, \_

MainDictionary:=langName)

If suggList.Count > 0 And Not (spellOK) Then \_

newWord = suggList.Item(1).Name

' If initial letter is a capital keep the capital

If LCase(initChar) <> initChar Then \_

newWord = UCase(Left(newWord, 1)) & Mid(newWord, 2)

' If final letter is a capital, make it all caps

If LCase(lastChar) <> lastChar Then \_

newWord = UCase(newWord)

' If it's the only word in the para (a FRedit list)

If wdsInPara < 3 And ActiveDocument.Words.Count > 10 Then

oldWord = myWord

Selection.Expand wdWord

Selection.Collapse wdCollapseEnd

If Asc(Selection) = 32 Then

Selection.MoveEnd , 1

oldWord = oldWord & "^32"

newWord = newWord & "^32"

End If

Selection.Expand wdParagraph

Selection.TypeText ChrW(172) & oldWord & "|" & newWord & CR

Selection.MoveRight , 1

If newWord = myWord Then Beep

Else

If newWord <> myWord Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

rngWd.Text = newWord

rngWd.Select

Selection.Collapse wdCollapseEnd

Else

spellOK = Application.CheckSpelling(myWord, \_

MainDictionary:=langName)

If spellOK Then

Selection.Collapse wdCollapseEnd

Beep

Else

If newWord = myWord Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Do

Loop Until Timer > myTime + 0.4

Beep

If langName = "English (United States)" \_

And Application.CheckSpelling(myWord, \_

MainDictionary:="English (United Kingdom)") Then

MsgBox ("Word's spellcheck is playing sillies!")

Exit Sub

Else

MsgBox ("Word offers no suggestion")

End If

End If

End If

End If

End If

End Sub

**Sub SpellAlyse()**

**'** Paul Beverley - Version 25.05.23

' Complete spellchecking system

ignoreNumbers = True

spellingListName = "SpellAlyse"

spellingFreqName = "SpellAlyse frequencies"

ignoreDoc = "zzSwitchList"

myFind = "´a,´e,¨a,¨e,¨o,¨u,ˆo"

myReplace = "á,é,ä,ë,ö,ü,ô"

CR = vbCr

CR2 = CR & CR

sp1 = " "

sp2 = sp1 & sp1

myScreenOff = True

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

Dim lossLen As Long

Dim newLen As Long

numParas = FUT.Paragraphs.Count

numWords = FUT.Content.ComputeStatistics(wdStatisticWords)

myProfile = numWords / numParas

If myProfile < 1.1 Then GoTo autoCorrect

' List all possible spelling errors

Set rng = ActiveDocument.Content

rng.End = 2

thisLanguage = rng.LanguageID

langText = Languages(rng.LanguageID).NameLocal

langPrompt = "Spellcheck with " & langText & " dictionary. OK?"

If doingSeveralMacros = False Then

myResponse = MsgBox(langPrompt, vbQuestion + vbYesNoCancel, \_

"SpellAlyse")

If myResponse <> vbYes Then Exit Sub

End If

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

timeStart = Timer

allExcepts = CR

'Collect all words from all word lists

For Each myDoc In Documents

Debug.Print myDoc.Name

pNum = myDoc.Paragraphs.Count

myNum = 3

If pNum < 3 Then myNum = pNum

Set rng = myDoc.Paragraphs(myNum).Range

rng.Start = 0

If InStr(LCase(rng.Text), "elist") > 0 Then

For Each myPar In myDoc.Paragraphs

myWord = Trim(myPar.Range.Words(1).Text)

If Len(myWord) > 2 Then

allExcepts = allExcepts & myWord & CR

Debug.Print allExcepts

End If

DoEvents

Next myPar

End If

DoEvents

Next myDoc

Debug.Print allExcepts

' Create a text-only copy in another file

Set rngOld = FUT.Content

Documents.Add

Set erList = ActiveDocument

Set rng = ActiveDocument.Content

rng.LanguageID = thisLanguage

rng.FormattedText = rngOld.FormattedText

numNotes = FUT.Endnotes.Count

If numNotes > 0 Then

rng.Collapse wdCollapseEnd

rng.FormattedText = FUT.StoryRanges(wdEndnotesStory).FormattedText

End If

numNotes = FUT.Footnotes.Count

If numNotes > 0 Then

rng.Collapse wdCollapseEnd

rng.FormattedText = FUT.StoryRanges(wdFootnotesStory).FormattedText

End If

' copy all the textboxes to the end of the text

shCount = FUT.Shapes.Count

If shCount > 0 Then

Selection.EndKey Unit:=wdStory

For j = 1 To shCount

Set shp = FUT.Shapes(j)

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

Set rng = shp.TextFrame.TextRange

If Len(rng.Text) > 1 Then

Selection.Text = rng.Text

Selection.EndKey Unit:=wdStory

End If

End If

End If

DoEvents

Next

End If

' Add a newline for safety

Selection.TypeText CR

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^2"

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = "([!a-zA-Z])['" & ChrW(8216) & "]"

.Replacement.Text = "\1"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Change ligature characters into character pairs

myFind = myFind & "," & ChrW(-1280) & "," & ChrW(-1279) & \_

"," & ChrW(-1278) & "," & ChrW(-1277) & "," & ChrW(-1276) \_

& "," & ChrW(185) & "," & ChrW(178) & "," & ChrW(179)

myReplace = myReplace & ",ff,fi,fl,ffi,ffl, , , , , "

fnd = Split(myFind, ",")

rpl = Split(myReplace, ",")

For i = 0 To UBound(fnd)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Preparing file for spellchecking" & CR2

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

DoEvents

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

If ignoreNumbers = True Then

.Text = "[!a-zA-Z^13'" & ChrW(8217) \_

& ChrW(248) & "-" & ChrW(591) \_

& ChrW(697) & "-" & ChrW(703) \_

& ChrW(&H591) & "-" & ChrW(&H5FF) \_

& ChrW(7680) & "-" & ChrW(7935) & "]"

Else

.Text = "[!a-zA-Z0-9^13'" & ChrW(8217) \_

& ChrW(248) & "-" & ChrW(591) \_

& ChrW(697) & "-" & ChrW(703) \_

& ChrW(&H591) & "-" & ChrW(&H5FF) \_

& ChrW(7680) & "-" & ChrW(7935) & "]"

End If

.Replacement.Text = sp1

.Wrap = wdFindContinue

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Text = "([a-zA-Z])['" & ChrW(8217) & "][!a-zA-Z]"

.Replacement.Text = "\1 "

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^p"

.Replacement.Text = sp1

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^t"

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^p"

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Create spelling error list

Set rng = ActiveDocument.Content

txt = rng.Text

' Ensure a single space between words

Do

nowLen = Len(txt)

txt = Replace(txt, sp2, sp1)

DoEvents

Loop Until Len(txt) = nowLen

erList1 = ""

erListFreq1 = ""

erList2 = ""

erListFreq2 = ""

If Left(txt, 1) <> sp1 Then txt = sp1 & txt

Do

spPos = InStr(Mid(txt, 2), " ")

wd = Trim(Left(txt, spPos))

spOK = Application.CheckSpelling(wd, MainDictionary:=langText)

spEX = InStr(allExcepts, CR & wd & CR)

nowLen = Len(txt)

txt = Replace(txt, sp1 & wd & sp1, sp1)

txt = Replace(txt, sp1 & wd & sp1, sp1)

lossLen = Len(txt)

If spOK = False And spEX = False And Len(wd) > 2 Then

' record it as a spelling error

numErrs = Int((nowLen - lossLen) / Len(wd))

numText = " . . . " & Trim(Str(numErrs))

If LCase(wd) = wd Then

erList2 = erList2 & wd & CR

erListFreq2 = erListFreq2 & wd & numText & CR

Else

erList1 = erList1 & wd & CR

erListFreq1 = erListFreq1 & wd & numText & CR

End If

StatusBar = wd & numText

Debug.Print wd & numText

End If

DoEvents

Loop Until Len(txt) < 5

Selection.WholeStory

Selection.Delete

Selection.TypeText Replace(erList2, CR2, CR)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeText CR

listStart = Selection.Start

Selection.TypeText CR & Replace(erList1, CR2, CR)

Selection.Start = listStart

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.WholeStory

Selection.LanguageID = thisLanguage

Selection.Style = wdStyleNormal

Selection.Collapse wdCollapseStart

Selection.TypeText spellingListName & vbCr

ActiveDocument.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading1)

Documents.Add

Selection.TypeText Replace(erListFreq2, CR2, CR)

Selection.WholeStory

DoEvents

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

DoEvents

Selection.EndKey Unit:=wdStory

Selection.TypeText CR

listStart = Selection.Start

Selection.TypeText CR & Replace(erListFreq1, CR2, CR)

Selection.Start = listStart

DoEvents

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

DoEvents

Selection.WholeStory

Selection.LanguageID = thisLanguage

Selection.Style = wdStyleNormal

Selection.HomeKey Unit:=wdStory

Selection.TypeText spellingFreqName & vbCr

ActiveDocument.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading1)

Selection.Collapse wdCollapseStart

erList.Activate

Selection.HomeKey Unit:=wdStory

Application.ScreenUpdating = True

StatusBar = ""

If doingSeveralMacros = False Then

totTime = Int(10 \* (Timer - timeStart) / 60) / 10

If totTime > 2 Then myResponse = MsgBox((totTime & " minutes"), \_

vbOKOnly, "SpellAlyse")

Beep

Else

FUT.Activate

End If

Exit Sub

autoCorrect:

Application.ScreenUpdating = True

allText = FUT.Content.Text

If InStr(allText, "|") > 0 Then GoTo createListPair

myResponse = MsgBox("Auto-create FRedit list items?", \_

vbQuestion + vbYesNoCancel, "SpellAlyse")

If myResponse <> vbYes Then Exit Sub

Beep

myResponse = MsgBox("Turn track changes on?", \_

vbQuestion + vbYesNoCancel, "SpellAlyse")

If myResponse = vbCancel Then Exit Sub

myRev = (myResponse = vbYes)

If myRev Then ActiveDocument.TrackRevisions = True

' Add auto-created FRedit items

Set rng = ActiveDocument.Content

' (only first character, in case of split language)

rng.End = rng.Start + 1

langName = Languages(rng.LanguageID).NameLocal

i = 0

Do

i = i + 1

Set myPara = FUT.Paragraphs(i)

wd = Replace(myPara.Range.Text, CR, "")

If LCase(wd) = wd And wd <> "" Then

DoEvents

spellOK = Application.CheckSpelling(wd, \_

MainDictionary:=langName)

Set suggList = Application.GetSpellingSuggestions(wd, \_

MainDictionary:=langName)

newWord = ""

DoEvents

If suggList.Count > 0 And Not (spellOK) Then

newWord = suggList.Item(1).Name

If myRev = True Then

Set rng = myPara.Range.Duplicate

rng.MoveEnd , -1

myFR = "~<" & wd & ">|" & newWord

rng.Text = myFR

Else

myPara.Range.Text = "~<" & wd & ">|" & newWord & CR

End If

myPara.Range.Select

Selection.Collapse wdCollapseEnd

DoEvents

End If

End If

DoEvents

Loop Until i > 3 And myPara = CR

ActiveDocument.TrackRevisions = False

Exit Sub

createListPair:

' Make Flist + MarkIt list + Elist

Application.ScreenUpdating = True

myResponse = MsgBox("Create exceptions and FRedit lists?", \_

vbQuestion + vbYesNoCancel, "SpellAlyse")

If myResponse <> vbYes Then Exit Sub

Documents.Add

Set eList = ActiveDocument

Selection.TypeText Text:="| Elist" & CR2

eList.Paragraphs(1).Style = eList.Styles(wdStyleHeading1)

Set eRng = ActiveDocument.Content

Documents.Add

Set fList = ActiveDocument

Selection.TypeText Text:="| FRedit" & CR2

fList.Paragraphs(1).Style = fList.Styles(wdStyleHeading1)

Set fRng = ActiveDocument.Content

' FUT is now the spelling error list

For i = 1 To FUT.Paragraphs.Count

DoEvents

Set itemRng = FUT.Paragraphs(i).Range

If InStr(itemRng.Text, spellingListName) = 0 \_

And Len(itemRng.Text) > 3 Then

DoEvents

If InStr(itemRng.Text, "|") > 0 Then

fRng.Collapse wdCollapseEnd

fRng.FormattedText = itemRng.FormattedText

Else

myFontCol = itemRng.Font.Color

myHiCol = itemRng.HighlightColorIndex

If myFontCol > 0 Or myHiCol > 0 Then

fRng.Collapse wdCollapseEnd

fRng.Text = "~<" & Replace(itemRng.Text, CR, \_

"") & ">" & "|^&" & CR

fRng.Expand wdParagraph

fRng.Font.StrikeThrough = True

If myFontCol > 0 Then fRng.Font.Color = myFontCol

If myHiCol > 0 Then fRng.HighlightColorIndex = myHiCol

fRng.Collapse wdCollapseEnd

Else

eRng.Collapse wdCollapseEnd

eRng.Text = itemRng.Text

End If

End If

End If

DoEvents

Next i

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub DeleteAllSpellingErrors()**

' Paul Beverley - Version 08.05.19

' Deletes all spelling errors

minLengthSpell = 2

myColour = wdBrightGreen

If Selection.Start = Selection.End Then

Set rng = ActiveDocument.Content

Else

Set rng = Selection.range.Duplicate

End If

myLanguage = Selection.LanguageID

For Each wd In rng.Words

If Len(wd) >= minLengthSpell Then

If Application.CheckSpelling(wd, \_

MainDictionary:=Languages(myLanguage).NameLocal) = False Then

wd.Font.DoubleStrikeThrough = True

wd.HighlightColorIndex = wdBrightGreen

End If

End If

Next wd

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.DoubleStrikeThrough = True

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub FReditCopy()**

' Paul Beverley - Version 20.01.16

' Copy word to make FRedit script item

' Shift-Alt-f6

Selection.EndKey Unit:=wdLine

Selection.MoveStartWhile cset:=ChrW(124) & Chr(9) & " .", Count:=wdBackward

If Selection <> Chr(13) Then Selection.Delete

Selection.Expand wdParagraph

Selection.MoveEnd , -1

Selection.Copy

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=ChrW(124)

Selection.Paste

End Sub

**Sub FReditCopyPlus()**

' Paul Beverley - Version 18.01.18

' Copy word to make FRedit script item + highlight + case insensitive

' Highlighting?

myHighlight = wdGray25

myHighlight = 0

' Text colouring?

myColour = 0

myColour = wdColorBlue

' Case?

anyCase = True

anyCase = False

' Tracked?

tracked = True

tracked = False

Selection.Expand wdParagraph

If ActiveDocument.range.End = Selection.End Then

Selection.Collapse wdCollapseEnd

Selection.TypeParagraph

Selection.MoveLeft , 1

Selection.Expand wdParagraph

End If

If Len(Selection) > 3 Then

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.TypeParagraph

Else

Selection.Collapse wdCollapseStart

Selection.TypeParagraph

Selection.MoveLeft , 1

End If

' Selection.PasteAndFormat (wdFormatPlainText)

Selection.PasteSpecial DataType:=wdPasteText

Set rng = ActiveDocument.Content

rng.Start = Selection.Start - 1

rng.End = Selection.Start

If rng = " " Then rng.Delete

Selection.Expand wdParagraph

tabPos = InStr(Selection, Chr(9))

If tabPos > 0 Then

Selection.MoveStart wdCharacter, tabPos - 1

Selection.MoveEnd wdCharacter, -1

Selection.Delete

Selection.Expand wdParagraph

End If

Selection.MoveEnd wdCharacter, -1

dotPos = InStr(Selection, " . .")

lineStart = Selection.Start

If dotPos > 0 Then

Selection.MoveStart , dotPos - 1

Selection.Delete

Selection.MoveStart , -1

If Selection = "." Then Selection.Delete

Selection.Start = lineStart

End If

Selection.Copy

Selection.EndKey Unit:=wdLine

Selection.TypeText Text:=ChrW(124)

Selection.Paste

Selection.Expand wdParagraph

Selection.range.HighlightColorIndex = myHighlight

Selection.Font.Color = myColour

Selection.range.Font.StrikeThrough = Not (tracked)

Selection.HomeKey Unit:=wdLine

If anyCase = True Then Selection.TypeText Text:=ChrW(172)

Selection.EndKey Unit:=wdLine

End Sub

**Sub FReditCopyWholeWord()**

**'** Paul Beverley - Version 17.01.12

' Create FRedit item for whole-word F&R

Selection.Paragraphs(1).Range.Select

tabPos = InStr(Selection, Chr(9))

If tabPos > 0 Then

Selection.MoveStart wdCharacter, tabPos - 1

Selection.MoveEnd , -1

Selection.Delete

Selection.Paragraphs(1).Range.Select

End If

Selection.MoveEnd , -1

Selection.Copy

Selection.Collapse wdCollapseStart

Selection.TypeText Text:="~<"

Selection.EndKey Unit:=wdLine

Selection.TypeText Text:=">" & ChrW(124)

Selection.Paste

Selection.MoveRight Unit:=wdCharacter, Count:=1

End Sub

**Sub FReditSame()**

' Paul Beverley - Version 21.08.19

' Create FRedit item with ^&

addColorAndStrike = True

myColour = wdColorPink

Selection.Expand wdParagraph

If Len(Selection) < 2 Then

Selection.MoveRight Count:=1

Exit Sub

End If

Selection.MoveEnd wdCharacter, -1

tabPos = InStr(Selection, Chr(9))

If tabPos = 0 Then tabPos = InStr(Selection, " .")

If tabPos > 0 Then

Selection.MoveStart wdCharacter, tabPos - 1

Selection.Delete

Else

Selection.Collapse wdCollapseEnd

End If

Selection.TypeText Text:=ChrW(124) & "^&"

If addColorAndStrike = True Then

Selection.Expand wdParagraph

Selection.Font.StrikeThrough = True

Selection.range.Font.Color = myColour

Selection.Collapse wdCollapseEnd

End If

Selection.MoveRight , 1

End Sub

**Sub FReditSwap()**

' Paul Beverley - Version 20.01.16

' Swaps the two sides of a FRedit item

Selection.Expand wdParagraph

Selection.MoveEnd , -1

allText = Selection

padPos = InStr(allText, ChrW(124))

allText = Mid(allText, padPos + 1) & ChrW(124) & Left(allText, padPos - 1)

Selection.TypeText allText

Selection.MoveRight wdWord, 1

End Sub

**Sub FReditListProcess()**

' Paul Beverley - Version 08.02.22

' Tidies up a spelling FRedit list from cursor downwards

makeCopyingNotTracked = True

' lightHighlight = wdGray25

lightHighlight = wdNoHighlight

' lightColour = wdColorBlack

lightColour = wdColorDarkBlue

strongHighlight = wdBrightGreen

changeColour = wdColorDarkBlue

minLength = 7

myResponse = MsgBox("Any case? (Yes)" & vbCr & vbCr & "or Case Sensitive? (No)", \_

vbQuestion + vbYesNoCancel, "FRedit List Process")

If myResponse = vbCancel Then Exit Sub

doAnyCase = (myResponse = vbYes)

Selection.Expand wdParagraph

Do While Len(Selection) > 1

Selection.Expand wdParagraph

Selection.End = Selection.Start + 1

nowHighlight = Selection.Range.HighlightColorIndex

nowColour = Selection.Range.Font.Color

Selection.Expand wdParagraph

Selection.MoveEnd , -1

padPos = InStr(Selection, ChrW(124))

If padPos = 0 Then

' Check if text colour is not black

doCopy = (nowColour <> wdBlack And nowColour <> wdColorAutomatic)

' Check if it is highlighted

If Selection.Range.HighlightColorIndex > 0 Then doCopy = True

' Check if italic, bold or underline

If Selection.Font.Italic Then doCopy = True

If Selection.Font.Bold Then doCopy = True

If Selection.Font.Underline Then doCopy = True

' If it has one of these then FRedit must copy it

If doCopy = True Then

Selection.Collapse wdCollapseStart

If doAnyCase = True Then

Selection.TypeText Text:=ChrW(172)

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.TypeText Text:=ChrW(124) & "^&"

Else

Selection.TypeText Text:="~<"

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.TypeText Text:=">" & ChrW(124) & "^&"

End If

If makeCopyingNotTracked = True Then

Selection.Expand wdParagraph

Selection.Font.StrikeThrough = True

End If

End If

Else

oldWord = Left(Selection, padPos - 1)

newWord = Mid(Selection, padPos + 1)

If Len(oldWord) > minLength Then

' The word is long enough not to bother with whole word only

' so leave it as it is, but just add bent pipe if case insensitive

If doAnyCase = True Then

Selection.Collapse wdCollapseStart

Selection.TypeText Text:=ChrW(172)

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

End If

Else

If makeCopyingNotTracked = True Then

' First line is: errorword|^&

If doAnyCase = True Then Selection.TypeText Text:=ChrW(172)

Selection.TypeText Text:=oldWord & ChrW(124) & "^&"

Selection.Expand wdParagraph

Selection.Font.StrikeThrough = True

If oldWord <> newWord Then

Selection.Range.HighlightColorIndex = lightHighlight

Selection.Range.Font.Color = lightColour

' Next line is: ~<errorword>|correctword

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:="~<" & oldWord & ">" & \_

ChrW(124) & newWord & vbCr

Selection.MoveLeft , 2

Selection.Expand wdParagraph

Selection.Range.HighlightColorIndex = nowHighlight

Selection.Range.Font.Color = nowColour

Else

Selection.Range.HighlightColorIndex = nowHighlight

End If

Selection.Collapse wdCollapseStart

Else

' We're not tracking

' First line is: errorword|^&

If doAnyCase = True Then Selection.TypeText Text:=ChrW(172)

Selection.TypeText Text:=oldWord & ChrW(124) & "^&"

Selection.Expand wdParagraph

Selection.Range.HighlightColorIndex = lightHighlight

' Next line is: ~<errorword>|correctword

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:="~<" & oldWord & ">" & ChrW(124) \_

& newWord & vbCr

Selection.MoveLeft , 2

Selection.Expand wdParagraph

Selection.Range.HighlightColorIndex = strongHighlight

Selection.Range.Font.Color = changeColour

' Selection.range.Font.Color = nowColour

End If

DoEvents

End If

DoEvents

End If

' Now check if it has still got no vertical bar

Selection.Expand wdParagraph

If InStr(Selection, ChrW(124)) = 0 Then

Selection.Delete

Selection.MoveLeft , 1

Else

DoEvents

End If

Selection.MoveDown , 1

Selection.Expand wdParagraph

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , -1

Selection.MoveLeft , 1

Beep

End Sub

**Sub SpellingListProcess()**

' Paul Beverley - Version 27.04.16

' Tidy up a spelling FRedit list from cursor downwards

doTrack = True

lightHighlight = wdGray25

strongHighlight = wdBrightGreen

changeColour = wdColorDarkBlue

minLen = 7

Selection.Expand wdParagraph

Do While Len(Selection) > 1

Selection.Expand wdParagraph

Selection.End = Selection.Start + 1

nowHighlight = Selection.range.HighlightColorIndex

nowColour = Selection.range.Font.Color

Selection.Expand wdParagraph

Selection.MoveEnd , -1

padPos = InStr(Selection, ChrW(124))

If padPos = 0 Then

' Check if text colour is not black

doCopy = (nowColour <> wdBlack And nowColour <> wdColorAutomatic)

' Check if it is highlighted

If Selection.range.HighlightColorIndex > 0 Then doCopy = True

' Check if italic, bold or underline

If Selection.Font.Italic Then doCopy = True

If Selection.Font.Bold Then doCopy = True

If Selection.Font.Underline Then doCopy = True

' If it has one of these then FRedit must copy it

If doCopy = True Then

Selection.Collapse wdCollapseStart

Selection.TypeText Text:="~<"

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.TypeText Text:=">" & ChrW(124) & "^&"

If doTrack = True Then

Selection.Expand wdParagraph

Selection.Font.StrikeThrough = True

End If

End If

Else

oldWord = Left(Selection, padPos - 1)

newWord = Mid(Selection, padPos + 1)

If Len(oldWord) > minLen Then

' The word is long enough not to bother with whole word only

' so leave it as it is, right?

Else

If doTrack = True Then

' First line is: errorword|^&

Selection.TypeText Text:=oldWord & ChrW(124) & "^&"

Selection.Expand wdParagraph

Selection.Font.StrikeThrough = True

If oldWord <> newWord Then

Selection.range.HighlightColorIndex = lightHighlight

' Next line is: ~<errorword>|correctword

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:="~<" & oldWord & ">" & \_

ChrW(124) & newWord & vbCr

Selection.MoveLeft , 2

Selection.Expand wdParagraph

Selection.range.HighlightColorIndex = nowHighlight

Selection.range.Font.Color = nowColour

Else

Selection.range.HighlightColorIndex = nowHighlight

End If

Selection.Collapse wdCollapseStart

Else

' We're not tracking

' First line is: errorword|^&

Selection.TypeText Text:=oldWord & ChrW(124) & "^&"

Selection.Expand wdParagraph

Selection.range.HighlightColorIndex = lightHighlight

' Next line is: ~<errorword>|correctword

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:="~<" & oldWord & ">" & ChrW(124) \_

& newWord & vbCr

Selection.MoveLeft , 2

Selection.Expand wdParagraph

Selection.range.HighlightColorIndex = strongHighlight

Selection.range.Font.Color = changeColour

' Selection.range.Font.Color = nowColour

End If

End If

End If

' Now check if it has still got no vertical bar

Selection.Expand wdParagraph

If InStr(Selection, ChrW(124)) = 0 Then

Selection.Delete

Selection.MoveLeft , 1

End If

Selection.MoveDown , 1

Selection.Expand wdParagraph

Loop

Selection.Collapse wdCollapseEnd

End Sub

**Sub HyphenationToFRedit()**

' Paul Beverley - Version 04.02.22

' Creates FRedit items from HyphenAlyse or PropernounAlyse files

makeListCaseInsensitive = True

useProperNounToFRedit = True

myList = "anti,cross,eigen,hyper,inter,meta,mid,multi," \_

& "non,over,post,pre,pseudo,quasi,semi,sub,super"

CR = vbCr

inAtable = Selection.Information(wdWithInTable)

If Not (inAtable) Then GoTo ProperNouns

wantOthers = (Selection.Start = Selection.End)

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

' rng.Select

startCell = Replace(rng.Text, Chr(7), "")

startCell = Replace(startCell, Chr(13), "")

isBlank = (Len(startCell) < 3)

colNum = rng.Cells(1).ColumnIndex

rng.Expand wdRow

myCell = Split(Chr(7) & rng.Text, Chr(7))

For i = 1 To 4

myText = Replace(myCell(i), vbCr, "")

' myText = myCell(i)

sdfgs = Len(myText)

leadPos = InStr(myText, " . ")

If leadPos > 0 Then

myText = Left(myText, leadPos - 1)

End If

myCell(i) = myText

Next i

preWord = ""

If Len(myCell(1)) > 2 Then

myText = myCell(1)

spPos = InStr(myText, "-")

preWord = Left(myText, spPos - 1)

postWord = Mid(myText, spPos + 1)

End If

myText = myCell(2)

If preWord = "" And myText > "" Then

spPos = InStr(myText, " ")

preWord = Left(myText, spPos - 1)

postWord = Mid(myText, spPos + 1)

End If

myText = myCell(4)

If preWord = "" And myText > "" Then

spPos = InStr(myText, ChrW(8211))

preWord = Left(myText, spPos - 1)

postWord = Mid(myText, spPos + 1)

End If

If preWord = "" Then

myList = "," & myList & ","

myText = myCell(3)

For i = 3 To 6

preWord = Left(myText, i)

If InStr(myList, "," & preWord & ",") > 0 Then Exit For

Next i

If i > 6 Then

Beep

MsgBox ("Can't find this prefix")

Exit Sub

End If

postWord = Replace(myText, preWord, "")

End If

Select Case colNum

Case 1: myJoin = "-"

Case 2: myJoin = " "

Case 3: myJoin = ""

Case 4: myJoin = ChrW(8211)

Case Else

Beep

MsgBox ("Something has gone wrong.")

Exit Sub

End Select

targetWord = preWord & myJoin & postWord

oneWord = preWord & postWord

hyphWord = preWord & "-" & postWord

twoWords = preWord & " " & postWord

dashWords = preWord & ChrW(8211) & postWord

myFRitem = ""

If makeListCaseInsensitive = True Then

cs = ChrW(172)

Else

cs = ""

End If

If wantOthers Then

If targetWord = oneWord Then

myFRitem = cs & twoWords & ChrW(124) & oneWord & CR & \_

cs & hyphWord & ChrW(124) & oneWord & CR & \_

cs & dashWords & ChrW(124) & oneWord & CR

End If

If targetWord = twoWords Then

myFRitem = cs & oneWord & ChrW(124) & twoWords & CR & \_

cs & hyphWord & ChrW(124) & twoWords & CR & \_

cs & dashWords & ChrW(124) & twoWords & CR

End If

If targetWord = hyphWord Then

myFRitem = cs & twoWords & ChrW(124) & hyphWord & CR & \_

cs & oneWord & ChrW(124) & hyphWord & CR & \_

cs & dashWords & ChrW(124) & hyphWord & CR

End If

If targetWord = dashWords Then

myFRitem = cs & twoWords & ChrW(124) & dashWords & CR & \_

cs & oneWord & ChrW(124) & dashWords & CR & \_

cs & hyphWord & ChrW(124) & dashWords & CR

End If

Else

If targetWord = oneWord Then

If myCell(1) > "" Then

myFRitem = myFRitem & cs & hyphWord & ChrW(124) & oneWord & CR

End If

If myCell(2) > "" Then

myFRitem = myFRitem & cs & twoWords & ChrW(124) & oneWord & CR

End If

If myCell(4) > "" Then

myFRitem = myFRitem & cs & dashWords & ChrW(124) & oneWord & CR

End If

If InStr(myFRitem, ChrW(124)) = 0 Then

myFRitem = cs & hyphWord & ChrW(124) & oneWord & CR

End If

End If

If targetWord = twoWords Then

If myCell(1) > "" Then

myFRitem = cs & hyphWord & ChrW(124) & twoWords & CR

End If

If myCell(3) > "" Then

myFRitem = myFRitem & cs & oneWord & ChrW(124) & twoWords & CR

End If

If myCell(4) > "" Then

myFRitem = myFRitem & cs & dashWords & ChrW(124) & twoWords & CR

End If

If InStr(myFRitem, ChrW(124)) = 0 Then

myFRitem = cs & hyphWord & ChrW(124) & twoWords & CR

End If

End If

If targetWord = hyphWord Then

If myCell(2) > "" Then

myFRitem = cs & twoWords & ChrW(124) & hyphWord & CR

End If

If myCell(3) > "" Then

myFRitem = cs & oneWord & ChrW(124) & hyphWord & CR

End If

If myCell(4) > "" Then

myFRitem = cs & dashWords & ChrW(124) & hyphWord & CR

End If

If InStr(myFRitem, ChrW(124)) = 0 Then

myFRitem = cs & oneWord & ChrW(124) & hyphWord & CR

End If

End If

End If

If myFRitem = "" Then

Beep

MsgBox ("For prefixes, place the cursor in a blank cell.")

Else

gottaFReditList = False

Set thisDoc = ActiveDocument

For Each myWnd In Application.Windows

myWnd.Document.Activate

Set rng = ActiveDocument.Content

' Finding a '|' in the first 250 characters means it's a list

rng.End = rng.Start + 250

If InStr(rng.Text, "|") > 0 Then

gottaFReditList = True

Exit For

End If

Next myWnd

If gottaFReditList = False Then

thisDoc.Activate

MsgBox "Can't find a FRedit list file"

Exit Sub

End If

Selection.Expand wdParagraph

If Len(Selection) > 1 Then Selection.Collapse wdCollapseEnd

Selection.TypeText myFRitem

End If

thisDoc.Activate

Exit Sub

ProperNouns:

If useProperNounToFRedit = True Then

Call ProperNounToFRedit

Else

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

myWord = Trim(Selection)

myFRitem = myWord & "|" & myWord & vbCr

gottaFReditList = False

Set thisDoc = ActiveDocument

For Each myWnd In Application.Windows

myWnd.Document.Activate

Set rng = ActiveDocument.Content

' Finding a '|' in the first 250 characters means it's a list

rng.End = rng.Start + 250

If InStr(rng.Text, "|") > 0 Then

gottaFReditList = True

Exit For

End If

Next myWnd

If gottaFReditList = False Then

thisDoc.Activate

MsgBox "Can't find a FRedit list file"

Exit Sub

End If

Selection.Expand wdParagraph

If Len(Selection) > 1 Then Selection.Collapse wdCollapseEnd

Selection.TypeText myFRitem

thisDoc.Activate

End If

End Sub

**Sub ProperNounToFRedit()**

' Paul Beverley - Version 04.02.22

' Picks up alternative spellings in a PN query list for a FRedit list

Set thisDoc = ActiveDocument

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

myRepl = Trim(rng.Text)

rng.Expand wdParagraph

rng.Collapse wdCollapseStart

rng.End = rng.End - 1

rng.Expand wdParagraph

sdfsd = rng.Text

myFandRs = ""

Do While Len(rng.Text) > 2

myFind = Trim(rng.Words(3))

If myFind = "=" Then myFind = Trim(rng.Words(5))

myFandRs = myFandRs & myFind & "|" & myRepl & vbCr

rng.Collapse wdCollapseStart

rng.End = rng.End - 1

rng.Expand wdParagraph

Loop

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

Do While Len(rng.Text) > 2

myFind = Trim(rng.Words(3))

If myFind = "=" Then myFind = Trim(rng.Words(5))

myFandRs = myFandRs & myFind & "|" & myRepl & vbCr

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

Loop

gottaFReditList = False

Set thisDoc = ActiveDocument

For Each myWnd In Application.Windows

myWnd.Document.Activate

Set rng = ActiveDocument.Content

rng.End = rng.Start + 250

If InStr(rng.Text, "|") > 0 Then

gottaFReditList = True

Exit For

End If

Next myWnd

If gottaFReditList = False Then

thisDoc.Activate

MsgBox "Can't find a FRedit list file"

Exit Sub

End If

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=myFandRs

thisDoc.Activate

End Sub

**Sub FReditListCreate()**

' Paul Beverley - Version 20.09.18

' Adds text to list items and applies formatting

txtBefore = "~<"

txtafter = ">|^&"

' e.g. for the word followed by a comma, colon or apostrophe,

' you could use this one:

' txtafter = "[,':]|^&"

doItalic = False

doBold = False

addColour = wdColorBlack

' addColour = wdColorBlue

' addHighlight = wdNoHighlight

addHighlight = wdYellow

txtBefore = Replace(txtBefore, "^", "^^")

txtafter = Replace(txtafter, "^", "^^")

Set rng = ActiveDocument.Content

rng.InsertAfter Text:=vbCr

rng.InsertBefore Text:=vbCr

curlyNOw = Options.AutoFormatAsYouTypeReplaceQuotes

Options.AutoFormatAsYouTypeReplaceQuotes = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.MatchWildcards = False

.Text = "^p"

.Replacement.Text = "zxzx^pabab"

.Execute Replace:=wdReplaceAll

.Text = "ababzxzx"

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

.Text = "zxzx"

.Replacement.Text = txtafter

.Execute Replace:=wdReplaceAll

.Text = "abab"

.Replacement.Text = txtBefore

.Execute Replace:=wdReplaceAll

End With

Options.AutoFormatAsYouTypeReplaceQuotes = curlyNOw

' Add features

Set rng = ActiveDocument.Content

If doItalic = True Then rng.Font.Italic = True

If doBold = True Then rng.Font.Bold = True

If addColour <> 0 Then rng.Font.Color = addColour

rng.HighlightColorIndex = addHighlight

ActiveDocument.Paragraphs(1).range.Delete

numParas = ActiveDocument.Paragraphs.Count

ActiveDocument.Paragraphs(numParas).range.Delete

End Sub

**Sub EmailFormatter()**

' Paul Beverley - Version 05.11.13

' Formats paragraphs in an email

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^11"

.Wrap = wdFindContinue

.Replacement.Text = " "

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.Text = "^p^p"

.Replacement.Text = "zczc"

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.Text = "^p"

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.Text = "zczc"

.Replacement.Text = "^p^p"

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub PDFsoftHyphenRemove()**

' Paul Beverley - Version 04.01.14

' Unhyphenate split words

' Highlight the result (use zero for no highlight)

myColour = wdGray25

Selection.HomeKey Unit:=wdStory

langText = Languages(Selection.LanguageID).NameLocal

' Go and find the first occurrence

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-zA-Z]-^13"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

' If you want to count them...

splitPoint = rng.Start + 1

rng.End = rng.Start + 1

rng.Start = rng.Start - 60

wordOne = rng.Words.Last

myStart = rng.End - Len(wordOne)

rng.Start = splitPoint + 2

rng.End = rng.Start + 25

wordTwo = Trim(rng.Words.First)

myEnd = rng.Start + Len(wordTwo)

gotOne = False

' If it's a spelling error...

oneWord = wordOne & wordTwo

If InStr(oneWord, "\_") = 0 Then

If Application.CheckSpelling(oneWord, \_

MainDictionary:=langText) = True Then

rng.Start = splitPoint

rng.End = splitPoint + 2

rng.Delete

rng.Select

gotOne = True

rng.Start = myEnd - 2

rng.End = myEnd - 1

rng.Select

rng.Text = vbCr

End If

End If

myCount = myCount + 1

If myColour > 0 And gotOne = True Then

rng.Start = myStart

rng.End = myEnd - 2

rng.HighlightColorIndex = myColour

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

rng.Select

MsgBox "Changed: " & myCount

End Sub

**Sub PDFhardHyphenRestore()**

' Paul Beverley - Version 18.04.13

' Hyphenate falsely concatenated words

' Highlight the result?

hyphColour = wdNoHighlight

' or

hyphColour = wdGray25

minWordLen = 5

' For diagnostic purposes use yellow

singleColour = wdYellow

singleColour = wdNoHighlight

Set rng = ActiveDocument.Content

langText = Languages(Selection.LanguageID).NameLocal

Selection.HomeKey Unit:=wdStory

timeNow = Timer

For Each myPara In ActiveDocument.Paragraphs

Set rng = myPara.Range

rng.End = rng.End - 1

myWord = rng.Words.Last

myLen = Len(myWord)

' If it's a spelling error...

If myLen > minWordLen Then

If Application.CheckSpelling(myWord, MainDictionary:=langText) \_

= False Then

For i = 2 To myLen - 2

wordOne = Left(myWord, myLen - i)

If Application.CheckSpelling(wordOne, \_

MainDictionary:=langText) = True Then

wordTwo = Right(myWord, i)

If Application.CheckSpelling(wordTwo, \_

MainDictionary:=langText) = True Then

myPara.Range.Select

Selection.MoveEnd , -1

Selection.Start = Selection.End - myLen

Selection.InsertBefore Text:="zczc"

Selection.Start = Selection.End - i

Selection.InsertBefore Text:="pqpq"

Exit For

End If

End If

Next i

End If

End If

Next myPara

' Now check all zczc

myCount = 0

oldColour = Options.DefaultHighlightColorIndex

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc\*^13"

.Replacement.Text = ""

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

myCount = 0

Selection.HomeKey Unit:=wdStory

While rng.Find.Found

rng.End = rng.End - 1

wholeBit = rng

pqpqWord = Replace(wholeBit, "zczc", "")

hyphWord = Replace(pqpqWord, "pqpq", "-")

singleWord = Replace(pqpqWord, "pqpq", "")

Set rng = ActiveDocument.Content

With rng.Find

.Text = hyphWord

.Execute

End With

' Does the hyphenated Paul Beverley - Version occur anywhere else in the text?

If rng.Find.Found Then

' If so, change it and highlight it

Set rng = ActiveDocument.Content

Options.DefaultHighlightColorIndex = hyphColour

myCount = myCount + 1

With rng.Find

.Text = wholeBit

.Replacement.Text = hyphWord

.MatchCase = False

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

StatusBar = "Hyphenating : " & hyphWord

Else

' If not restore the original word

Set rng = ActiveDocument.Content

Options.DefaultHighlightColorIndex = singleColour

With rng.Find

.Text = wholeBit

.Replacement.Text = singleWord

.MatchCase = False

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

StatusBar = "Not hyphenating : " & singleWord

End If

Set rng = ActiveDocument.Content

With rng.Find

.Text = "zczc\*^13"

.MatchCase = False

.MatchWildcards = True

.Execute

End With

Wend

Options.DefaultHighlightColorIndex = oldColour

MsgBox "Changed: " & myCount & " different words."

End Sub

**Sub PDFHyphenChecker()**

' Paul Beverley - Version 25.10.19

' Checks the line-end hyphenation of a converted PDF

minLength = 4

Set rng = ActiveDocument.range(0, 1)

myLanguage = rng.LanguageID

CR = vbCr: CR2 = CR & CR

Set rng = ActiveDocument.Content

OKposn = InStr(rng.Text, "OKwords")

If OKposn = 0 Then

rng.InsertAfter Text:=CR2 & "OKwords" & CR2

Else

OKwords = Mid(rng.Text, OKposn)

End If

stopNow = False

Set rng = ActiveDocument.range(0, Selection.End)

paraNum = rng.Paragraphs.Count

For pn = paraNum To ActiveDocument.Paragraphs.Count

Set rng = ActiveDocument.Paragraphs(pn).range

numWds = rng.Words.Count

If numWds > 2 And UCase(rng) <> LCase(rng) Then

w = numWds

Do

w = w - 1

wd = Trim(rng.Words(w))

Loop Until LCase(wd) <> UCase(wd)

DoEvents

Set rngOK = ActiveDocument.Content

OKstart = InStr(rngOK.Text, "OKwords")

If OKstart = 0 Then

MsgBox "OKwordslist corrupted or missing?"

Selection.EndKey Unit:=wdStory

Exit Sub

Else

rngOK.Start = OKstart + 6

spOK = (InStr(rngOK.Text, CR & wd & CR) > 0)

End If

If Not (spOK Or Application.CheckSpelling(wd, \_

MainDictionary:=Languages(myLanguage).NameLocal) \_

Or Len(wd) < minLength - 1) Then

For i = 2 To Len(wd) - 2

wd1 = Left(wd, i)

wd2 = Mid(wd, i + 1)

ok1 = Application.CheckSpelling(wd1, \_

MainDictionary:=Languages(myLanguage).NameLocal)

ok2 = Application.CheckSpelling(wd2, \_

MainDictionary:=Languages(myLanguage).NameLocal)

gotPair = ok1 And ok2

If gotPair Then Exit For

Next i

rng.Words(w).Select

If gotPair Then

newWord = wd1 & "-" & wd2

Selection.Text = newWord

Selection.Collapse wdCollapseStart

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.3

Beep

myResponse = MsgBox("Is it OK to change: " & newWord, vbQuestion \_

+ vbYesNoCancel, "PDFHyphenChecker")

If myResponse <> vbYes Then

WordBasic.EditUndo

If myResponse = vbCancel Then

ActiveDocument.Paragraphs(pn + 1).range.Select

Selection.Collapse wdCollapseEnd

Exit Sub

Else

rngOK.InsertAfter Text:=wd & CR

End If

End If

Else

Selection.Collapse wdCollapseEnd

myResponse = MsgBox("Is " & wd & " an OK word?", vbQuestion \_

+ vbYesNoCancel, "PDFHyphenChecker")

If myResponse = vbCancel Then

Exit Sub

Else

Debug.Print rngOK.Start, rngOK.End

rngOK.InsertAfter Text:=wd & CR

End If

End If

End If

End If

Next pn

End Sub

**Sub PDFHyphenRemover()**

' Paul Beverley - Version 28.10.19

' Finds all end-of-line hyphens, joining back to the next word

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "-^p"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

rng.Delete

rng.End = rng.End + 20

spPos = InStr(rng, " ")

Set rng2 = ActiveDocument.range(rng.Start + spPos - 1, rng.Start + spPos)

If spPos > 0 Then

rng2.Text = vbCr

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Beep

End Sub

**Sub PDFunderlineToLigature()**

' Paul Beverley - Version 04.01.14

' Restore underline characters into ligatures

' Colour if a suitable word found

okColour = wdGray25

' Colour if no suitable word found

nogoColour = wdTurquoise

' replace underlines with...

ch = ChrW(126): ' A tilde ~

myJump = 100

Selection.HomeKey Unit:=wdStory

langText = Languages(Selection.LanguageID).NameLocal

' first get rid of odd underlines

StatusBar = "Removing obvious non-words"

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\_\_"

.Wrap = wdFindContinue

.Replacement.Text = ch & ch

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

StatusBar = "Removing obvious non-words"

With rng.Find

.Text = "\_([A-Z])"

.Replacement.Text = "\1" & ChrW(8211) & "\1"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

StatusBar = "Removing obvious non-words"

With rng.Find

.Text = "([!a-zA-Z])\_([!a-zA-Z])"

.Replacement.Text = "\1" & ChrW(8211) & "\2"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

StatusBar = "Removing obvious non-words"

With rng.Find

.Text = "([!a-zA-Z])\_([!bngrtx][!a-zA-Z])"

.Replacement.Text = "\1" & ch & "\2"

.Execute Replace:=wdReplaceAll

End With

StatusBar = "Removing obvious non-words"

With rng.Find

.Text = "([!a-zA-Z])\_([!a-zA-Z])"

.Replacement.Text = "\1" & ch & "\2"

.Execute Replace:=wdReplaceAll

End With

StatusBar = "Removing obvious non-words"

With rng.Find

.Text = "<([a-zA-Z])\_([!a-zA-Z])"

.Replacement.Text = "\1" & ch & "\2"

.Execute Replace:=wdReplaceAll

End With

' Go and find the first underline character

Set rng = ActiveDocument.Content

With rng.Find

.Text = "\_"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

StatusBar = ""

okChars = "abcdefghijklmnopqrstuvwxyz\_ABCDEFGHIJKLMNOPQRSTUVWXYZ"

myCount = 0

Do While rng.Find.Found = True

' find the whole word

rng.MoveStartWhile cset:=okChars, Count:=wdBackward

rng.MoveEndWhile cset:=okChars, Count:=wdForward

foundWord = rng

newWord = ""

' Try each of the four ligatures in turn (most likely first)

For i = 1 To 4

Select Case i

Case 1: lig = "fi"

Case 2: lig = "ff"

Case 3: lig = "fl"

Case 4: lig = "ffi"

End Select

tryWord = Replace(foundWord, "\_", lig)

isOK = Application.CheckSpelling(tryWord, MainDictionary:=langText)

If isOK = True Then

newWord = tryWord

Exit For

End If

Next i

If newWord > "" Then

' If we've found a new word, replace it

rng.Text = newWord

rng.HighlightColorIndex = okColour

Else

' If not, highlight them all

rng.Text = Replace(foundWord, "\_", ch)

rng.HighlightColorIndex = nogoColour

End If

myCount = myCount + 1

myTest = myCount / myJump

If Int(myTest) = myTest Then rng.Select

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub LigatureConverter()**

' Paul Beverley - Version 09.06.17

' Replaces funny codes for fi/ff/fl/ffi in converted PDF

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Chr(12)

.Wrap = wdFindContinue

.Replacement.Text = "fi"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Chr(14)

.Wrap = wdFindContinue

.Replacement.Text = "ffi"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Chr(11)

.Wrap = wdFindContinue

.Replacement.Text = "ff"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

' Each fl has been replaced by chr(13), which

' is an ordinary newline. Bother!

' So check all line ends, and see if

' any of them for a valid word.

Selection.HomeKey Unit:=wdStory

myLanguage = Selection.range.LanguageID

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Za-z]@^13[a-z]{1,}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

' Note where the end of the found item is

endNow = rng.End

myWord = Replace(rng.Text, Chr(13), "fl")

If Application.CheckSpelling(myWord, MainDictionary:= \_

Languages(myLanguage).NameLocal) = True Then

rng.Select

Selection.TypeText Text:=myWord

End If

rng.Start = endNow

rng.End = endNow

rng.Find.Execute

Loop

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub PDFfunniesToLigatures()**

' Paul Beverley - Version 18.04.13

' Restore ligatures that have been converted to odd characters

fi\_Code = "W"

fl\_Code = "U"

ffi\_Code = "Z"

ff\_Code = "V"

' Colour if a suitable word found

okColour = wdGray25

' Colour if no suitable word found

nogoColour = wdYellow

nogoColour = wdNoHighlight

Selection.HomeKey Unit:=wdStory

langText = Languages(Selection.LanguageID).NameLocal

oldColour = Options.DefaultHighlightColorIndex

okCount = 0

nogoCount = 0

okChars = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"

For i = 1 To 4

Select Case i

Case 1: lig = "fi": myCode = fi\_Code

Case 2: lig = "ff": myCode = ff\_Code

Case 3: lig = "fl": myCode = fl\_Code

Case 4: lig = "ffi": myCode = ffi\_Code

End Select

If myCode > "" Then

' Go and find the first code character

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myCode

.Font.Color = wdColorAutomatic

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchCase = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While rng.Find.Found = True

' find the whole word

rng.MoveStartWhile cset:=okChars, Count:=wdBackward

rng.MoveEndWhile cset:=okChars, Count:=wdForward

foundWord = rng

newWord = ""

' Try each of the four ligatures in turn (most likely first)

tryWord = Replace(foundWord, myCode, lig)

If Application.CheckSpelling(tryWord, MainDictionary:=langText) \_

= True Then newWord = tryWord

If newWord > "" And Len(newWord) > 2 Then

' If we've found a new word, replace them all.

Options.DefaultHighlightColorIndex = okColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "([!a-zA-Z])" & foundWord & "([!a-zA-Z])"

.MatchCase = True

.Wrap = wdFindContinue

.MatchWildcards = True

.Replacement.Text = "\1" & newWord & "\2"

.Replacement.Highlight = True

.Replacement.Font.Color = wdColorBlue

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

okCount = okCount + 1

StatusBar = newWord & " – OK"

Else

' If not, colour them all

Options.DefaultHighlightColorIndex = nogoColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[!a-zA-Z]" & foundWord & "[!a-zA-Z]"

.MatchCase = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Color = wdColorRed

.MatchWildcards = True

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

StatusBar = foundWord & " – Can't find it"

nogoCount = nogoCount + 1

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myCode

.Font.Color = wdColorAutomatic

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchCase = True

.Execute

End With

Loop

End If

Next i

Set rng = ActiveDocument.Content

rng.Font.Color = wdColorAutomatic

Options.DefaultHighlightColorIndex = oldColour

MsgBox "Successfully changed words: " & okCount & vbCrLf & vbCrLf \_

& "Words not to be changed: " & nogoCount

End Sub

**Sub PDFspellAll()**

' Paul Beverley - Version 27.10.22

' Underlines all spelling errors

' Highlight the result (use zero or wdNoHighlight for no highlight)

myColour = wdYellow

' myColour = 0

Selection.HomeKey Unit:=wdStory

langText = Languages(Selection.LanguageID).NameLocal

wasIgnore = Options.IgnoreMixedDigits

Options.IgnoreMixedDigits = False

j = ActiveDocument.Words.Count

StatusBar = "Spellchecking. To go: " & Str(j)

Set rng = ActiveDocument.Content

For Each wd In ActiveDocument.Words

Do While InStr(ChrW(8217) & "'", Right(wd.Text, 1)) > 0

wd.MoveEnd , -1

DoEvents

Loop

If Right(wd, 2) = ChrW(8217) & "s" Then wd.MoveEnd , -2

If Right(wd, 2) = "'s" Then wd.MoveEnd , -2

If wd.Font.StrikeThrough = False Then

If Application.CheckSpelling(wd, MainDictionary:=langText) = False Then

wd.Font.Underline = True

wd.HighlightColorIndex = wdYellow

End If

End If

j = j - 1

If j Mod 100 = 0 Then

StatusBar = "Spellchecking. To go: " & Str(j)

DoEvents

End If

Next wd

StatusBar = ""

Options.IgnoreMixedDigits = wasIgnore

End Sub

**Sub PDFspellIgnoreProperNouns()**

' Paul Beverley - Version 17.11.22

' Underlines all spelling errors except proper nouns

' Highlight the result (use zero or wdNoHighlight for no highlight)

myColour = wdYellow

' myColour = wdNoHighlight

Selection.HomeKey Unit:=wdStory

langText = Languages(Selection.LanguageID).NameLocal

wasIgnore = Options.IgnoreMixedDigits

Options.IgnoreMixedDigits = False

allAlphas = ""

For j = 192 To 255

If j <> 215 And j <> 247 Then allAlphas = allAlphas & ChrW(j)

Next j

For j = 65 To 90

allAlphas = allAlphas & ChrW(j)

Next j

For j = 97 To 122

allAlphas = allAlphas & ChrW(j)

Next j

j = ActiveDocument.Words.Count

Set rng = ActiveDocument.Content

For Each wd In ActiveDocument.Words

If Len(wd) > 2 Then

Do While InStr(ChrW(8217) & "'", Right(wd.Text, 1)) > 0

wd.MoveEnd , -1

DoEvents

Loop

If Right(wd, 2) = ChrW(8217) & "s" Then wd.MoveEnd , -2

If Right(wd, 2) = "'s" Then wd.MoveEnd , -2

If wd.Font.StrikeThrough = False Then

If Application.CheckSpelling(wd, MainDictionary:=langText) = False Then

w = Trim(wd.Text)

cap = Left(w, 1)

isApropernoun = (LCase(cap) <> cap)

' but if it contains numbers, it's not a proper noun

For k = 1 To Len(w)

If Asc(Mid(w, k, 1)) < 65 Then isApropernoun = False: Exit For

Next k

' Check the previous character

rng.Start = wd.Start - 1

rng.End = wd.Start

If rng > "" Then

If Asc(rng) = 13 Or Asc(rng) = 9 Then isApropernoun = False

If Left(rng, 1) = "(" Then rng.MoveStart , -1

rng.MoveStart , -1

If InStr(allAlphas & ";:,", Left(rng, 1)) = 0 Then \_

isApropernoun = False

End If

If isApropernoun = False Then

wd.Font.Underline = True

wd.HighlightColorIndex = wdYellow

End If

End If

End If

j = j - 1

If j Mod 100 = 0 Then

StatusBar = "Spellchecking. To go: " & Str(j)

DoEvents

End If

End If

Next wd

StatusBar = ""

Options.IgnoreMixedDigits = wasIgnore

End Sub

**Sub MultiFileText()**

' Paul Beverley - Version 07.02.23

' Collects text plus simple formatting from multiple files

spaceEquations = True

listOff = True

addFilename = True

convertMathsItems = True

remindAboutCancel = False

myScreenOff = True

CR2 = vbCr & vbCr

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

' First find if this is a Mac or a PC!

myFullName = ActiveDocument.FullName

myName = ActiveDocument.Name

myFolder = Replace(myFullName, myName, "")

myDelimiter = Application.PathSeparator

If myDelimiter = "/" Then

InAMac = True

Else

InAMac = False

End If

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 \_

And InStr(LCase(rng.Text), ".rtf") = 0 \_

And InStr(LCase(rng.Text), ".pdf") = 0 Then

' If not a file list then navigate to the required folder

If remindAboutCancel = True Then myResponse = \_

MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & myDelimiter)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or \_

InStr(LCase(myFile), ".rtf") > 0 Or \_

InStr(LCase(myFile), ".pdf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=myDelimiter, count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Collect unformatted text from ALL the files in" & \_

" directory: " & dirName & "?", vbQuestion + vbYesNoCancel, \_

"Multifile Text Collection")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Collect unformatted text from the files listed here?", \_

vbQuestion + vbYesNoCancel, "Multifile Text Collection")

If myResponse <> vbYes Then Exit Sub

End If

If InAMac = False Then

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Move Left:=20, Top:=300

Application.Resize Width:=1000, Height:=200

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=myDelimiter, count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "=" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

Selection.HomeKey Unit:=wdStory

Set alltextDoc = Documents.Add

Set alltextRng = ActiveDocument.Content

Selection.HomeKey Unit:=wdStory

gotLanguage = False

For i = 1 To numFiles

' Get the folder name, and then the text for the files in the list

StatusBar = allMyFiles(i)

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set myDoc = Application.Documents.Open(fileName:=thisFile)

If gotLanguage = False Then

Selection.MoveEnd , 1

myLanguage = Selection.Range.LanguageID

gotLanguage = True

End If

numEqns = ActiveDocument.OMaths.Count

If numEqns > 0 And spaceEquations = True Then

For Each myMath In ActiveDocument.OMaths

myMath.Range.InsertBefore Text:=" "

DoEvents

numEqns = numEqns - 1

Next myMath

End If

DoEvents

myDoc.Revisions.AcceptAll

myDoc.TrackRevisions = False

If listOff = True Then myDoc.ConvertNumbersToText

If ActiveDocument.Endnotes.Count > 0 Then

Set thisDocRange = myDoc.Content

thisDocRange.Collapse wdCollapseEnd

thisDocRange.FormattedText = \_

myDoc.StoryRanges(wdEndnotesStory).FormattedText

End If

If ActiveDocument.Footnotes.Count > 0 Then

Set thisDocRange = myDoc.Content

thisDocRange.Collapse wdCollapseEnd

thisDocRange.FormattedText = \_

myDoc.StoryRanges(wdFootnotesStory).FormattedText

End If

' copy all the textboxes to the end of the text

shCount = myDoc.Shapes.Count

If shCount > 0 Then

Selection.EndKey Unit:=wdStory

Selection.TypeText vbCr & vbCr

For j = 1 To shCount

Set shp = ActiveDocument.Shapes(j)

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

Set rng = shp.TextFrame.TextRange

Selection.FormattedText = rng.FormattedText

Selection.EndKey Unit:=wdStory

End If

End If

Next

End If

StatusBar = allMyFiles(i)

Selection.HomeKey Unit:=wdStory

DoEvents

If addFilename = True Then Selection.TypeText CR2 & "[[[[ " & \_

allMyFiles(i) & " ]]]]]" & CR2

If convertMathsItems = True Then

numMaths = ActiveDocument.OMaths.Count

If numMaths > 0 Then

For Each myMath In ActiveDocument.OMaths

myMath.Range.Select

eqText = Replace(Selection.Text, vbCr, " ")

Selection.MoveStart , -1

Selection.Delete

startHere = Selection.Start

Selection.TypeText eqText

Selection.Start = startHere

Next myMath

End If

End If

myDoc.Fields.Unlink

' Accept all track changes

myDoc.Range.Revisions.AcceptAll

myDoc.TrackRevisions = False

Set rng = ActiveDocument.Content

DoEvents

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = wdFindContinue

.MatchWildcards = False

.MatchCase = True

.Replacement.Text = "=i=i=^&=j=j="

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Bold = True

.Replacement.Text = "=b=b=^&=c=c="

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Superscript = True

.Replacement.Text = "=u=u=^&=v=v="

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Subscript = True

.Replacement.Text = "=s=s=^&=t=t="

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

alltextRng.End = alltextDoc.Range.End

alltextRng.Start = alltextDoc.Range.End

alltextRng.Text = rng.Text

myDoc.Close SaveChanges:=wdDoNotSaveChanges

Next i

Set rng = alltextDoc.Content

rng.LanguageID = myLanguage

rng.NoProofing = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "=s=s=(\*)=t=t="

.MatchWildcards = True

.Replacement.Text = "\1"

.Replacement.Font.Subscript = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "=u=u=(\*)=v=v="

.Replacement.Text = "=\1"

.Replacement.Font.Superscript = True

.Font.Underline = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "=b=b=(\*)=c=c="

.Replacement.Text = "\1"

.Replacement.Font.Bold = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "=i=i=(\*)=j=j="

.Replacement.Text = "\1"

.Replacement.Font.Italic = True

.Execute Replace:=wdReplaceAll

' Remove optional hyphens

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Replacement.Text = ""

.Text = ChrW(172)

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

totChars = rng.End

' Add one '[' for each file, to check they are all there

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[[[[ "

.Wrap = wdFindContinue

.MatchWildcards = False

.MatchWholeWord = False

.Replacement.Text = "[[[[[ "

.Execute Replace:=wdReplaceAll

End With

ActiveDocument.Paragraphs(3).Range.Bold = True

Set rng = ActiveDocument.Content

totCharsNow = rng.End

If InAPC = True Then

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Resize Width:=1000, Height:=500

End If

Beep

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[[[[["

.Wrap = wdFindContinue

.MatchWildcards = False

.MatchWholeWord = False

.Replacement.Text = ""

.Execute

End With

Selection.Collapse wdCollapseEnd

Application.ScreenUpdating = True

If totCharsNow - totChars <> numFiles Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

MsgBox "Warning: All files might not have been included."

End If

Exit Sub

ReportIt:

Application.Visible = True

On Error GoTo 0

Resume

End Sub

**Sub MultiFileWord()**

' Paul Beverley - Version 28.11.20

' Multiple Word file concatenation

deleteImages = True

insertNotesWithinText = True

embedTextboxText = True

acceptTCs = True

listOff = True

unAllLinkFields = False

unLinkFieldsExceptEqns = False

addTitle = True

myFontSize = 30

titleHighlightColour = wdYellow

remindAboutCancel = False

Dim allMyFiles(200) As String

If deleteImages = False Then moreText = "and images "

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

CR = vbCr

CR2 = CR & CR

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

If remindAboutCancel = True Then myResponse = \_

MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Collect formatted text " & moreText & \_

"from ALL the files in" & " directory:" & dirName & \_

" ?", vbQuestion + vbYesNoCancel, "Multifile Word")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Collect formatted text " & moreText & \_

"from the files listed here?", vbQuestion + vbYesNoCancel, \_

"Multifile Word")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

' Now pick up formatted text from all the files

Set allWordDoc = Documents.Add

Set targetRng = ActiveDocument.Content

For i = 1 To numFiles

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set thisDoc = Application.Documents.Open(FileName:=thisFile)

Set thisRng = thisDoc.Content

StatusBar = allMyFiles(i)

DoEvents

ActiveDocument.TrackRevisions = False

embedTextboxText = True

If listOff = True Then ActiveDocument.ConvertNumbersToText

' Delete all images

If ActiveDocument.InlineShapes.Count > 0 \_

And deleteImages = True Then

For Each myFig In ActiveDocument.InlineShapes

If myFig.Type = wdInlineShapePicture Then myFig.Delete

Next

End If

If insertNotesWithinText = True Then

numNotes = ActiveDocument.Footnotes.Count

If numNotes > 0 Then

thisRng.InsertAfter Text:=CR & "Footnotes:" & CR2

thisRng.Font.Bold = True

thisRng.Collapse wdCollapseEnd

thisRng.FormattedText = \_

thisDoc.StoryRanges(wdFootnotesStory).FormattedText

' Delete all notes

For j = numNotes To 1 Step -1

ActiveDocument.Footnotes(j).Delete

Next j

End If

numNotes = ActiveDocument.Endnotes.Count

If numNotes > 0 Then

thisRng.InsertAfter Text:=CR & "Endnotes:" & CR2

thisRng.Font.Bold = True

thisRng.Collapse wdCollapseEnd

thisRng.FormattedText = \_

thisDoc.StoryRanges(wdEndnotesStory).FormattedText

' Delete all notes

For j = numNotes To 1 Step -1

ActiveDocument.Endnotes(j).Delete

Next j

End If

End If

' copy all the textbox text into the text itself

If embedTextboxText = True And ActiveDocument.Shapes.Count > 0 Then

Selection.EndKey Unit:=wdStory

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

Set rng2 = shp.TextFrame.TextRange

Selection.TypeText CR2

Selection.FormattedText = rng2.FormattedText

End If

End If

Next

For sh = ActiveDocument.Shapes.Count To 1 Step -1

If ActiveDocument.Shapes(sh).TextFrame.HasText Then

ActiveDocument.Shapes(sh).Delete

End If

Next

End If

If unLinkFieldsExceptEqns = True Then

For Each fld In ActiveDocument.Fields

If fld.Type <> 58 Then

fld.Unlink

DoEvents

End If

Next fld

Else

If unAllLinkFields = True Then Selection.Fields.Unlink

End If

' Accept all track changes

If acceptTCs = True Then Selection.Range.Revisions.AcceptAll

' Having closed the file, paste text into combined file

If addTitle = True Then

Selection.HomeKey Unit:=wdStory

fileTitle = allMyFiles(i)

fileTitle = Replace(fileTitle, ".DOCX", "")

fileTitle = Replace(fileTitle, ".docx", "")

fileTitle = Replace(fileTitle, ".doc", "")

fileTitle = Replace(fileTitle, ".DOC", "")

fileTitle = Replace(fileTitle, ".rtf", "")

Selection.TypeText fileTitle & vbCr

Selection.HomeKey Unit:=wdStory

Selection.Expand wdParagraph

Selection.Font.Size = myFontSize

Selection.Font.Bold = True

Selection.Range.HighlightColorIndex = titleHighlightColour

Selection.Start = Selection.End

End If

DoEvents

Set allWordRange = allWordDoc.Content

Set thisDocRange = thisDoc.Content

allWordRange.Collapse wdCollapseEnd

allWordRange.FormattedText = thisDocRange.FormattedText

thisDoc.Close SaveChanges:=wdDoNotSaveChanges

DoEvents

FilesTotal = FilesTotal + 1

Next i

Beep

MsgBox "Please now WAIT until the flashing cursor returns." & vbCr \_

& vbCr & "Not doing so can cause Word to crash." & vbCr & \_

vbCr & " If unsure, only try pressing the cursor right key."

End Sub

**Sub MultiFileReferenceCollator()**

' Paul Beverley - Version 13.12.17

' Collects all references (or foot/endnotes) from multiple files

collectNotes = True

' refTitle = "^pReferences^p"

refTitle = "<H1>References"

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

CR = vbCr

CR2 = CR & CR

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Collect formatted text " & \_

"from ALL the files in" & " directory:" & dirName & \_

" ?", vbQuestion + vbYesNoCancel, "Multifile Word")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Collect formatted text " & moreText & \_

"from the files listed here?", vbQuestion + vbYesNoCancel, \_

"Multifile Word")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

' Now pick up formatted text from all the files

Set theseRefsDoc = Documents.Add

Set theseRefsRng = ActiveDocument.Content

Set allRefsDoc = Documents.Add

Set allRefsRng = ActiveDocument.Content

For i = 1 To numFiles

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set thisdoc = Application.Documents.Open(FileName:=thisFile)

thisFileName = thisdoc.Name

dotPos = InStr(thisFileName, ".")

thisFileName = Left(thisFileName, dotPos - 1)

ActiveDocument.TrackRevisions = False

fnNum = ActiveDocument.Footnotes.Count

enNum = ActiveDocument.Endnotes.Count

If collectNotes = True Then

If fnNum > 0 Then

Set thisRng = thisdoc.StoryRanges(wdFootnotesStory)

StatusBar = allMyFiles(i)

theseRefsRng.FormattedText = thisRng.FormattedText

DoEvents

End If

If enNum > 0 Then

Set thisRng = thisdoc.StoryRanges(wdEndnotesStory)

StatusBar = allMyFiles(i)

theseRefsRng.FormattedText = thisRng.FormattedText

DoEvents

End If

Else

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = refTitle

.MatchCase = False

.MatchWildcards = False

.Execute

End With

If Selection.Find.Found Then

Selection.Collapse wdCollapseEnd

Selection.Start = 0

Selection.Delete

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^12^13]\<"

.MatchCase = False

.MatchWildcards = True

.Execute

End With

If Selection.Find.Found Then

myStart = Selection.Start + 1

Selection.EndKey Unit:=wdStory

Selection.Start = myStart

Selection.Delete

End If

DoEvents

End If

theseRefsRng.Collapse wdCollapseEnd

theseRefsRng.FormattedText = thisdoc.Content.FormattedText

End If

With theseRefsRng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "([!^13])^13"

.Wrap = wdFindContinue

.Replacement.Text = "\1 [" & thisFileName & "]^p"

.Forward = True

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

allRefsRng.Collapse wdCollapseEnd

allRefsRng.FormattedText = theseRefsRng.FormattedText

thisdoc.Close SaveChanges:=wdDoNotSaveChanges

theseRefsRng.Text = ""

Next i

allRefsDoc.Activate

Set rng = ActiveDocument.Content

With rng.Find

.Text = "^m"

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

rng.Sort SortOrder:=wdSortOrderAscending

rng.InsertAfter Text:=vbCr

For Each myPara In ActiveDocument.Paragraphs

ch = myPara.range.Characters(1)

If LCase(ch) <> UCase(ch) Then

myPara.range.Select

Selection.Collapse wdCollapseStart

Exit For

End If

Next myPara

Selection.Start = 0

Selection.Delete

theseRefsDoc.Close SaveChanges:=wdDoNotSaveChanges

allRefsDoc.Activate

Selection.HomeKey Unit:=wdStory

End Sub

**Sub MultifileTrackChangeReport()**

' Paul Beverley - Version 24.09.18

' Creates a file of sentences containing TCs in multiple files

mySpace = vbCr & vbCr

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

On Error GoTo ReportIt

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Extract track-changed sentences from ALL " & \_

"the files in this folder:" & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "Multifile Track-change reporter")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Extract track-changed sentences from just " & \_

"the files listed here?", \_

vbQuestion + vbYesNoCancel, "Multifile Track-change reporter")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

Documents.Add

Set myReport = ActiveDocument

Set rptRng = ActiveDocument.Content

For i = 1 To numFiles

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set sourceText = Application.Documents.Open(FileName:=thisFile)

StatusBar = allMyFiles(i)

DoEvents

ActiveDocument.TrackRevisions = False

myFileName = Left(sourceText.Name, InStr(sourceText.Name, ".") - 1)

rptRng.Text = vbCr & vbCr & "zczc" & myFileName & vbCr & "zczc" & vbCr

rptRng.Collapse wdCollapseEnd

rptRng.InsertAfter Text:=mySpace

rptRng.Collapse wdCollapseEnd

sntEnd = 0

For Each rv In sourceText.Revisions

If rv.range.End > sntEnd Then

Set rng = rv.range.Duplicate

rng.Expand wdSentence

rptRng.FormattedText = rng.FormattedText

rptRng.Collapse wdCollapseEnd

rptRng.InsertAfter Text:=mySpace

rptRng.Collapse wdCollapseEnd

rng.Collapse wdCollapseEnd

sntEnd = rng.End

End If

Next rv

sourceText.Close SaveChanges:=False

Next

myReport.Activate

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc(\*)" & vbCr & "zczc"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Style = wdStyleHeading2

.Forward = True

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Edited sentences from: " & \_

myFolder & vbCr & vbCr & vbCr & vbCr

ActiveDocument.Paragraphs(1).range.Style = wdStyleHeading2

Selection.MoveLeft , 2

Beep

Exit Sub

ReportIt:

If Err.Number = 5825 Then

Resume Next

Else

On Error GoTo 0

Resume

End IfS

End Sub

**Sub CopyTextSimple()**

**'** Paul Beverley - Version 13.06.24

' Creates a text-only copy, with some features preserved

doItalic = True

doBold = True

doSub = True

doSuper = True

doListNumbers = True

myColour = wdGray25

myScreenOff = True

If doListNumbers = True Then ActiveDocument.ConvertNumbersToText

myComments = ""

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

rng.Revisions.AcceptAll

If numberCmnts > 0 Then ActiveDocument.DeleteAllComments

wds1 = rng.Words.Count

rng.Font.Hidden = False

wds2 = rng.Words.Count

If wds2 <> wds1 Then

WordBasic.EditUndo

myResponse = MsgBox("Original contains " & wds2 - wds1 & \_

" words of hidden text. Include it?", \_

vbQuestion + vbYesNoCancel, "CopyText")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbYes Then

ActiveDocument.ActiveWindow.View.ShowHiddenText = True

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Hidden = True

.Wrap = wdFindContinue

.Replacement.Font.Hidden = False

.Replacement.Highlight = True

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

DoEvents

End With

ActiveDocument.ActiveWindow.View.ShowHiddenText = False

End If

End If

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

numNotes = ActiveDocument.Endnotes.Count

If numNotes > 0 Then

myComments = myComments & "| endnotes = yes (" \_

& Trim(Str(numNotes)) & ")" & vbCr

Set rng = ActiveDocument.Content

rng.Collapse wdCollapseEnd

rng.InsertAfter Text:=vbCr & "Endnotes:" & vbCr & vbCr

rng.Collapse wdCollapseEnd

rng.FormattedText = \_

ActiveDocument.StoryRanges(wdEndnotesStory).FormattedText

' Delete all notes

For j = numNotes To 1 Step -1

ActiveDocument.Endnotes(j).Delete

DoEvents

Next j

End If

numNotes = ActiveDocument.Footnotes.Count

If numNotes > 0 Then

myComments = myComments & "| footnotes = yes (" \_

& Trim(Str(numNotes)) & ")" & vbCr

Set rng = ActiveDocument.Content

rng.Collapse wdCollapseEnd

rng.InsertAfter Text:=vbCr & "Footnotes:" & vbCr & vbCr

rng.Collapse wdCollapseEnd

rng.FormattedText = \_

ActiveDocument.StoryRanges(wdFootnotesStory).FormattedText

' Delete all notes

For j = numNotes To 1 Step -1

ActiveDocument.Footnotes(j).Delete

DoEvents

Next j

End If

' copy all the textboxes to the end of the text

shCount = ActiveDocument.Shapes.Count

tbCount = 0

If shCount > 0 Then

Selection.EndKey Unit:=wdStory

Selection.TypeText vbCr & "Textboxes:" & vbCr & vbCr

For j = 1 To shCount

Set shp = ActiveDocument.Shapes(j)

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

Set rng = shp.TextFrame.TextRange

If Len(rng.Text) > 1 Then

tbCount = tbCount + 1

Selection.FormattedText = rng.FormattedText

Selection.EndKey Unit:=wdStory

End If

End If

End If

DoEvents

Next

If tbCount > 0 Then myComments = myComments & \_

"| textboxes = yes (" & Trim(Str(tbCount)) \_

& ")" & vbCr

End If

' Add a newline for safety

Selection.TypeText vbCr

Selection.HomeKey Unit:=wdStory

If myComments > "" Then Selection.TypeText myComments & vbCr

' Watch out for hard spaces and spaced dots for ellipses

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^s"

.Wrap = wdFindContinue

.Replacement.Text = " "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ". . ."

.Wrap = wdFindContinue

.Replacement.Text = "…"

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = wdFindContinue

.Replacement.Text = "vkvk^&kvkv"

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

If doItalic = True Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = wdFindContinue

.Replacement.Text = "zczc^&czcz"

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

If doBold = True Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Bold = True

.Wrap = wdFindContinue

.Replacement.Text = "jqjq^&qjqj"

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

If doSub = True Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Subscript = True

.Wrap = wdFindContinue

.Replacement.Text = "xbxb^&bxbx"

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

If doSuper = True Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Superscript = True

.Wrap = wdFindContinue

.MatchCase = True

.Replacement.Text = "xsxs^&sxsx"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

' Copy pure text after the end of the formatted text

endNow = ActiveDocument.Range.End

Set rngNew = ActiveDocument.Content

rngNew.Collapse wdCollapseEnd

rngNew.Text = rng.Text

' Delete the formatted text

Set rng = ActiveDocument.Content

rng.Collapse wdCollapseStart

' One character less, to avoid the safety newline

rng.End = endNow - 1

rng.Delete

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^12^14]"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

If doSuper = True Then

Set rngNew = ActiveDocument.Content

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "xsxs(\*)sxsx"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Font.Superscript = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "XSXS(\*)SXSX"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Font.Superscript = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

If doSub = True Then

Set rngNew = ActiveDocument.Content

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "xbxb(\*)bxbx"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Font.Subscript = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "XBXB(\*)BXBX"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Font.Subscript = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

If doBold = True Then

Set rngNew = ActiveDocument.Content

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "jqjq(\*)qjqj"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Font.Bold = True

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "JQJQ(\*)QJQJ"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Font.Bold = True

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

If doItalic = True Then

Set rngNew = ActiveDocument.Content

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc(\*)czcz"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Font.Italic = True

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "ZCZC(\*)CZCZ"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Font.Italic = True

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

' Restore Highlight

Set rngNew = ActiveDocument.Content

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "vkvk(\*)kvkv"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Highlight = True

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "VKVK(\*)KVKV"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Highlight = True

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Options.DefaultHighlightColorIndex = oldColour

' Tidy up picture markers

Set rngNew = ActiveDocument.Content

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^1"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{3,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[vkjzcqxbsVKJZCQXBS]{4,}"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

If tbCount = 0 Then

With rngNew.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Textboxes:"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

Beep

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub CopyTextVerySimple()**

' Paul Beverley - Version 28.10.20

' Creates a text-only copy, with no features preserved

Set rngOld = ActiveDocument.Content

myLanguage = rngOld.LanguageID

Set oldDoc = ActiveDocument

Documents.Add

Set rng = ActiveDocument.Content

rng.LanguageID = myLanguage

rng.Text = rngOld.Text

numNotes = oldDoc.Endnotes.Count

If numNotes > 0 Then

Set rng = ActiveDocument.Content

rng.Collapse wdCollapseEnd

rng.InsertAfter Text:=vbCr & "Endnotes:" & vbCr & vbCr

rng.Collapse wdCollapseEnd

rng.Text = \_

oldDoc.StoryRanges(wdEndnotesStory).FormattedText

End If

numNotes = oldDoc.Footnotes.Count

If numNotes > 0 Then

Set rng = ActiveDocument.Content

rng.Collapse wdCollapseEnd

rng.InsertAfter Text:=vbCr & "Footnotes:" & vbCr & vbCr

rng.Collapse wdCollapseEnd

rng.Text = \_

oldDoc.StoryRanges(wdFootnotesStory).FormattedText

End If

' copy all the textboxes to the end of the text

shCount = oldDoc.Shapes.Count

If shCount > 0 Then

Selection.EndKey Unit:=wdStory

Selection.TypeText vbCr & "Textboxes:" & vbCr & vbCr

For j = 1 To shCount

Set shp = oldDoc.Shapes(j)

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

Set rng = shp.TextFrame.TextRange

If Len(rng.Text) > 1 Then

Selection.Text = rng.Text

Selection.EndKey Unit:=wdStory

End If

End If

End If

Next

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^2"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Add a newline for safety

Selection.TypeText vbCr

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub ChapterChopper()**

**'** Paul Beverley - Version 20.10.23

' Chops a chapter into separate files

' On a Mac, you will need something like:

myFolder = "/Users/Paul/My Documents/My book folder/"

' On Windows, you will need something like:

myFolder = "C:\Documents and Settings\Paul\My book folder\"

myFind = "[^13^12][0-9][0-9] "

myMoveStart = 1

myFontSize = 22

myFirstFilename = "00"

myChapterPrefix = "Chapter "

' myChapterPrefix = ""

myResponse = MsgBox("Have you created a ""aaBlank"" file?", \_

vbQuestion + vbYesNo, "ChapterChopper")

If myResponse <> vbYes Then Exit Sub

Set myChapter = ActiveDocument.Content

Set rngSource = ActiveDocument.Content

theVeryEnd = rngSource.End

With rngSource.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

doContinue = False

myNextFilename = myFirstFilename

myCount = 0

Do While rngSource.Find.Found = True

If myMoveStart > 0 Then rngSource.Start = rngSource.Start + myMoveStart

myChapter.End = rngSource.Start

If doContinue = False Then rngSource.Select

If rngSource.Font.Size = myFontSize Then

myCount = myCount + 1

myFileName = myNextFilename

myNextFilename = Trim(rngSource.Text)

myChapter.End = rngSource.Start

Documents.Open fileName:=myFolder & "aaBlank.docx"

Set rng = ActiveDocument.Content

rng.FormattedText = myChapter.FormattedText

If doContinue = False Then

myResponse = MsgBox("Save " & myChapterPrefix & \_

myFileName & " and continue?", \_

vbQuestion + vbYesNo, "ChapterChopper")

If myResponse <> vbYes Then Exit Sub

doContinue = True

End If

ActiveDocument.SaveAs fileName:=myFolder & myChapterPrefix & myFileName

Debug.Print myFileName

DoEvents

ActiveDocument.Close SaveChanges:=False

myChapter.Collapse wdCollapseEnd

End If

rngSource.Collapse wdCollapseEnd

rngSource.Find.Execute

DoEvents

Loop

myFileName = myNextFilename

myChapter.End = theVeryEnd

Documents.Open fileName:=myFolder & "aaBlank.docx"

Set rng = ActiveDocument.Content

rng.FormattedText = myChapter.FormattedText

ActiveDocument.SaveAs fileName:=myFolder & myChapterPrefix & myFileName

ActiveDocument.Close SaveChanges:=False

MsgBox "Files created: " & myCount + 1

End Sub

**Sub FileChopper()**

' Paul Beverley - Version 29.08.15

' Chops text into a number of smaller file using page breaks

firstChapterNumber = 1

' If you want prelims as chapter zero, use

' firstChapterNumber = 0

myPostfix = ""

' For Macs only, use

' myPostfix = ".docx"

' To use ONLY section breaks

' myBreak = "^b"

' To use ONLY page breaks:

' myBreak = "^m"

' To use EITHER page breaks or section breaks:

myBreak = "^m^b"

myPrefix = InputBox("Filename prefix?", "File Chopper", "Chapter")

If InStr(myBreak, "^m") > 0 Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^m"

.Wrap = False

.Replacement.Text = "^&zczczc"

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

End If

If InStr(myBreak, "^b") > 0 Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^b"

.Wrap = False

.Replacement.Text = "^&zczczc"

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

End If

' Go and find the first break

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczczc"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

rng.Collapse wdCollapseStart

startChapter = 0

myCount = firstChapterNumber

' Split at each subsequent break

Do

rng.Find.Execute

If rng.Find.Found = False Then

rng.Start = endChapter + 7

rng.End = ActiveDocument.range.End

Else

endChapter = rng.End - 7

rng.Start = startChapter

rng.End = endChapter

End If

rng.Copy

Documents.Add

Selection.Paste

myFile = Trim(Str(myCount))

If Len(myFile) = 1 Then myFile = "0" & myFile

myFile = myPrefix & myFile & myPostfix

ActiveDocument.SaveAs FileName:=myFile

ActiveDocument.Close

rng.End = endChapter + 7

rng.Collapse wdCollapseEnd

startChapter = rng.Start

rng.Select

myCount = myCount + 1

Loop Until rng.Find.Found = False

WordBasic.editundo

MsgBox "Final chapter number: " & myCount - 1

End Sub

**Sub FileLister()**

' Paul Beverley - Version 19.09.18

' Lists all files in a folder

showAllFiles = True

myResponse = MsgBox("Navigate to chosen folder; then click 'Cancel'" \_

, , "FileLister")

Dialogs(wdDialogFileOpen).Show

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 \_

Or showAllFiles Then Selection.TypeText myFile & vbCr

myFile = Dir()

Loop

Selection.HomeKey Unit:=wdStory

Selection.InsertAfter Text:=dirPath & vbCr & vbCr

Selection.Collapse wdCollapseStart

End Sub

**Sub AcronymAlyse()**

' Paul Beverley - Version 27.11.21

' Lists all acronyms, with frequency

doBeeps = True

minCount = 3

myColour = wdGray25

deleteTableBorders = True

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

myResponse = MsgBox("Acronym Lister: Include numbers?", \_

vbQuestion + vbYesNoCancel)

If myResponse = vbCancel Then Exit Sub

' First create a copy of the existing document

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Selection.HomeKey Unit:=wdStory

Selection.TypeParagraph

' Remove all apostrophes (close single quotes)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = True

.Replacement.Text = " "

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

allText = rng.Text

rng.Delete

Selection.InsertAfter Text:=allText

' Remove all apostrophes (close single quotes)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & ChrW(8217) & "']"

.Replacement.Text = " "

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Add highlight to all words with/without numbers

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

If myResponse = vbYes Then

.Text = "<[0-9A-Za-z]{2,}>"

Else

.Text = "<[A-Za-z]{2,}>"

End If

.Replacement.Text = "^&"

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

If doBeeps = True Then Beep

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

If myResponse = vbYes Then

.Text = "<[0-9A-Za-z]{1,}-[0-9A-Za-z]{1,}>"

Else

.Text = "<[A-Za-z]{1,}-[A-Za-z]{1,}>"

End If

.Replacement.Text = "^&"

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

If doBeeps = True Then Beep

' Remove highlight from words in all lowercase

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[a-z0-9]@>"

.Replacement.Text = "^&"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

If doBeeps = True Then Beep

' Remove highlight from words in all lowercase

' followed by a hyphen

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[a-z0-9]@>\-"

.Replacement.Text = "^&"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

If doBeeps = True Then Beep

' Remove highlight from initial-cap words

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][\-a-z]@>"

.Replacement.Text = "^&"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Remove highlight from words ending in hyphen

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[-A-Za-z]@->"

.Replacement.Text = "^&"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Remove all unhighlighted characters

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = False

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Remove all blank lines

Set rng = ActiveDocument.Content

wasEnd = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13{2,}"

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Execute Replace:=wdReplaceAll

.Execute Replace:=wdReplaceAll

End With

DoEvents

rng.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

rng.HighlightColorIndex = 0

rng.InsertAfter Text:=vbCr & vbCr

Set rng = ActiveDocument.Content

' Remove all single hyphens

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p-"

.Highlight = False

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Remove all blank lines (again)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13{2,}"

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Execute Replace:=wdReplaceAll

.Execute Replace:=wdReplaceAll

End With

DoEvents

Selection.HomeKey Unit:=wdStory

ct = 1

oldWord = ActiveDocument.Paragraphs(2).Range.Text

For i = 3 To ActiveDocument.Paragraphs.Count

newWord = Trim(ActiveDocument.Paragraphs(i).Range.Text)

If newWord = oldWord Then

ct = ct + 1

Else

Set rng = ActiveDocument.Paragraphs(i - 1).Range

rng.End = rng.End - 1

rng.InsertAfter Text:=vbTab & Trim(Str(ct))

oldWord = newWord

If ct < minCount Then rng.HighlightColorIndex = myColour

ct = 1

End If

Next i

For i = ActiveDocument.Paragraphs.Count To 1 Step -1

Set pa = ActiveDocument.Paragraphs(i).Range

If InStr(pa.Text, vbTab) = 0 Then

ActiveDocument.Paragraphs(i).Range.Delete

End If

Next i

ActiveDocument.Content.InsertAfter vbCr

' Add title and set as table

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Acronym list" & vbCr

startTable = Selection.End

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

ActiveDocument.Paragraphs(1).Range.HighlightColorIndex \_

= wdNoHighlight

Selection.Start = startTable

Selection.End = ActiveDocument.Range.End

Selection.ConvertToTable Separator:=wdSeparateByTabs

Set tb = ActiveDocument.Tables(1)

tb.AutoFitBehavior (wdAutoFitContent)

If deleteTableBorders = True Then

tb.Borders(wdBorderTop).LineStyle = wdLineStyleNone

tb.Borders(wdBorderLeft).LineStyle = wdLineStyleNone

tb.Borders(wdBorderBottom).LineStyle = wdLineStyleNone

tb.Borders(wdBorderRight).LineStyle = wdLineStyleNone

tb.Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

tb.Borders(wdBorderVertical).LineStyle = wdLineStyleNone

End If

Selection.HomeKey Unit:=wdStory

If doBeeps = True Then Beep

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub AcronymLister()**

' Paul Beverley - Version 17.09.18

' List all acronyms

createFReditList = False

doBeeps = True

myResponse = MsgBox("Acronym Lister: Include numbers?", \_

vbQuestion + vbYesNoCancel)

If myResponse = vbCancel Then Exit Sub

If Options.DefaultHighlightColorIndex = 0 Then \_

Options.DefaultHighlightColorIndex = wdYellow

' First create a copy of the existing document

Set rng = ActiveDocument.Content

Documents.Add

Selection.TypeParagraph

Selection.Text = rng.Text

Selection.HomeKey Unit:=wdStory

' Remove all apostrophes (close single quotes)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & ChrW(8217) & "']"

.Replacement.Text = " "

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

' Add highlight to all words with/without numbers

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

If myResponse = vbYes Then

.Text = "<[-0-9A-Za-z]{2,}>"

Else

.Text = "<[-A-Za-z]{2,}>"

End If

.Replacement.Text = "^&"

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

If doBeeps = True Then Beep

' Remove highlight from words in all lowercase

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[-a-z0-9]@>"

.Replacement.Text = "^&"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

If doBeeps = True Then Beep

' Remove highlight from initial-cap words

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z][-a-z]@>"

.Replacement.Text = "^&"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Remove highlight from words ending in hyphen

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[-A-Za-z]@->"

.Replacement.Text = "^&"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Remove highlight from words starting in hyphen

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<-[-A-Za-z]@>"

.Replacement.Text = "^&"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Remove all unhighlighted characters

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = False

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Remove all blank lines

Do

Set rng = ActiveDocument.Content

wasEnd = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13{2,}"

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Loop Until wasEnd = rng.End

' Create a list of unique abbrevs

Set dict = CreateObject("Scripting.Dictionary")

'Set comparemode; use vbBinaryCompare

'for case-sensitive filtering

dict.comparemode = vbTextCompare

'Iterate through all the paragraphs in the doc.

For Each myPara In ActiveDocument.Paragraphs

'If we've already encountered this item,

'then delete the paragraph.

If dict.Exists(myPara.range.Text) Then

myPara.range.Delete

Else

'If we haven't already encountered this item,

'then add it to the dictionary's keys.

dict.Add myPara.range.Text, ""

End If

Next myPara

Set dict = Nothing

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.range.HighlightColorIndex = 0

Selection.HomeKey Unit:=wdStory

If createFReditList = False Then

If doBeeps = True Then Beep

Exit Sub

End If

' Create a FRedit list

ActiveDocument.Content.Copy

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Replacement.Text = "zczc^p"

.Replacement.Highlight = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13(\*)zczc"

.Replacement.Text = "^p\1|^^&"

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc"

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Selection.range.HighlightColorIndex = wdColorAutomatic

Selection.InsertAfter Text:=vbCr & "| FRedit list:" & vbCr

Selection.HomeKey Unit:=wdStory

Selection.Paste

If doBeeps = True Then Beep

End Sub

**Sub SortIt()**

' Paul Beverley - Version 02.11.19

' Sort the selected text

anyCase = False

promptForWholeText = False

If Selection.End = Selection.Start And promptForWholeText \_

= True Then

myResponse = MsgBox("Sort the WHOLE file?", \_

vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

Selection.WholeStory

End If

Selection.Sort ExcludeHeader:=False, FieldNumber:="Paragraphs", \_

CaseSensitive:=Not (anyCase), SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

End Sub

**Sub** **DuplicatesRemove()**

' Paul Beverley - Version 26.01.24

' Remove duplicate items from a list

' Make this True if you want to completely delete

' any lines that appear more than once

' removeBothDuplicates = True

removeBothDuplicates = False

anyCase = True

' Exchange rogue linefeed (ascii 11)

Set rng = ActiveDocument.Content

rng.InsertAfter Text:=vbCr & vbCr

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^11"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

If Selection.End = Selection.Start Then

myResponse = MsgBox("Remove duplicates from the WHOLE file?", \_

vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

listEnd = ActiveDocument.Paragraphs.Count

listStart = 1

Else

myStart = Selection.Start

myEnd = Selection.End

For j = 1 To ActiveDocument.Paragraphs.Count

If ActiveDocument.Paragraphs(j).Range.Start > myStart Then Exit For

Next j

listStart = j - 1

For j = ActiveDocument.Paragraphs.Count To 1 Step -1

If ActiveDocument.Paragraphs(j).Range.Start < myEnd Then Exit For

Next j

listEnd = j

End If

For j = listEnd - 1 To listStart + 1 Step -1

Set rng1 = ActiveDocument.Paragraphs(j).Range

Set rng2 = ActiveDocument.Paragraphs(j - 1).Range

If anyCase = True Then

r1 = LCase(rng1)

r2 = LCase(rng2)

Else

r1 = rng1

r2 = rng2

End If

' If r1 = r2 Then rng1.Font.Bold = True: rng2.Font.Bold = True

If r1 = r2 Then

rng1.Delete

If removeBothDuplicates = True Then

rng1.Font.Underline = True

rng2.Font.Underline = True

End If

End If

StatusBar = "Lines to go: " & Str(j)

DoEvents

Next j

StatusBar = ""

If removeBothDuplicates = True Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

ActiveDocument.Paragraphs(listStart).Range.Select

Selection.End = Selection.Start

End Sub

**Sub SortAndRemoveDups()**

**'** Paul Beverley - Version 19.09.18

' Sort the selected text

If Selection.End = Selection.Start Then

myResponse = MsgBox("Sort the WHOLE file?", \_

vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

Selection.WholeStory

End If

myStart = Selection.Start

myEnd = Selection.End

' Exchange rogue linefeed (ascii 10)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^10"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.Sort ExcludeHeader:=False, FieldNumber:="Paragraphs", SortOrder:=wdSortOrderAscending

Selection.Start = myStart

Selection.End = myEnd

For j = 1 To ActiveDocument.Paragraphs.Count

If ActiveDocument.Paragraphs(j).Range.Start > myStart Then Exit For

Next j

listStart = j - 1

For j = ActiveDocument.Paragraphs.Count To 1 Step -1

If ActiveDocument.Paragraphs(j).Range.Start < myEnd Then Exit For

Next j

listEnd = j

For j = listEnd To listStart + 1 Step -1

Set rng1 = ActiveDocument.Paragraphs(j).Range

Set rng2 = ActiveDocument.Paragraphs(j - 1).Range

If rng1 = rng2 Then rng1.Delete

StatusBar = "Lines to go: " & Str(j)

Next j

StatusBar = ""

ActiveDocument.Paragraphs(listStart).Range.Select

Selection.End = Selection.Start

End Sub

**Sub SortCaseSense()**

**'** Paul Beverley - Version 17.02.11

' Sort into separate lists: Lcase/Ucase

' <ctrl-alt-shift-S>

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13([A-Z])"

.Replacement.Text = "^pzzzz\1"

.Forward = True

.Wrap = wdFindContinue

.Format = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zzzz"

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindContinue

.Format = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

End Sub

**Sub SortTextBlocks()**

' Paul Beverley - Version 28.10.17

' Alpha sorts blocks of text by first line

If Selection.Start = Selection.End Then

doingAll = True

Set rng = ActiveDocument.Content

Else

doingAll = False

Selection.Copy

Documents.Add

Selection.Paste

Set rng = ActiveDocument.Content

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{3,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p^p"

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p^p"

.Wrap = wdFindContinue

.Replacement.Text = "zczc"

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Replacement.Text = "pqpq"

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc"

.Wrap = wdFindContinue

.Replacement.Text = "zczc^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

rng.Sort ExcludeHeader:=False, FieldNumber:="Paragraphs"

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "pqpq"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

If doingAll = False Then

Selection.WholeStory

Selection.Copy

ActiveDocument.Close SaveChanges:=False

Selection.Paste

End If

End Sub

**Sub SortNumberedList()**

' Paul Beverley - Version 03.07.12

' Sort references, ignoring the number at the beginning

numChars = 20

myDelimiter = " "

' If it's a tab, use...

' myDelimiter = Chr(9)

For Each myPara In ActiveDocument.Paragraphs

If Len(myPara.Range.Text) > numChars Then

myPara.Range.Select

firstChars = Left(myPara.Range.Text, numChars)

firstChars = Right(firstChars, Len(firstChars) - InStr(firstChars, myDelimiter))

Selection.InsertBefore Text:="xx" & firstChars & "yy"

End If

Next myPara

Selection.WholeStory

Selection.Sort ExcludeHeader:=False, FieldNumber:="Paragraphs", \_

SortOrder:=wdSortOrderAscending, SortFieldType:=wdSortFieldAlphanumeric

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "xx\*yy"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub BibSortWithDittos()**

' Paul Beverley - Version 07.07.20

' Sorts bibliographic list including ditto marks

' myDitto1 = "-^t"

myDitto2 = "^+^+^+"

myDitto1 = "<bib>--"

' Don't do this with track changes on!

ActiveDocument.TrackRevisions = False

' Can't use auto lists, either!

Call AutoListOff

If Selection.Start <> Selection.End Then

Set rng = Selection.Range.Duplicate

Else

Set rng = ActiveDocument.Content

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13" & ChrW(8220) & "([a-zA-Z ]{1,})"

.Wrap = wdFindStop

.Replacement.Text = "^p\1vbvb"

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<bib>" & ChrW(8220)

.Wrap = wdFindStop

.Replacement.Text = "<bib>"

.Replacement.Font.Underline = True

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p" & myDitto1

.Wrap = wdFindStop

.Replacement.Text = "zczc"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p" & myDitto2

.Wrap = wdFindStop

.Replacement.Text = "cqcq"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

rng.Sort SortOrder:=wdSortOrderAscending

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc"

.Wrap = wdFindStop

.Replacement.Text = "^p" & myDitto1

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "cqcq"

.Wrap = wdFindStop

.Replacement.Text = "^p" & myDitto2

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.Text = "^13([a-zA-Z ]{1,})vbvb"

.Wrap = wdFindStop

.Replacement.Text = "^p" & ChrW(8220) & "\1"

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Font.Underline = True

.Text = "<bib>"

.Wrap = wdFindStop

.Replacement.Text = "<bib>" & ChrW(8220)

.Replacement.Font.Underline = False

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End Sub

**Sub SortOnTextAfterDelimiter()**

' Paul Beverley - Version 28.10.17

' Sorts the selected list based on second 'field'

If Selection.End = Selection.Start Then

myResponse = MsgBox("Sort the whole file?", \_

vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{2,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

Else

Set rng = Selection.range.Duplicate

End If

myChar = InputBox("Sort character?" & vbCr, "List sorter")

If Len(myChar) > 0 Then

Select Case myChar

Case ChrW(8211): myChar = "^="

Case ChrW(8212): myChar = "^+"

End Select

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myChar

.Wrap = wdFindStop

.Replacement.Text = myChar & "|" & vbTab

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

End If

rng.Sort FieldNumber:="Field 2"

If Len(myChar) > 0 Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = myChar & "|" & vbTab

.Replacement.Text = myChar

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

End If

End Sub

**Sub SurnameSorter()**

' Paul Beverley - Version 01.10.18

' Sorts a name list on surname, but allowing for postfixes

myPostfixes = "| BSc| MSc| OBE|, Jr.|, Sr.|"

myWd = Split(myPostfixes, "|")

Set rng = ActiveDocument.Content

For i = 0 To UBound(myWd)

wd = myWd(i)

If Len(wd) > 0 Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = wd

.Wrap = wdFindContinue

.Replacement.Text = "qpqp" & Trim(Str(i)) & "qpqp"

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

Next i

For i = 1 To ActiveDocument.Paragraphs.Count

Set pa = ActiveDocument.Paragraphs(i).range

If pa.Words.Count > 2 Then

sName = pa.Words(pa.Words.Count - 1)

Debug.Print sName

pa.InsertBefore Text:="zczc" & sName & vbTab

End If

Next i

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

For i = 0 To UBound(myWd)

wd = myWd(i)

If Len(wd) > 0 Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = "qpqp" & Trim(Str(i)) & "qpqp"

.Replacement.Text = wd

.Execute Replace:=wdReplaceAll

End With

End If

Next i

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = "zczc\*^t"

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

rng.InsertAfter Text:=vbCr

End Sub

**Sub SortListInText()**

' Paul Beverley - Version 16.01.21

' Sorts items in a list in the text alphabetically.

' This is based on an idea by Gary Smith

' If last item might possibly have a conjunction only

' e.g. "fish, chips and peas", then say "True"

allowSplitAtConjunction = True

If Selection.Start = Selection.End Then

Selection.MoveStartUntil cset:="(", Count:=wdBackward

Selection.MoveEndUntil cset:=")", Count:=wdForward

Else

theEnd = Selection.End

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

theStart = Selection.Start

Selection.End = theEnd

Selection.Collapse wdCollapseEnd

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = theStart

End If

myList = Selection

myDelim = ", "

If InStr(myList, ";") > 0 Then myDelim = "; "

myBits = Split(myList, myDelim)

lastItem = UBound(myBits)

ReDim myGroup(lastItem) As String

myGroup = myBits

' Is the final "item" really a single item, or rather

' a pair, separated by "and" or "&"?

lastText = myGroup(lastItem)

serialcomma = True

If InStr(lastText, " and ") + InStr(lastText, " & ") > 0 \_

And allowSplitAtConjunction = True Then

myResponse = MsgBox(lastText & vbCr & "Is this a single item?", \_

vbQuestion + vbYesNoCancel, "SortListInText")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then

myList = Replace(myList, " & ", ", & ")

myList = Replace(myList, " and ", ", and ")

myBits = Split(myList, myDelim)

lastItem = UBound(myBits)

myGroup = myBits

lastText = myGroup(lastItem)

serialcomma = False

End If

End If

spPos = InStr(lastText, " ")

If spPos > 0 Then firstWd = Left(lastText, spPos) Else firstWd = ""

' If there's a conjunction, delete it

If firstWd = "and " Or firstWd = "& " Then

lastText = myGroup(lastItem)

lastText = Mid(lastText, Len(firstWd) + 1)

myGroup(lastItem) = lastText

Else

firstWd = ""

End If

WordBasic.SortArray myGroup()

tempStr = ""

For i = 0 To lastItem - 1

tempStr = tempStr & myGroup(i) & myDelim

Next i

If serialcomma = False Then

tempStr = Left(tempStr, Len(tempStr) - 2) & " "

End If

tempStr = tempStr & firstWd & myGroup(i)

etalItalic = (Selection.Font.Italic > 0)

myStart = Selection.Start

Selection.TypeText tempStr

If etalItalic = True Then

Selection.Start = myStart

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "et al"

.Wrap = wdFindStop

.Replacement.Text = "^&"

.Replacement.Font.Italic = True

.Forward = True

.Execute Replace:=wdReplaceAll

End With

Selection.Collapse wdCollapseEnd

End If

End Sub

**Sub CitationListSortByYear()**

' Paul Beverley - Version 16.01.21

' Sorts items in a citation list in the text by date.

sortReversed = True

If Selection.Start = Selection.End Then

Selection.MoveStartUntil cset:="(", Count:=wdBackward

Selection.MoveEndUntil cset:=")", Count:=wdForward

Else

theEnd = Selection.End

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

theStart = Selection.Start

Selection.End = theEnd

Selection.Collapse wdCollapseEnd

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = theStart

End If

myList = Selection

myDelim = ", "

If InStr(myList, ";") > 0 Then myDelim = "; "

myBits = Split(myList, myDelim)

lastItem = UBound(myBits)

ReDim myGroup(lastItem) As String

myGroup = myBits

For i = 0 To lastItem

txt = myGroup(i)

myYear = Right(txt, 5)

myGroup(i) = myYear & Left(txt, Len(txt) - 5)

Next i

If sortReversed Then

WordBasic.SortArray myGroup(), 1

Else

WordBasic.SortArray myGroup(), 0

End If

For i = 0 To lastItem

txt = myGroup(i)

myYear = Left(txt, 5)

myGroup(i) = Mid(txt, 6) & myYear

Next i

tempStr = ""

For i = 0 To lastItem - 1

tempStr = tempStr & myGroup(i) & myDelim

Next i

tempStr = tempStr & myGroup(i)

Selection.TypeText tempStr

End Sub

**Sub CopyToList()**

**'** Paul Beverley - Version 01.09.23

' Copies selected text into a list file

addReturn = True

' addBlankLine = True

addBlankLine = False

' keyWord = "queries"

keyWord = "list"

wordsToAvoid = "switch"

' wordsToAvoid = "FRedit,switch"

copyWholePara = False

copyWholePara = True

copySentenceInstead = False

includeFormatting = True

myHighlightColour = wdColorBlack

' To add highlight, use:

' myHighlightColour = wdYellow

goBackToSource = True

alwaysCopyAtEnd = False

Set thisDoc = ActiveDocument

wds = Split("," & LCase(wordsToAvoid), ",")

If Selection.Start = Selection.End Then

If copyWholePara = True Then

If copySentenceInstead = True Then

Selection.Expand wdSentence

Else

Selection.Expand wdParagraph

End If

Else

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

Else

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.MoveEnd , -1

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

rng.Select

End If

Set sourceText = Selection.Range

gottaList = False

For Each myDoc In Application.Documents

thisName = myDoc.Name

nm = LCase(thisName)

gottaList = False

If InStr(nm, LCase(keyWord)) > 0 Then gottaList = True

For i = 1 To UBound(wds)

If InStr(nm, wds(i)) > 0 Then gottaList = False

Next i

If gottaList = True Then

myDoc.Activate

Exit For

End If

Next myDoc

If gottaList = False Then

Beep

MsgBox "Can't find a list."

Exit Sub

End If

If alwaysCopyAtEnd = True Then

Selection.EndKey Unit:=wdStory

Else

hereNow = Selection.Start

Selection.Expand wdParagraph

If Selection.Start = hereNow Then Selection.Collapse wdCollapseStart

If Len(Selection) = 1 Then

Selection.Collapse wdCollapseStart

Else

Selection.Collapse wdCollapseEnd

End If

End If

If includeFormatting = True Then

Selection.Range.FormattedText = sourceText.FormattedText

Else

Selection.Text = sourceText.Text

End If

Selection.MoveRight , Len(sourceText)

If InStr(sourceText, vbCr) = 0 Then

Selection.TypeText vbCr

End If

If addBlankLine = True Then

Selection.TypeText vbCr

End If

If goBackToSource = True Then thisDoc.Activate

If myHighlightColour > 0 Then

sourceText.HighlightColorIndex = myHighlightColour

Selection.Collapse wdCollapseEnd

End If

End Sub

**Sub AcronymFinder()**

**'** Paul Beverley - Version 11.02.11

' Find a group of words for an acronym

myText = Trim(Selection)

Selection.Start = Selection.End

If Len(myText) = 1 Then

myText = InputBox("Find?", "Acronym finder", Selection)

If Len(myText) = 0 Then Exit Sub

End If

myFind = "<"

abbrLength = Len(myText)

For i = 1 To abbrLength

If i = 4 Then Exit For

myChar = Mid(myText, i, 1)

myChar = "[" & LCase(myChar) & UCase(myChar) & "]"

myFind = myFind & myChar & "[a-z]{1,}^32"

Next i

myFind = Left(myFind, Len(myFind) - 3)

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

If Selection.Find.Found = True Then

If abbrLength = 4 Then Selection.MoveRight Unit:=wdWord, Count:=2, Extend:=wdExtend

If abbrLength = 5 Then Selection.MoveRight Unit:=wdWord, Count:=3, Extend:=wdExtend

Exit Sub

End If

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Text = "[[["

.MatchWildcards = False

.Execute

End With

Beep

Selection.MoveUp Unit:=wdLine, Count:=2

Selection.MoveDown Unit:=wdLine, Count:=2

Selection.Find.Execute

Selection.Find.Text = myFind

Selection.Find.MatchWildcards = True

End Sub

**Sub AcronymDefinitionLister()**

' Paul Beverley - Version 17.09.17

' Creates a list of acronyms with definitions

myBuffer = 2

myMax = 8

myMaxText = Trim(Str(myMax))

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

Selection.HomeKey unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\<\*\>"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

' Go and find the first acronym in brackets

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\([A-Za-z0-9\-" & ChrW(8211) & "]{2," & myMaxText & "}\)"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

allAcronyms = ""

Do While rng.Find.Found = True

myPosn = rng.End

myAcronym = Replace(rng, "(", "")

myAcronym = Replace(myAcronym, ")", "")

lenAcro = Len(myAcronym)

myAcronym = myAcronym & "xzx"

myAcronym = Replace(myAcronym, "sxzx", "")

myAcronym = Replace(myAcronym, "xzx", "")

rng.Collapse wdCollapseStart

If LCase(myAcronym) <> UCase(myAcronym) And LCase(Mid(myAcronym, 2)) <> Mid(myAcronym, 2) Then

rng.Select

Selection.MoveLeft unit:=wdCharacter, Count:=1

Selection.MoveStart unit:=wdWord, Count:=-lenAcro - myBuffer

myDefn = Trim(Selection)

returnPos = InStr(myDefn, Chr(13))

myDefn = Mid(myDefn, returnPos + 1)

If Left(myDefn, 3) = "in " Then myDefn = Mid(myDefn, 4)

If Left(myDefn, 3) = "on " Then myDefn = Mid(myDefn, 4)

If Left(myDefn, 3) = "of " Then myDefn = Mid(myDefn, 4)

If Left(myDefn, 4) = "the " Then myDefn = Mid(myDefn, 5)

If Left(myDefn, 4) = "and " Then myDefn = Mid(myDefn, 5)

If Left(myDefn, 4) = "The " Then myDefn = Mid(myDefn, 5)

If Left(myDefn, 2) = "a " Then myDefn = Mid(myDefn, 3)

If Left(myDefn, 2) = ". " Then myDefn = Mid(myDefn, 3)

allAcronyms = allAcronyms & myAcronym & Chr(9) & myDefn & vbCr

End If

rng.Start = myPosn

' Go and find the next occurence (if there is one)

rng.Find.Execute

posNow = rng.End

If posNow = posWas Then

rng.Select

Selection.MoveDown unit:=wdLine, Count:=1

rng.Start = Selection.Start

rng.Find.Execute

posNow = rng.End

End If

posWas = posNow

Loop

Selection.WholeStory

' type the acronyms to replace the copy of the text

Selection.TypeText allAcronyms

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.HomeKey unit:=wdStory

' Remove duplicate line

For j = ActiveDocument.Paragraphs.Count To 2 Step -1

Set rng1 = ActiveDocument.Paragraphs(j).range

Set rng2 = ActiveDocument.Paragraphs(j - 1).range

If rng1 = rng2 Then rng1.Delete

Next j

Selection.HomeKey unit:=wdStory

Selection.TypeText Text:="Acronym list" & vbCr & vbCr

Beep

End Sub

**Sub AcronymToSmallCaps()**

' Paul Beverley - Version 31.12.13

' Converts all acronyms to small caps

myColour = False

' or if you want them highlighted:

' myColour = wdTurquoise

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z]{3,}"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

myCount = 0

Do While Selection.Find.Found = True

Selection.Text = LCase(Selection.Text)

If myColour <> False then Selection.Range.HighlightColorIndex = myColour

Selection.Font.SmallCaps = True

Selection.Collapse wdCollapseEnd

Selection.Find.Execute

Loop

End Sub

**Sub TaggedTextToSmallCaps()**

' Paul Beverley - Version 01.02.20

' Finds tagged text, lowercases it and changes to small caps

tagPre = "<sc>"

tagPost = "</sc>"

removeTags = True

' thinCode = "<th>/</th>"

thinCode = ""

tPr = Replace(tagPre, "\", "\\")

tPr = Replace(tPr, "<", "\<")

tPr = Replace(tPr, ">", "\>")

tPo = Replace(tagPost, "\", "\\")

tPo = Replace(tPo, "<", "\<")

tPo = Replace(tPo, ">", "\>")

mySearch = tPr & "\*" & tPo

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySearch

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

newText = Replace(rng.Text, tagPre, "")

newText = Replace(newText, tagPost, "")

newText = LCase(newText)

rng.Text = newText

rng.Font.SmallCaps = True

myEnd = rng.End

If removeTags = False Then

rng.InsertBefore Text:=tagPre

rng.InsertAfter Text:=tagPost

myEnd = myEnd + Len(tagPre) + Len(tagPost)

rng.End = rng.Start + Len(tagPre)

rng.Font.SmallCaps = False

rng.End = myEnd

rng.Start = myEnd - Len(tagPost)

rng.Font.SmallCaps = False

End If

rng.Start = myEnd

rng.Find.Execute

DoEvents

Loop

If Len(thinCode) > 0 Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = thinCode

.Wrap = wdFindContinue

.Replacement.Text = ChrW(8201)

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

MsgBox "Changed: " & myCount & " small caps"

End Sub

**Sub TagsShowHide()**

' Paul Beverley - Version 08.09.20

' Changes tags into hidden text and then reveals them again

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\<[/A-Z1-5\-]{2,}\>"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Hidden = True

.Forward = True

.MatchWildcards = True

.Execute

DoEvents

If .Found = False Then

rng.Font.Hidden = False

Else

.Execute Replace:=wdReplaceAll

End If

End With

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub TableStripper()**

' Paul Beverley - Version 13.01.21

' Strips out all tables into a separate file

myFormat = "[xxx near here]"

Set thisDoc = ActiveDocument

Documents.Add

Set tabDoc = ActiveDocument

thisDoc.Activate

Selection.HomeKey Unit:=wdStory

thisMany = 0

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Table "

.Replacement.Text = ""

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While Selection.Find.Found = True

myCaptionStart = Selection.Start

myNumberStart = Selection.End

Selection.Expand wdParagraph

' Only do anything if 'Table' is the first word of the paragraph

If Selection.Start = myCaptionStart Then

myCaptionEnd = Selection.End

tabCaption = Selection

' Move back and pick up, say, 'Table 1.6'

Selection.End = myNumberStart

Do

Selection.MoveEnd , Count:=1

DoEvents

Loop Until InStr(ChrW(9) & " :", Right(Selection.Text, 1)) > 0

tabTitle = Selection

' now check for table below

Selection.Collapse wdCollapseEnd

Selection.MoveDown Unit:=wdParagraph, Count:=2

gotOne = False

If Selection.Information(wdWithInTable) = True Then

gotOne = True

Do

Selection.MoveDown Unit:=wdParagraph, Count:=1

Loop Until Selection.Information(wdWithInTable) = False

Selection.MoveUp Unit:=wdParagraph, Count:=1

Selection.Start = myCaptionEnd

Selection.Cut

Else

Selection.Start = myCaptionStart

Selection.End = myCaptionStart

Selection.MoveUp Unit:=wdParagraph, Count:=2

If Selection.Information(wdWithInTable) = True Then

gotOne = True

Do

Selection.MoveUp Unit:=wdLine, Count:=1

Loop Until Selection.Information(wdWithInTable) = False

Selection.MoveDown Unit:=wdParagraph, Count:=1

Selection.End = myCaptionStart

Selection.Cut

Selection.MoveDown Unit:=wdParagraph, Count:=1

End If

End If

' Paste it into the tables document

If gotOne = True Then

Selection.InsertAfter Replace(myFormat, "xxx", tabTitle) & vbCr

Selection.HomeKey Unit:=wdLine

tabDoc.Activate

Selection.TypeText tabCaption

Selection.Paste

Selection.TypeParagraph

Selection.TypeParagraph

thisMany = thisMany + 1

thisDoc.Activate

Selection.MoveDown Unit:=wdParagraph, Count:=1

Else

Selection.MoveDown Unit:=wdParagraph, Count:=2

End If

Else

Selection.Collapse wdCollapseEnd

End If

Selection.Find.Execute

Loop

tabDoc.Activate

Selection.TypeText Str(thisMany) & " tables extracted" & vbCr

End Sub

**Sub TablesToTabText()**

' Paul Beverley - Version 27.06.17

' Converts all tables into tab-separated text

For i = ActiveDocument.Tables.Count To 1 Step -1

ActiveDocument.Tables(i).range.Select

Selection.Cut

Selection.InsertBefore Text:=vbCr

Selection.range.Style = ActiveDocument.Styles(wdStyleNormal)

Selection.Collapse wdCollapseStart

Selection.PasteSpecial DataType:=wdPasteText

Next i

End Sub

**Sub FigCallouts()**

' Paul Beverley - Version 12.03.13

' Figure callout inserter

thisChapter = InputBox("Chapter number?", "Figure callouts")

' chapNumsExist = MsgBox("Existing chapter numbers?", vbQuestion + vbYesNo)

' addChapNums = MsgBox("Add chapter numbers?", vbQuestion + vbYesNo)

chapNumsExist = vbNo

addChapNums = vbNo

Callout = "<Figure " & thisChapter & ".XXXX about here>"

findThis = "Fig. "

orThis = "Figure "

orMaybeThis = "Figures "

orEvenThis = "Figs "

orHowAbout = "Table "

chapNum = ""

If chapNumsExist = vbYes Then chapNum = thisChapter & "."

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

i = 0

Selection.HomeKey Unit:=wdStory

Do

i = i + 1

figNum = Trim(Str(i))

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = findThis & chapNum & figNum & "[!0-9]"

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

gotOne = Selection.Find.Found

If gotOne = False Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = orThis & chapNum & figNum & "[!0-9]"

.Forward = True

.MatchWildcards = True

.Execute

End With

gotOne = Selection.Find.Found

End If

If gotOne = False Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = orMaybeThis & chapNum & figNum & "[!0-9]"

.Forward = True

.MatchWildcards = True

.Execute

End With

gotOne = Selection.Find.Found

End If

If gotOne = False Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = orEvenThis & chapNum & figNum & "[!0-9]"

.Forward = True

.MatchWildcards = True

.Execute

End With

gotOne = Selection.Find.Found

End If

If gotOne = False Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = orHowAbout & chapNum & figNum & "[!0-9]"

.Forward = True

.MatchWildcards = True

.Execute

End With

gotOne = Selection.Find.Found

End If

If gotOne = False Then

myResponse = MsgBox(findThis & figNum & " missing!" \_

& vbCr & vbCr & "Continue?", vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

Else

If addChapNums = vbYes Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " "

.Forward = True

.MatchWildcards = False

.Execute

End With

Selection.TypeText " " & thisChapter & "."

End If

' Or make the following line Selection.MoveDown

' if you prefer the callout AFTER the paragraph

Selection.MoveUp Unit:=wdParagraph, Count:=1

myTypeThis = Replace(Callout, "XXXX", figNum)

Selection.TypeText myTypeThis & vbCr

Selection.MoveUp Unit:=wdParagraph, Count:=1, Extend:=wdExtend

Selection.Style = ActiveDocument.Styles(wdStyleNormal)

' Selection.Range.HighlightColorIndex = wdYellow

Selection.MoveRight Unit:=wdCharacter, Count:=1

End If

Loop Until 0

ActiveDocument.TrackRevisions = nowTrack

Selection.Find.MatchWildcards = False

End Sub

**Sub TableCallouts()**

**'** Paul Beverley - Version 14.12.10

' Table callout inserter

thisChapter = InputBox("Chapter number?", "Table callouts")

chapNumsExist = MsgBox("Existing chapter numbers?", vbQuestion + vbYesNo)

addChapNums = MsgBox("Add chapter numbers?", vbQuestion + vbYesNo)

' chapNumsExist = vbYes

' addChapNums = vbNo

Callout = "<Table " & thisChapter & ".XXXX about here>"

findThis = "Table "

orThis = "Tables "

If chapNumsExist = vbYes Then

chapNum = thisChapter & "."

Else

chapNum = ""

End If

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

i = 0

Selection.HomeKey Unit:=wdStory

Do

i = i + 1

tableNum = Trim(Str(i))

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = findThis & chapNum & tableNum & "[!0-9]"

.Forward = True

.MatchWildcards = True

.Execute

End With

gotOne = Selection.Find.Found

If gotOne = False Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = orThis & chapNum & tableNum & "[!0-9]"

.Forward = True

.MatchWildcards = True

.Execute

End With

gotOne = Selection.Find.Found

End If

If gotOne = False Then

myResponse = MsgBox(findThis & tableNum & " missing!" \_

& vbCrLf & vbCrLf & "Continue?", vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

Else

If addChapNums = vbYes Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " "

.Forward = True

.MatchWildcards = False

.Execute

End With

Selection.TypeText Text:=" " & thisChapter & "."

End If

Selection.MoveUp Unit:=wdParagraph, Count:=1

typeThis = Replace(Callout, "XXXX", tableNum)

Selection.TypeText Text:=typeThis & vbCrLf

Selection.MoveUp Unit:=wdParagraph, Count:=1, Extend:=wdExtend

Selection.Style = ActiveDocument.Styles(wdStyleNormal)

Selection.Range.HighlightColorIndex = wdYellow

Selection.MoveRight Unit:=wdCharacter, Count:=1

End If

Loop Until 0

ActiveDocument.TrackRevisions = nowTrack

Selection.Find.MatchWildcards = False

End Sub

**Sub FigStrip()**

**'** Paul Beverley - Version 25.05.23

' Strips out all figures and leave a callout

myFormat = "<xxx about here>"

' myFig = "[^13^n^m^l]Fig"

' myFig = "\<Cap\>Fig"

myFig = "^13Fig."

captionWithText = True

captionWithFigs = True

' If Word throws up and error 4605 about pasting

' increase the delay value to, say, 1000

myDelay = 500

myFind = Replace(myFig, "\<", "<")

myFind = Replace(myFind, "\>", ">")

myResponse = MsgBox("Searching for: " & ChrW(8220) & myFind & \_

ChrW(8221), vbQuestion + vbYesNoCancel, "Figure Stripper")

If myResponse <> vbYes Then Exit Sub

ActiveDocument.TrackRevisions = False

Set thisDoc = ActiveDocument

Documents.Add

Set figDoc = ActiveDocument

thisDoc.Activate

oldFind = Selection.Find.Text

Selection.HomeKey Unit:=wdStory

thisMany = 0

Do

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFig

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

gotOne = Selection.Find.Found

Selection.MoveDown Unit:=wdParagraph, count:=1

myEnd = Selection.Start

Selection.MoveUp Unit:=wdParagraph, count:=1

' Cursor now at start of caption line

captionStart = Selection.Start

Do

Selection.MoveRight Unit:=wdCharacter, count:=1

DoEvents

Loop Until Selection = " " Or Selection = vbTab

Do

Selection.MoveRight Unit:=wdCharacter, count:=1

DoEvents

Loop Until Selection = " " Or Selection = vbTab

Selection.Start = captionStart

figTitle = Selection

' Now start to look for one or more figures

figEnd = captionStart

If gotOne Then

Do

Selection.MoveUp Unit:=wdLine, count:=1

Selection.Expand wdParagraph

DoEvents

Loop Until Selection.Words.count > 2

Selection.Collapse wdCollapseEnd

Selection.End = myEnd

' Is this a figure i've found?[[[

figsBefore = ActiveDocument.InlineShapes.count \_

+ ActiveDocument.Shapes.count

Selection.Cut

figsAfter = ActiveDocument.InlineShapes.count \_

+ ActiveDocument.Shapes.count

If figsAfter = figsBefore Then

' If not, put it back

Selection.Paste

Selection.MoveLeft Unit:=wdWord, count:=2

Else

Selection.InsertAfter Replace(myFormat, "xxx", figTitle) & vbCr

Selection.ParagraphFormat.Alignment = wdAlignParagraphLeft

Selection.HomeKey Unit:=wdLine

figDoc.Activate

For i = 1 To myDelay

DoEvents

Next i

Selection.Paste

' But mark the caption ready to copy it back

Selection.MoveUp Unit:=wdParagraph, count:=1, Extend:=wdExtend

' Either

If captionWithFigs = True Then

Selection.Copy

Else

Selection.Cut

Selection.TypeText figTitle

End If

Selection.Start = Selection.End

Selection.TypeParagraph

Selection.TypeParagraph

thisMany = thisMany + 1

thisDoc.Activate

' If caption wanted in text, paste it back in

If captionWithText = True Then

Selection.MoveDown Unit:=wdParagraph, count:=1

Selection.Paste

End If

End If

End If

DoEvents

Loop Until gotOne = False

figDoc.Activate

Selection.TypeText Str(thisMany) & " figures extracted" & vbCr

Selection.Find.Text = oldFind

Selection.Find.MatchWildcards = False

Beep

End Sub

**Sub TableEdit()**

**'** Paul Beverley - Version 03.08.24

' Edits items in the cells of some or all tables

stripEnds = True

stripThese = "^p^t., "

addEmDash = True

doTrack = True

doHighlight = False

myColour = wdGray25

myTrack = ActiveDocument.TrackRevisions

If doTrack = False Then ActiveDocument.TrackRevisions = False

stripThese = Replace(stripThese, "^p", vbCr)

stripThese = Replace(stripThese, "^t", vbTab)

If Selection.start = Selection.End Then

Set workRange = ActiveDocument.Tables

doTheLot = True

Else

Set workRange = Selection.Tables

End If

Set rng = ActiveDocument.Content

For Each myTable In workRange

For Each myCell In myTable.Range.Cells

myText = Left(myCell.Range, Len(myCell.Range) - 2)

newText = ""

' an em dash

If addEmDash = True And Trim(myText) = "" Then newText = ChrW(8212): myText = ""

If Trim(myText) = ChrW(8211) Then newText = ChrW(8212): myText = ""

If Trim(myText) = "-" Then newText = ChrW(8212): myText = ""

' en dash with figures to minus sign

If Left(myText, 1) = ChrW(8211) And Len(myText) > 1 Then

newText = Replace(myText, ChrW(8211), ChrW(8722)): myText = ""

End If

' hyphen to minus sign

If Left(myText, 1) = "-" And Len(myText) > 2 Then

newText = Replace(myText, "-", ChrW(8722))

End If

' space-hyphen to minus sign

If Left(myText, 2) = " -" Then

newText = Replace(myText, " -", ChrW(8722))

End If

If newText > "" Then

myCell.Range = newText

If doHighlight = True Then

ActiveDocument.TrackRevisions = False

myCell.Range.HighlightColorIndex = myColour

ActiveDocument.TrackRevisions = myTrack

End If

End If

' Now strip any trailing tabs or CRs

If stripEnds = True Then

Set rng = myCell.Range

rng.MoveEnd , -1

rng.start = rng.End - 1

' Strip odd unwanted characters

Do While InStr(stripThese, rng.Text) > 0

rng.Delete

DoEvents

rng.MoveStart , -1

Loop

End If

If myCell.Range.Characters(1) = "." Then myCell.Range.InsertBefore "0"

Next myCell

Next myTable

Beep

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub TableEmDasher()**

' Paul Beverley - Version 05.03.13

' Empty cells and hyphen/en dash to em dash

dashIfEmpty = True

trackIt = False

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

For Each myTable In ActiveDocument.Tables

For Each myCell In myTable.Range.Cells

myText = Trim(Left(myCell.Range, Len(myCell.Range) - 2))

If myText = "-" Or myText = ChrW(8211) Or (dashIfEmpty = True \_

And myText = "") Then

myCell.Range = ChrW(8212)

End If

Next myCell

Next myTable

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub CellsAddChar()**

**'** Paul Beverley - Version 20.10.11

' Check that there is a full point ending each cell

myChar = "."

Selection.Font.Shadow = True

For Each myCell In Selection.Tables(1).Range.Cells

Set rng = myCell.Range

myText = rng

If rng.Font.Shadow = True And InStr(Right(myText, 3), myChar) = 0 Then

rng.InsertAfter myChar

End If

Next myCell

Selection.Tables(1).Range.Font.Shadow = False

End Sub

**Sub TableCellsInitialCaps()**

' Paul Beverley - Version 01.09.16

' Applies an initial capital to every cell in the selected range

trackIt = False

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

For Each myTable In Selection.range.Tables

For Each myCell In myTable.range.Cells

myText = Trim(Left(myCell.range, Len(myCell.range) - 2))

firstChar = Left(myText, 1)

newChar = UCase(firstChar)

If newChar <> firstChar Then

myCell.range = newChar & Mid(myText, 2)

End If

Next myCell

Next myTable

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub TableBordersToggle()**

' Paul Beverley - Version 14.08.19

' Switches table borders and rules on and off

If Selection.Borders(wdBorderTop).LineStyle = wdLineStyleNone Then

With Selection.Borders(wdBorderTop)

.LineStyle = Options.DefaultBorderLineStyle

.LineWidth = Options.DefaultBorderLineWidth

.Color = Options.DefaultBorderColor

End With

With Selection.Borders(wdBorderLeft)

.LineStyle = Options.DefaultBorderLineStyle

.LineWidth = Options.DefaultBorderLineWidth

.Color = Options.DefaultBorderColor

End With

With Selection.Borders(wdBorderBottom)

.LineStyle = Options.DefaultBorderLineStyle

.LineWidth = Options.DefaultBorderLineWidth

.Color = Options.DefaultBorderColor

End With

With Selection.Borders(wdBorderRight)

.LineStyle = Options.DefaultBorderLineStyle

.LineWidth = Options.DefaultBorderLineWidth

.Color = Options.DefaultBorderColor

End With

With Selection.Borders(wdBorderHorizontal)

.LineStyle = Options.DefaultBorderLineStyle

.LineWidth = Options.DefaultBorderLineWidth

.Color = Options.DefaultBorderColor

End With

With Selection.Borders(wdBorderVertical)

.LineStyle = Options.DefaultBorderLineStyle

.LineWidth = Options.DefaultBorderLineWidth

.Color = Options.DefaultBorderColor

End With

Else

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(wdBorderVertical).LineStyle = wdLineStyleNone

Selection.Borders(wdBorderDiagonalDown).LineStyle = wdLineStyleNone

Selection.Borders(wdBorderDiagonalUp).LineStyle = wdLineStyleNone

End If

End Sub

**Sub TextBoxFrameCut()**

**'** Paul Beverley - Version 01.06.10

' Remove textboxes and frames

' Original version by richardwalshe@prufrock.co.uk

Dim sh As Shape

Dim fr As Frame

For Each sh In ActiveDocument.Shapes

If sh.Type = msoGroup Then sh.Ungroup

Next sh

For Each sh In ActiveDocument.Shapes

If sh.TextFrame.HasText Then

' Leaves images intact

sh.TextFrame.TextRange.Copy

' Finds where the anchor for the textbox is

' N.B. This is not necessarily where the textbox

' has ended up

sh.Anchor.Paragraphs(1).Range.Select

Selection.Collapse

sh.Delete

' Marks material so correctness of position

' can be checked

Selection.TypeText Text:="<TBX>"

Selection.Paste

Selection.TypeText Text:="</TBX>"

End If

Next sh

For Each fr In ActiveDocument.Frames

' Removes frames with similar tagging for later checking

fr.Select

Selection.Collapse

Selection.TypeText Text:="<FRM>"

fr.Select

Selection.Collapse Direction:=wdCollapseEnd

Selection.TypeText Text:="</FRM>"

' This is the bit that actually removes the frames

fr.Select

fr.Delete

Next fr

End Sub

**Sub ListAllStyles()**

**'** Paul Beverley - Version 22.04.24

' Creates a list of all styles name

myStyles = ""

myStyles = "Normal,Default,"

displayParaWords = True

numWords = 6

deleteTableBorders = False

Set myFile = ActiveDocument

myAllText = ""

For myArea = 1 To 3

doThisArea = False

' Main text area

If myArea = 1 Then

If Selection.Start = Selection.End Then

myResponse = MsgBox("Scan the whole document?!", \_

vbQuestion + vbYesNo, "StyleLister")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

Else

Set rng = Selection.Range.Duplicate

End If

doThisArea = True

End If

' Footnotes, if any

If ActiveDocument.Footnotes.Count > 0 And myArea = 2 Then

doThisArea = True

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

StatusBar = "Scanning footnotes"

End If

' Endnotes, if any

If ActiveDocument.Endnotes.Count > 0 And myArea = 3 Then

doThisArea = True

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

StatusBar = "Scanning endnotes"

End If

If doThisArea = True Then

For Each myPar In rng.Paragraphs

Set myPar = myPar.Range

myPar.Select

If myPar.Hyperlinks.Count = 0 Then

thisStyle = myPar.Style

If InStr(myStyles, thisStyle) = 0 Then

myStyles = myStyles & thisStyle & ","

Set testChar = myPar.Characters(1)

pageNum = testChar.Information(wdActiveEndAdjustedPageNumber)

myLineText = thisStyle & vbTab & "p." & Trim(Str(pageNum))

If displayParaWords Then

myStart = testChar.Start

If myPar.Words.Count > numWords Then

myEnd = myPar.Words(numWords).End

Else

myEnd = myPar.End - 1

End If

myLen = myEnd - myStart

myWords = Left(myPar.Text, myLen)

' Get rid of footnote markers

myWords = Replace(myWords, Chr(2), "")

myLineText = myLineText & vbTab & Trim(myWords)

Debug.Print myLineText

myAllText = myAllText & myLineText & vbCr

End If

End If

DoEvents

End If

Next myPar

End If

Next myArea

' Display the results in a table

Documents.Add

Selection.TypeText Text:=myAllText

Set rng = ActiveDocument.Content

rng.Sort

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Style list"

startTable = Selection.End + 1

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

rng.Start = startTable

rng.End = ActiveDocument.Range.End

rng.ConvertToTable Separator:=wdSeparateByTabs

Set tb = ActiveDocument.Tables(1)

tb.AutoFitBehavior (wdAutoFitContent)

If deleteTableBorders = True Then

tb.Borders(wdBorderTop).LineStyle = wdLineStyleNone

tb.Borders(wdBorderLeft).LineStyle = wdLineStyleNone

tb.Borders(wdBorderBottom).LineStyle = wdLineStyleNone

tb.Borders(wdBorderRight).LineStyle = wdLineStyleNone

tb.Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

tb.Borders(wdBorderVertical).LineStyle = wdLineStyleNone

End If

End Sub

**Sub SetTextBoxStyle()**

**'** Paul Beverley - Version 01.06.10

' Apply style to all textboxes

Dim myStoryRange As Range

Set myStoryRange = ActiveDocument.StoryRanges(wdTextFrameStory)

myStoryRange.Style = "List Bullet"

While Not (myStoryRange.NextStoryRange Is Nothing)

Set myStoryRange = myStoryRange.NextStoryRange

myStoryRange.Style = "List Bullet"

Wend

End Sub

**Sub BoxTextIntoBody()**

' Paul Beverley - Version 08.10.15

' Copy text out of textboxes, then delete the boxes

placeTextAfterParagraph = True

addHighlight = True

myColour = wdYellow

'beforeText = ""

'afterText = ""

beforeText = "<Box>"

afterText = "</Box>"

For Each shp In ActiveDocument.Shapes

If shp.Type = msoTextBox Then

shp.TextFrame.TextRange.Select

Selection.Copy

' set the range to insert the text

Set rng = shp.Anchor.Paragraphs(1).range

' paste the text after the range object

If placeTextAfterParagraph = True Then

rng.Collapse wdCollapseEnd

Else

rng.Collapse wdCollapseStart

End If

If Len(beforeText) > 0 Then rng.InsertBefore Text:=beforeText

rng.Collapse wdCollapseEnd

startText = rng.Start

rng.Paste

rng.Start = startText

If addHighlight = True Then

rng.HighlightColorIndex = myColour

End If

rng.Collapse wdCollapseEnd

If Len(afterText) > 0 Then rng.InsertBefore Text:=afterText

End If

Next shp

numShapes = ActiveDocument.Shapes.Count

For i = numShapes To 1 Step -1

Set shp = ActiveDocument.Shapes(i)

If shp.Type = msoTextBox Then

shp.Delete

End If

Next i

End Sub

**Sub FootnoteEndnoteFiddle()**

**'** Paul Beverley - Version 11.08.24

' Tidies up start/end of each footnote or endnote

addFullStop = False

noteNumberNotSuperscript = True

Set rngWas = Selection.Range.Duplicate

For myRun = 1 To 2

numNotes = 0

If myRun = 1 Then

numNotes = ActiveDocument.Footnotes.Count

Else

myRun = 2

End If

If myRun = 2 Then numNotes = ActiveDocument.Endnotes.Count

For i = 1 To numNotes

If myRun = 1 Then

Set rng = ActiveDocument.Footnotes(i).Range

Else

Set rng = ActiveDocument.Endnotes(i).Range

End If

rng.Select

' Remove leading space

Selection.Collapse wdCollapseStart

Selection.HomeKey wdLine

If noteNumberNotSuperscript = True Then

Selection.MoveEnd , 1

Selection.Range.Font.Superscript = False

Selection.Collapse wdCollapseStart

End If

Selection.MoveStart , 1

Selection.End = Selection.start + 1

If Selection.Text = " " Then Selection.Delete

' Remove trailing 'things'

rng.Collapse wdCollapseEnd

rng.MoveStart , -1

Do While InStr(" " & vbTab & vbCr, rng) > 0

rng.Delete

rng.MoveStart , -1

DoEvents

Loop

' Add full stop if none

If addFullStop Then

rng.Collapse wdCollapseEnd

rng.MoveStart , -1

If InStr(ChrW(8217) & ChrW(8221), rng) > 0 Then

rng.MoveStart , -1

rng.MoveEnd , -1

End If

If UCase(rng) <> LCase(rng) Then rng.InsertAfter Text:="."

DoEvents

End If

Next i

Next myRun

rngWas.Select

End Sub

**Sub FootnoteNumberNotItalic()**

**'** Paul Beverley - Version 01.06.10

' Makes changes to all footnotes

For i = 1 To ActiveDocument.Footnotes.Count

Set rng = ActiveDocument.Footnotes(i).Range

rng.End = rng.Start

rng.Start = rng.Start - 3

rng.Font.Italic = False

Next

End Sub

**Sub NoteDeleteDblSpace()**

**'** Paul Beverley - Version 01.06.10

' Deletes double spaces from endnotes

For i = 1 To ActiveDocument.Endnotes.Count

Set rng = ActiveDocument.Endnotes(i).Range

rng.Start = rng.Start - 1

rng.End = rng.Start + 1

mychar = rng

myASC = 0

' Check the very first character of the endnote

If Len(mychar) > 0 And i > 1 Then myASC = Asc(mychar)

' If it’s a newline or a paragraph mark, delete it

If myASC = 13 Or myASC = 11 Then rng.Delete

' Then do the same right through the endnotes

' i.e. delete any newlines, paras or spaces that

' occur at the beginning of an endnote, before

' the actual text of the note starts

Set rng = ActiveDocument.Endnotes(i).Range

rng.Start = rng.End - 1

KeepGoing = True

Do

mychar = rng

If Len(mychar) > 0 Then

myASC = Asc(mychar)

Else

myASC = 0

End If

If myASC = 13 Or myASC = 11 Or myASC = 32 Then

rng.Delete

rng.Start = rng.Start - 1

Else

KeepGoing = False

End If

Loop Until KeepGoing = False

Next

End Sub

**Sub EndNoteFiddleSuperscript()**

**'** Paul Beverley - Version 25.07.11

' Makes changes to superscript on all endnotes

For i = 1 To ActiveDocument.Endnotes.Count

Set rng = ActiveDocument.Endnotes(i).Range

rng.Start = rng.Start - 2

rng.End = rng.Start + 1

rng.Font.Superscript = True

Next

End Sub

**Sub FootnoteFiddleStartSpace()**

' Paul Beverley - Version 06.01.18

' Removes initial space from each footnote

For i = 1 To ActiveDocument.Footnotes.Count

Set rng = ActiveDocument.Footnotes(i).range

rng.End = rng.Start + 1

If rng.Text = " " Then rng.Delete

Next

End Sub

**Sub DeleteAllEndnotes()**

**'** Paul Beverley - Version 19.09.18

' Delete all endnotes

numberENs = ActiveDocument.Endnotes.Count

For i = numberENs To 1 Step -1

ActiveDocument.Endnotes(i).Delete

StatusBar = " " & i

Next i

End Sub

**Sub DeleteAllFootnotes()**

**'** Paul Beverley - Version 19.09.18

' Delete all footnotes

numberFNs = ActiveDocument.Footnotes.Count

For i = numberFNs To 1 Step -1

ActiveDocument.Footnotes(i).Delete

StatusBar = " " & i

Next i

End Sub

**Sub NotesUnembed()**

' Paul Beverley - Version 19.02.16

' Unembeds footnotes or endnotes

doHighlight = False

ActiveDocument.Footnotes.Convert

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

For Each myNote In ActiveDocument.Endnotes

If Len(myNote.Range.Text) > 1 Then myNote.Range.Copy

myNote.Reference.InsertBefore "zc" & myNote.Index & "cz"

Selection.EndKey Unit:=wdStory

Selection.TypeText myNote.Index & ". "

Selection.Paste

Selection.TypeText "Blank"

Selection.MoveStart , -5

Selection.Cut

Selection.TypeText vbCr

DoEvents

Next myNote

For Each myNote In ActiveDocument.Endnotes

myNote.Delete

Next myNote

' Remember the existing highlight colour

oldColour = Options.DefaultHighlightColorIndex

' Select preferred colour

Options.DefaultHighlightColorIndex = wdTurquoise

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zc([0-9]{1,})cz"

.Replacement.Text = "\1"

.Replacement.Font.Superscript = True

If doHighlight = True Then .Replacement.Highlight = True

.Wrap = wdFindContinue

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Restore original colour

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub NotesEmbed()**

**'** Paul Beverley - Version 12.03.24

' Embed footnotes or endnotes

deleteNoteNumber = False

deleteNoteNumber = True

Selection.HomeKey Unit:=wdLine

If Selection <> "1" And Asc(Selection) <> 9 Then

myResponse = MsgBox("Is this the first line of the notes?", \_

vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

End If

With Selection.Find

.Text = "^p^p"

.Replacement.Text = "^p"

.Forward = True

.Wrap = wdFindStop

.Format = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

'Put a bookmark at the beginning of the footnotes

ActiveDocument.Bookmarks.Add Name:="PBnoteStart"

' Make sure there's a CR at the end of the file

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Do

Selection.GoTo What:=wdGoToBookmark, Name:="PBnoteStart"

' Read the note number

myStart = Selection.Start

Selection.End = Selection.Start + 5

myText = Selection.Text

spPos = InStr(Selection, " ")

If spPos > 0 Then myText = Left(myText, spPos - 1)

tbPos = InStr(Selection, vbTab)

If tbPos > 0 Then myText = Left(myText, tbPos - 1)

myNote = Val(myText)

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

' Give up if you've reached the end

If myNote = 0 Then Exit Do

' Find the following note

myFind = "^p" & Trim(Str(myNote + 1))

With Selection.Find

.Text = myFind

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindStop

.Format = False

.MatchWildcards = False

.Execute

End With

If Selection.Find.Found = True Then

Selection.Collapse wdCollapseStart

Selection.MoveRight , 1

' Select the footnote

Selection.Start = myStart

Else

Selection.MoveRight , 2

Selection.End = ActiveDocument.Content.End - 1

If Right(Selection, 1) = vbCr Then Selection.MoveEnd , -1

End If

myNextNote = Selection

' Find a tab or, if not, the first space, i.e. after the note number

spacePlace = InStr(myPara, Chr(9))

If spacePlace = 0 Then spacePlace = InStr(myNextNote, " ")

Selection.MoveEnd , -1

Selection.MoveStart , spacePlace

Selection.Copy

Selection.Start = myStart

' Delete the used footnote

Selection.Delete

If myNote = 1 Then

Selection.HomeKey Unit:=wdStory

Else

' Go back to the previous citation

Selection.GoTo What:=wdGoToBookmark, Name:="PBlastNote"

End If

' Find the next citation (superscript number)

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Trim(Str(myNote))

.Font.Superscript = True

.Replacement.Font.Superscript = False

.Replacement.Text = ""

.Wrap = wdFindContinue

.Forward = True

.Execute

End With

' Delete the superscript number and add a footnote

If deleteNoteNumber = True Then Selection.Delete

' Bookmark the place

ActiveDocument.Bookmarks.Add Name:="PBlastNote"

' Add a footnote and paste in the text of the footnote

With ActiveDocument.Range(Start:=ActiveDocument.Content.Start, End:= \_

ActiveDocument.Content.End)

.Footnotes.Add Range:=Selection.Range, Reference:=""

End With

Selection.Paste

DoEvents

Loop Until myNote = 0

' Tidy up and go to the end

ActiveDocument.Bookmarks("PBnoteStart").Delete

ActiveDocument.Bookmarks("PBlastNote").Delete

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

End With

Selection.EndKey Unit:=wdStory

myResponse = MsgBox("Convert to endnotes, rather than footnotes?", \_

vbQuestion + vbYesNo)

If myResponse = vbYes Then ActiveDocument.Footnotes.Convert

End Sub

**Sub NotesUnembedBySections()**

**'** Paul Beverley - Version 28.12.11

' Unembed endnotes that are numbered in sections

doHighlight = True

' Remember the existing highlight colour

oldColour = Options.DefaultHighlightColorIndex

' Select preferred colour

Options.DefaultHighlightColorIndex = wdTurquoise

myNoteNum = 0

Set rng = ActiveDocument.Content

For I = 1 To ActiveDocument.Sections.Count

mySectionNotes = ""

Set secRng = ActiveDocument.Sections(I).Range

mySectionStart = secRng.Start

mySectionEnd = secRng.End

mySectionNoteNum = 0

charsAdded = 0

rng.Start = mySectionStart

rng.End = mySectionStart

Do

myNoteNum = myNoteNum + 1

mySectionNoteNum = mySectionNoteNum + 1

' Copy the note text

If myNoteNum <= ActiveDocument.Endnotes.Count Then

ActiveDocument.Endnotes(myNoteNum).Range.Copy

End If

' Find next note citation

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^2"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

foundOne = rng.Find.Found

absNotePos = rng.Start

absNotePos2 = rng.End

If foundOne = True Then

' If still within current section...

If rng.Start < mySectionEnd Then

' We've found a note in the section

myCitation = "zct" & mySectionNoteNum & "zct"

charsAdded = charsAdded + Len(myCitation)

rng.Start = absNotePos

' Add citation number as text

rng.InsertBefore myCitation

rng.Start = rng.End

startAgainHere = rng.Start

Set secRng = ActiveDocument.Sections(I).Range

mySectionEnd = secRng.End

secRng.Start = secRng.End - 1

secRng.End = secRng.Start

' Paste the note text

secRng.PasteAndFormat (wdFormatSurroundingFormattingWithEmphasis)

' or use just: secRng.Paste

' Add the note number (within this section)

secRng.InsertBefore vbCrLf & "znt" & mySectionNoteNum & "znt"

Else

myNoteNum = myNoteNum - 1

End If

End If

StatusBar = "Adding note number: " & Str(myNoteNum)

Loop Until foundOne = False Or absNotePos > mySectionEnd

Next I

' Convert citation numbers to superscript

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zct([0-9]{1,})zct"

.Replacement.Text = "\1"

.Replacement.Font.Superscript = True

If doHighlight = True Then .Replacement.Highlight = True

.Wrap = wdFindContinue

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Convert note numbers to superscript

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "znt([0-9]{1,})znt"

.Replacement.Text = "\1"

.Replacement.Font.Superscript = True

.Wrap = wdFindContinue

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Delete original notes

For Each myNote In ActiveDocument.Endnotes

myNote.Delete

myNoteNum = myNoteNum - 1

StatusBar = "Deleting note number: " & Str(myNoteNum)

Next myNote

StatusBar = "Finished! But please wait for it to reformat the document."

Selection.EndKey Unit:=wdStory

Beep

End Sub

**Sub RenumberSuperscript()**

' Paul Beverley - Version 28.11.20

' Renumbers all superscript numbers

' Check if user wants to work on whole file of selection

If Selection.End = Selection.Start Then

myResponse = MsgBox("Do this to the WHOLE file?", \_

vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

Set rng = ActiveDocument.Content

Else

Set rng = Selection.Range.Duplicate

End If

i = 1

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "[0-9]{1,}"

.Font.Superscript = True

.Replacement.Text = ""

.Wrap = wdFindStop

.Execute

End With

Do While rng.Start < Selection.End

rng.Text = Trim(Str(i))

i = i + 1

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

rng.Select

End Sub

**Sub RenumberNotes()**

' Paul Beverley - Version 02.12.20

' Renumbers all note numbers in selected text

' Check if user wants to work on whole file of selection

If Selection.End = Selection.Start Then

myResponse = MsgBox("Do this to the WHOLE file?", \_

vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

Set rngAll = ActiveDocument.Content

Else

Set rngAll = Selection.Range.Duplicate

End If

Set rng = rngAll.Duplicate

fstWord = rng.Words(1)

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

rng.Select

rng.End = rngAll.End

rng.Start = rng.Start - 1

If Val(fstWord) <> 1 Then

myResponse = MsgBox("Is this the first line of the notes?", \_

vbQuestion + vbYesNo)

If myResponse <> vbYes Then Exit Sub

End If

i = 1

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "^13[0-9]{1,}"

.Replacement.Text = ""

.Wrap = False

.Execute

End With

rng.Select

Do While rng.Find.Found = True And rng.End < rngAll.End

rng.MoveStart wdCharacter, 1

rng.Delete

rng.InsertAfter Text:=Trim(Str(i))

i = i + 1

DoEvents

rng.End = rngAll.End

rng.Find.Execute

Loop

Beep

End Sub

**Sub NotesCopyToInline()**

' Paul Beverley - Version 18.01.23

' Copies notes into inline notes in brackets

' Guillemets

' myOpen = ChrW(171)

' myClose = ChrW(187)

' myOpen = "["

' myClose = "]"

' myOpen = "{"

' myClose = "}"

myOpen = "<"

myClose = ">"

' If Word throws up and error 4605 about pasting

' increase the delay value to, say, 1000

myDelay = 500

If ActiveDocument.Endnotes.Count > 0 Then

For Each myNote In ActiveDocument.Endnotes

myNote.Range.Copy

myEnd = myNote.Footnotes(1).Reference.End

Set rng = ActiveDocument.Range(myEnd, myEnd)

rng.InsertAfter Text:=myClose

rng.Collapse wdCollapseStart

For i = 1 To myDelay

DoEvents

Next i

rng.Paste

rng.Collapse wdCollapseStart

rng.InsertAfter Text:=myOpen

Next myNote

End If

If ActiveDocument.Footnotes.Count > 0 Then

For Each myNote In ActiveDocument.Footnotes

myNote.Range.Copy

myEnd = myNote.Reference.End

Set rng = ActiveDocument.Range(myEnd, myEnd)

rng.InsertAfter Text:=myClose

rng.Collapse wdCollapseStart

For i = 1 To myDelay

DoEvents

Next i

rng.Paste

rng.Collapse wdCollapseStart

rng.InsertAfter Text:=myOpen

Next myNote

End If

End Sub

**Sub NotesReEmbedByNumber()**

' Paul Beverley - Version 13.01.21

' Re-embed footnotes or endnotes by number

startNum = 316

endNum = 351

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Selection.TypeText vbCr

Selection.MoveLeft , 1

'Put a bookmark at the beginning of the footnotes

ActiveDocument.Bookmarks.Add Name:="PBnotesStart"

Selection.EndKey Unit:=wdStory

Selection.TypeText Trim(Str(endNum + 1)) & vbCr

For i = startNum To endNum

Selection.GoTo what:=wdGoToBookmark, Name:="PBnotesStart"

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p" & Trim(Str(i))

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Selection.Collapse wdCollapseEnd

Selection.MoveEnd , 1

Do While InStr(". " & vbTab, Left(Selection.Text, 1)) > 0

Selection.MoveStart , -1

DoEvents

Loop

startNote = Selection.End

findThis = "^p" & Trim(Str(i + 1))

With Selection.Find

.Text = findThis

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Selection.Collapse wdCollapseStart

Selection.Start = startNote

Selection.Copy

' Go and look for footnote marker

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Trim(Str(i))

.Font.Superscript = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Replacement.Font.Superscript = False

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

' Add a footnote and paste in the text of the footnote

With ActiveDocument.Range(Start:=ActiveDocument.Content.Start, End:= \_

ActiveDocument.Content.End)

' With .FootnoteOptions

' .Location = wdBottomOfPage

' .NumberingRule = wdRestartContinuous

' .StartingNumber = 1

' .NumberStyle = wdNoteNumberStyleArabic

' End With

.Footnotes.Add Range:=Selection.Range, Reference:=""

End With

Selection.Paste

Next i

Beep

MsgBox "Finished!"

End Sub

**Sub NotesInlineToEmbed()**

' Paul Beverley - Version 24.11.20

' Copies in-line notes into embedded notes

' Guillemets

' myOpen = ChrW(171)

' myClose = ChrW(187)

' myOpen = "["

' myClose = "]"

' myOpen = "{"

' myClose = "}"

myOpen = "<"

myClose = ">"

For i = ActiveDocument.Endnotes.Count To 1 Step -1

ActiveDocument.Endnotes(i).Delete

StatusBar = " " & i

Next i

For i = ActiveDocument.Footnotes.Count To 1 Step -1

ActiveDocument.Footnotes(i).Delete

StatusBar = " " & i

Next i

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myOpen

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found

rng.End = ActiveDocument.Content.End

rng.Select

myEnd = rng.Start + InStr(rng.Text, myClose)

rng.End = myEnd - 1

rng.Start = rng.Start + 1

rng.Select

rng.Copy

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.Footnotes.Add Range:=Selection.Range, Reference:=""

Selection.Paste

Selection.Range.Font.ColorIndex = wdBlue

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Correct the line spacing

For i = 1 To ActiveDocument.Footnotes.Count

mySize = ActiveDocument.Footnotes(i).Range.Font.Size

ActiveDocument.Footnotes(i).Range.ParagraphFormat.LineSpacing = mySize

Next i

' Now delete inline notes

If Asc(myOpen) < 126 Then myOpen = "\" & myOpen

If Asc(myClose) < 126 Then myClose = "\" & myClose

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myOpen & "\*" & myClose

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

End Sub

**Sub ListRenumber()**

' Paul Beverley - Version 30.11.19

' Makes all following numbered items in a list consecutive

myPrompt = "Max paragraphs between items?" & vbCr & vbCr \_

& "(0 = single paragraphs only)"

myInput = InputBox(myPrompt, "ListRenumber")

If myInput = "" Then

Beep

Exit Sub

End If

maxBlank = Val(myInput)

Selection.Expand wdParagraph

myNum = Val(Selection)

Do

myNum = myNum + 1

cntDown = maxBlank + 2

Do

cntDown = cntDown - 1

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

If Len(Selection) < 2 Then

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

End If

newNum = Val(Selection)

If cntDown < 1 Then

Beep

Exit Sub

End If

Loop Until newNum > 0

If newNum <> myNum Then

myLen = 1

If newNum > 9 Then myLen = 2

If newNum > 99 Then myLen = 3

Set rng = Selection.range.Duplicate

rng.End = rng.Start + myLen

rng.Text = Trim(Str(myNum))

End If

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

DoEvents

Loop Until 0 Or Selection.End = ActiveDocument.Content.End

End Sub

**Sub FootnoteAdd()**

' Paul Beverley - Version 27.08.18

' Creates a new footnote (in a given style and/or sq. brackets)

addSquares = True

changeStyle = False

If addSquares = True Then

Selection.TypeText Text:="[]"

Selection.MoveStart , -2

Selection.Font.Superscript = True

Selection.Collapse wdCollapseStart

Selection.MoveRight , 1

End If

If changeStyle = True Then

' myNoteStyle = "Heading 2"

myNoteStyle = wdStyleHeading2

Selection.Style = ActiveDocument.Styles(myNoteStyle)

End If

Selection.Footnotes.Add range:=Selection.range

End Sub

**Sub EndnoteAdd()**

' Paul Beverley - Version 30.11.17

' Creates a new endnote (in a given style and/or sq. brackets)

' myEndnoteStyle = "Heading 2"

myEndnoteStyle = wdStyleHeading2

Selection.Endnotes.Add range:=Selection.range

' Selection.Style = ActiveDocument.Styles(wdStyleEndnoteText)

Selection.Style = ActiveDocument.Styles(myEndnoteStyle)

End Sub

**Sub FootnoteAdd()**

' Paul Beverley - Version 27.08.18

' Creates a new footnote (in a given style)

addSquares = True

changeStyle = False

If addSquares = True Then

Selection.TypeText Text:="[]"

Selection.MoveStart , -2

Selection.Font.Superscript = True

Selection.Collapse wdCollapseStart

Selection.MoveRight , 1

End If

If changeStyle = True Then

' myNoteStyle = "Heading 2"

myNoteStyle = wdStyleHeading2

Selection.Style = ActiveDocument.Styles(myNoteStyle)

End If

Selection.Endnotes.Add range:=Selection.range

End Sub

**Sub RemoveNumbersFromHeadings()**

' Paul Beverley - Version 15.10.19

' Removes automatic numbering from headings

' **Based on code provided by Santhosh Matthew Paul**

myTargets = "Heading 1,Heading 2,Heading 3,Heading 4,Heading 5"

For Each para In ActiveDocument.Paragraphs

inAtable = para.range.Information(wdWithInTable)

If inAtable = False Then

head = para.Style

If InStr(myTargets, head) > 0 Then

para.range.ListFormat.RemoveNumbers

para.range.Select

End If

End If

Next para

End Sub

**Sub DeleteAllBookmarks()**

**'** Paul Beverley - Version 18.06.10

' Deletes all bookmarks

ActiveDocument.Bookmarks.ShowHidden = True

numberBMs = ActiveDocument.Bookmarks.Count

If numberBMs > 0 Then

For Each myBM In ActiveDocument.Bookmarks

myBM.Delete

Next

End If

MsgBox ("Bookmarks deleted: " & Str(numberBMs))

End Sub

**Sub DeleteCommentsSelectively()**

' Paul Beverley - Version 10.02.21

' Accepts track changes of specific author/editor

' deleteName = "Microsoft"

deleteName = "Paul Beverley"

For Each cm In ActiveDocument.Comments

If InStr(cm.Author, deleteName) > 0 Then

cm.Scope.Select

cm.Delete

End If

DoEvents

Next cm

Beep

End Sub

**Sub CommentBracketsToBubbles()**

' Paul Beverley - Version 26.05.18

' Copies text in square brackets and into comment bubbles

deleteOriginalComments = True

carryFormatting = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[\*\]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

Set cmt = Selection.Comments.Add(range:=rng)

If carryFormatting = True Then

cmt.range.FormattedText = rng.FormattedText

Else

cmt.range.Text = rng.Text

End If

numChars = cmt.range.Characters.Count

cmt.range.Characters(numChars) = ""

cmt.range.Characters(1) = ""

If deleteOriginalComments Then rng.Cut

rng.Find.Execute

Loop

MsgBox "Comments copied: " & myCount

End Sub

**Sub CommentBubblesToBrackets()**

**'** Paul Beverley - Version 10.08.24

' Copies comments into brackets in running text

addBold = False

highlightComment = True

colourComment = wdColorBlue

highlightRange = True

colourRange = wdYellow

removeComments = True

numCmts = ActiveDocument.Comments.Count

ReDim rStart(numCmts) As Long

ReDim rEnd(numCmts) As Long

For i = 1 To numCmts

Set myCmt = ActiveDocument.Comments(i)

myCmt.Range.Copy

Set cmnt = myCmt.Scope

rStart(i) = cmnt.start

rEnd(i) = cmnt.End

cmnt.start = cmnt.start + Len(cmnt.Text)

myStart = cmnt.start

cmnt.InsertAfter Text:=" []"

cmnt.Collapse wdCollapseEnd

cmnt.MoveEnd , -1

cmnt.Paste

cmnt.start = myStart

If addBold = True Then cmnt.Font.Bold = True

If highlightComment Then

cmnt.MoveStart , 2

cmnt.Font.Color = colourComment

End If

Next i

If removeComments = True Then ActiveDocument.DeleteAllComments

Set rng = ActiveDocument.Content

nudge = 0

For i = 1 To numCmts

rng.start = rStart(i) - nudge

rng.End = rEnd(i) - nudge

rng.HighlightColorIndex = colourRange

nudge = nudge + 1

Next i

End Sub

**Sub CommentsAddIndexOnInitials()**

' Paul Beverley - Version 17.10.19

' Adds (and removes) serial numbers to initials in comments

myInit = "AQ" ' Use "AQ:" in each comment

Set rng = ActiveDocument.StoryRanges(wdCommentsStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myInit & "([0-9]{1,})\:"

.Wrap = wdFindContinue

.Replacement.Text = myInit & ":"

.Forward = True

.MatchWildcards = True

.Execute

End With

If rng.Find.Found = True Then

rng.Find.Execute Replace:=wdReplaceAll

Else

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myInit & ":"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

rng.Text = myInit & Trim(Str(myCount)) & ":"

rng.Start = EndNow + 4

rng.End = rng.Start

rng.Find.Execute

Loop

End If

End Sub

**Sub CommentsDeleteAllNotTagged()**

' Paul Beverley - Version 17.10.19

' Deletes all comments \*not\* starting with a specific tag

myTag = "AQ"

For Each cmt In ActiveDocument.Comments

If InStr(cmt.range.Text, myTag) = 0 Then cmt.Delete

Next cmt

End Sub

**Sub CommentsDeleteSelectively()**

' Paul Beverley - Version 13.05.20

' Deletes all comments that have a specific tag

myDeletionMarker = "\*\*\*"

For i = ActiveDocument.Comments.Count To 1 Step -1

Set cmt = ActiveDocument.Comments(i)

If InStr(cmt.Range.Text, myDeletionMarker) > 0 Then cmt.Delete

Next i

End Sub

**Sub DeleteAllLinks()**

**'** Paul Beverley - Version 30.10.12

' Delete all hyperlinks

' First the main text

linksHere = ActiveDocument.Hyperlinks.Count

linksTotal = linksHere

If linksHere > 0 Then

Selection.WholeStory

Selection.Fields.Unlink

End If

' then the endnotes, if there are any

linksHere = ActiveDocument.Endnotes.Count

linksTotal = linksTotal + linksHere

If linksHere > 0 Then

ActiveDocument.StoryRanges(wdEndnotesStory).Select

Selection.Fields.Unlink

End If

' then the footnotes, if there are any

linksHere = ActiveDocument.Footnotes.Count

linksTotal = linksTotal + linksHere

If linksHere > 0 Then

ActiveDocument.StoryRanges(wdFootnotesStory).Select

Selection.Fields.Unlink

End If

MsgBox ("Links deleted: " & Str(linksTotal))

End Sub

**Sub DeleteSomeLinks()**

' Paul Beverley - Version 31.10.12

' Delete hyperlinks that are not URLs

myColour = 0

myColour = wdGray25

linksDel = 0

linksTotal = 0

For pass = 1 To 3

Select Case pass

Case 1

linksHere = ActiveDocument.Hyperlinks.Count

If linksHere > 0 Then Set rng = ActiveDocument.Content

Case 2

linksHere = ActiveDocument.StoryRanges(wdEndnotesStory).Hyperlinks.Count

If linksHere > 0 Then Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3

linksHere = ActiveDocument.StoryRanges(wdFootnotesStory).Hyperlinks.Count

If linksHere > 0 Then Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

End Select

If linksHere > 0 Then

linksTotal = linksTotal + linksHere

For i = linksHere To 1 Step -1

myText = rng.Hyperlinks(i).TextToDisplay

myAddress = rng.Hyperlinks(i).Address

If InStr(myText, "www") = 0 And InStr(myText, "http") = 0 \_

And InStr(myAddress, "mailto") = 0 Then

Set rng2 = ActiveDocument.Hyperlinks(i).Range

If myColour > 0 Then rng2.HighlightColorIndex = myColour

ActiveDocument.Hyperlinks(i).Delete

linksDel = linksDel + 1

End If

Next i

End If

Next pass

linksDel = 0

MsgBox ("Links deleted: " & Str(linksDel) & " out of " & Str(linksTotal))

End Sub

**Sub ReferenceCheckWeb()**

' Paul Beverley - Version 30.08.16

' Check each hyperlink and highlight accordingly

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "References" & "^p"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

refsStart = rng.End

rng.End = ActiveDocument.Content.End

rng.Font.Color = wdColorRed

rng.Select

For Each lnk In ActiveDocument.Fields

thisLinkOK = True

If lnk.Kind <> 2 Then

lnk.Select

MsgBox "Different link kind"

thisLinkOK = False

End If

If lnk.Type <> 88 Then

lnk.Select

MsgBox "Different link type"

thisLinkOK = False

End If

linkCode = lnk.Code

myURL = Replace(linkCode, "HYPERLINK", "")

myURL = Trim(Replace(myURL, """", ""))

lnk.Select

If thisLinkOK = True Then

Set rng = ActiveDocument.Content

rng.Start = refsStart

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myURL

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If rng.Find.Found Then

rng.Font.Color = wdColorBlack

Selection.range.HighlightColorIndex = wdBrightGreen

Else

Selection.range.HighlightColorIndex = wdRed

End If

End If

' myVisibleText = lnk.Result

Next lnk

End Sub

**Sub FieldsUnlink()**

' Paul Beverley - Version 30.11.22

' Unlinks all fields except equations

doubleCheck = False

If doubleCheck = True Then

myResponse = MsgBox("Unlink all fields except equations?", vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

End If

linksDeleted = 0

' Do main text first

For i = ActiveDocument.Fields.Count To 1 Step -1

If ActiveDocument.Fields(i).Type <> 58 Then

linksDeleted = linksDeleted + 1

ActiveDocument.Fields(i).Unlink

If i Mod 50 = 0 Then StatusBar = "Links: " & Str(i)

End If

DoEvents

Next i

' Now do the textboxes, if there are any

If ActiveDocument.Shapes.Count > 0 Then

For shp = 1 To ActiveDocument.Shapes.Count

' Only check the text box if it has any text in it

If ActiveDocument.Shapes(shp).Type <> 24 And \_

ActiveDocument.Shapes(shp).Type <> 3 Then

If ActiveDocument.Shapes(shp).TextFrame.HasText Then

Set rng = ActiveDocument.Shapes(shp).TextFrame.TextRange

For Each fld In rng.Fields

If fld.Type <> 58 Then

fld.Unlink

linksDeleted = linksDeleted + 1

End If

Next fld

End If

End If

Next shp

DoEvents

End If

StatusBar = ""

MsgBox ("Fields unlinked: " & Str(linksDeleted))

End Sub

**Sub UnlinkCitationsAndRefs()**

' Paul Beverley - Version 29.06.20

' Unlinks reference citations (ignoring equations)

makeCitationItalic = True

endNoteCitation = 81

' Sort the citations

For Each fld In ActiveDocument.Fields

myType = fld.Type

If myType = wdFieldCitation Or myType = endNoteCitation Then

fld.Select

myStart = Selection.Start

Selection.Collapse wdCollapseEnd

fld.Unlink

Selection.Start = myStart

If makeCitationItalic = True Then Selection.Font.Italic = True

End If

Next fld

' Now the bibliography (reference list)

For Each fld In ActiveDocument.Fields

myType = fld.Type

If myType = wdFieldBibliography Then

fld.Select

fld.Unlink

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Exit Sub

End If

Next fld

End Sub

**Sub MendeleyPunctuationCorrection()**

' Paul Beverley - Version 13.09.19

' Moves punctuation marks to before the note indicator

For i = 1 To ActiveDocument.Fields.Count

Set fld = ActiveDocument.Fields(i)

tp = fld.Type

If tp = 81 Then

fld.Select

st = Selection.Start

Set rng = ActiveDocument.range(st, st)

Selection.MoveRight , 2

Do While Selection.Start > st + 2

st = Selection.Start

Selection.MoveRight , 2

Loop

Set rngpun = ActiveDocument.range(st - 1, st)

rngpun.Select

pun = Selection.Text

If InStr(",.?!:;", pun) > 0 Then

rng.InsertBefore pun

rngpun.Delete

End If

End If

Next i

Beep

End Sub

**Sub DeleteAllFigures()**

' Paul Beverley - Version 31.08.16

' Delete all images from a file that have a caption

For Each myPic In ActiveDocument.InlineShapes

Set rng = myPic.range

rng.Collapse wdCollapseEnd

rng.MoveEnd wdCharacter, 1

rng.Select

If Asc(rng) = 13 Then rng.Delete

rng.MoveEnd wdCharacter, 1

If Asc(rng) = 13 Then rng.Delete

rng.MoveEnd wdCharacter, 3

gotOne = (rng = "(a)" Or rng = "(b)" Or rng = "(c)")

rng.MoveEnd wdCharacter, 15

If InStr(rng, "Fig") > 0 Or gotOne = True Then

myPic.Delete

i = i + 1

End If

Next myPic

MsgBox "Deleted:" & i & " figures"

End Sub

**Sub DeleteAllInlineImages()**

' Paul Beverley - Version 06.12.16

' Delete all inline from a file

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

i = ActiveDocument.InlineShapes.Count

For Each pic In ActiveDocument.InlineShapes

pic.Delete

DoEvents

i = i - 1

StatusBar = " " & i

Next pic

Beep

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub DeleteAllImagesAndCloseUp()**

' Paul Beverley - Version 13.12.16

' Deletes all images and closes the gaps

For Each pic In ActiveDocument.InlineShapes

pic.range.Select

Selection.Delete

Selection.MoveEndWhile cset:=Chr(13), Count:=wdForward

Selection.MoveStartWhile cset:=Chr(13), Count:=wdBackward

Selection.MoveStart , 1

Selection.Delete

Next pic

End Sub

**Sub DeleteAllImagesAddCallout()**

' Paul Beverley - Version 06.12.16

' Deletes all images from a file and leave a caption line.

' chapNumber = Mid(ActiveDocument.Name, 2, 1)

' or maybe

' chapNumber = Left(ActiveDocument.Name, 2)

' If Left(chapNumber,1) = "0" Then chapNumber = Right(chapNumber,1)

chapNumber = "19"

myCaption = vbCr & "<Figure " & chapNumber & ".FigNum about here>" & vbCr

numPics = ActiveDocument.InlineShapes.Count

myResponse = MsgBox(Str(numPics) & " images to be deleted. OK?", vbQuestion \_

+ vbYesNoCancel, "Delete All Images")

If myResponse <> vbYes Then Exit Sub

i = 1

For Each pic In ActiveDocument.InlineShapes

pic.range.InsertAfter Text:=Replace(myCaption, "FigNum", Trim(Str(i)))

pic.Delete

i = i + 1

Next pic

End Sub

**Sub ItalicParaDelete()**

' Paul Beverley - Version 23.03.17

' Deletes all paragraphs that are mainly in italic

deletionFactor = 6

' i.e. delete if there are (deletionFactor) times

' as many italic words as roman words

For Each myPara In ActiveDocument.Paragraphs

numWords = myPara.range.Words.Count

numItalic = 0

For i = 1 To numWords

If myPara.range.Words(i).Font.Italic = True Then

numItalic = numItalic + 1

End If

Next i

numRoman = numWords - numItalic

If numItalic > deletionFactor \* numRoman Then

myPara.range.Font.DoubleStrikeThrough = True

End If

Next myPara

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.DoubleStrikeThrough = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub ComboBoxAccept()**

' Paul Beverley - Version 14.04.16

' Find combo boxes and replace with the currently selected text

' Provided by Howard Silcock of New Zealand

For Each cc In ActiveDocument.ContentControls

cc.LockContentControl = False

cc.Delete

Next

End Sub

**Sub ColonUnbold()**

' Paul Beverley - Version 06.01.18

' Romanises bold colons that are followed by roman text

Set rng = ActiveDocument.Content

' Go and find the first occurrence

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ":"

.Font.Bold = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While rng.Find.Found = True

rng.End = rng.Start + 3

dontDoIt = (InStr(rng, Chr(13)) > 0)

rng.Start = rng.Start + 2

If rng.Font.Bold = False And dontDoIt = False Then

rng.Start = rng.Start - 2

rng.Font.Bold = False

End If

rng.Collapse wdCollapseEnd

' Go and find the next occurence (if there is one)

rng.Find.Execute

Loop

Beep

End Sub

**Sub AutoListOffSimple()**

**'** Paul Beverley - Version 01.06.10

' Auto-lists to text

ActiveDocument.ConvertNumbersToText

End Sub

**Sub AutoListOff()**

' Paul Beverley - Version 22.11.16

' Changes auto-bulleted listing to real bullets

changeBullets = True

newBlackBullet = ChrW(8226)

' newBlackBullet = "\*": ' asterisk

newWhiteBullet = ChrW(9702)

newSquareBullet = ChrW(9642)

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

ActiveDocument.ConvertNumbersToText

If changeBullets = True Then

normalFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

' One common type of bullet uses Symbol font

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Text = ChrW(&HF0B7) & "^t"

.Forward = True

.Replacement.Text = newBlackBullet & "^t"

.Replacement.Font.Name = normalFont

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

' Another uses Wingding font

Set rng = ActiveDocument.range

With rng.Find

.Text = ChrW(&HF0FC) & "^t"

.Font.Name = "Wingdings"

.Replacement.Text = newBlackBullet & "^t"

.Replacement.Font.Name = normalFont

.Execute Replace:=wdReplaceAll

End With

' Sub-bullet points sometimes use lowercase o

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Text = "^po^t"

.Forward = True

.Replacement.Text = "^p" & newWhiteBullet & "^t"

.Replacement.Font.Name = normalFont

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

' Sub-sub-bullet points are square bullets

Set rng = ActiveDocument.range

With rng.Find

.Text = ChrW(&HF0A7) & "^t"

.Replacement.Text = newSquareBullet & "^t"

.Replacement.Font.Name = normalFont

.Execute Replace:=wdReplaceAll

End With

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub FirstNotIndent()**

**'** Paul Beverley - Version 01.06.10

' Full-out paragraph under all headings

StyleList = "Heading 1, Heading 2, Heading 3, and any more you want"

NoIndentStyle = "Body Text"

IndentStyle = "Body Text First Indent"

For i = 1 To (ActiveDocument.Paragraphs.Count - 1)

StyleNow = ActiveDocument.Paragraphs(i).Range.Style

StyleNext = ActiveDocument.Paragraphs(i + 1).Range.Style

If InStr(StyleList, StyleNow) And StyleNext = IndentStyle Then

ActiveDocument.Paragraphs(i + 1).Range.Style = NoIndentStyle

End If

Next

End Sub

**Sub JustifyOFF()**

**'** Paul Beverley - Version 01.06.10

' Turn format off on all paragraphs

For Each myPara In ActiveDocument.Paragraphs

If myPara.Range.ParagraphFormat.Alignment = wdAlignParagraphJustify Then

myPara.Range.ParagraphFormat.Alignment = wdAlignParagraphLeft

End If

Next

End Sub

**Sub FunnyChange()**

**'** Paul Beverley - Version 01.06.10

' Make various complicated changes to all paragraphs

For Each myPara In ActiveDocument.Paragraphs

If myPara.Range.ParagraphFormat.Alignment = wdAlignParagraphJustify \_

And myPara.Range.Font.Size = 14 And myPara.Range.Font.Bold = True Then

myPara.Range.ParagraphFormat.Alignment = wdAlignParagraphLeft

myPara.Range.Font.Size = 12

myPara.Range.Font.Bold = False

myPara.Range.Font.Italic = True

End If

Next

End Sub

**Sub SuperSubConvert()**

' Paul Beverley - Version 20.11.17

' Changes weird super/subscript format to proper ones

myColour = wdYellow

myUline = wdPink

myColour2 = wdRed

maxRaise = 7

minRaise = 3

maxLower = -7

minLower = -2

doSelection = (Len(Selection) > 1)

If doSelection Then

Set rng = Selection.range.Duplicate

Else

Set rng = ActiveDocument.Content

End If

For Each myPara In rng.Paragraphs

If myPara.range.Font.Position > 9999 Then

For Each wd In myPara.range.Words

If myPara.range.Font.Position > 9999 Then

For Each ch In wd.Characters

myPos = ch.Font.Position

If ch.Font.Underline = 1 Then

ch.Font.Underline = False

ch.HighlightColorIndex = myUline

If doSelection = False Then ch.Select

End If

' Raise text to superscript

If myPos >= minRaise And myPos <= maxRaise Then

ch.Font.Position = 0

ch.Font.Superscript = True

ch.HighlightColorIndex = myColour

If doSelection = False Then ch.Select

End If

' Lowered text to subscript

If myPos <= minLower And myPos >= maxLower Then

ch.Font.Position = 0

ch.Font.Subscript = True

ch.HighlightColorIndex = myColour2

If doSelection = False Then ch.Select

End If

Next ch

End If

DoEvents

Next wd

End If

Next myPara

Beep

End Sub

**Sub SymbolFontCheck()**

**'** Paul Beverley - Version 23.02.10

' Greek symbol font checker

startNumber = &HF000

endNumber = &HF0FF

For i = startNumber To endNumber

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(i)

.MatchCase = True

.Execute

End With

myFont = rng.Font.Name

If rng.Find.Found = True Then

If myFont = "Symbol" Then rng.HighlightColorIndex = wdYellow

If Left(myFont, 4) = "Wing" Then rng.HighlightColorIndex = wdRed

rng.Start = rng.End

rng.Select

Selection.TypeText Text:="<&H" & Hex(i) & ">|"

Selection.Start = rng.End

Selection.Font.Reset

Selection.Range.Style = ActiveDocument.Styles(wdStyleNormal)

End If

StatusBar = " checking: " & Str(endNumber - i)

Next i

StatusBar = "Finished"

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<&H^?^?^?^?^?^?"

.Forward = True

.Wrap = wdFindContinue

.MatchCase = False

.MatchWholeWord = False

.MatchWildcards = False

.Execute

End With

End Sub

**Sub SymbolToUnicode()**

' Paul Beverley - Version 11.11.13

' Converts Symbol font characters to Unicode characters

' myColourFound = wdYellow

myColourFound = wdNoHighlight

myColourNotFound = wdTurquoise

myColourWarning = wdRed

myResponse = MsgBox("Go ahead without a test run?", vbQuestion \_

+ vbYesNoCancel, "SymbolToUnicode")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then

testRun = True

Else

testRun = False

End If

' Greek type

myList = "F044,0394; F046,03A6; F061,03B1; F062,03B2; F065,03B5;"

myList = myList & "F067,03B3; F068,03B7; F071,03B8; F069,03B9; F063,03C7;"

myList = myList & "F056,03C2; F074,03C4; F077,03C9; F078,03BE; F079,03C8;"

myList = myList & "F057,03A9; F066,03D5; F06B,03BA; F072,03C1; F073,03C3;"

myList = myList & "F06A,03C6; F06C,03BB; F06D,03BC; F06E,03BD; F070,03C0;"

myList = myList & "F075,03C5; F076,03C9; F047,0393; F07A,03B6; F059,03A8;"

myList = myList & "F04C,039B; F050,03A0; F051,0398; F053,03A3; F058,039E;"

myList = myList & "F04A,03D1; F064,03B4; F009,03B4; (both delta)"

myList = myList & ""

' Maths symbol type

myList = myList & "F0AC,2190; F0AD,2191; F0AE,2192; F0AF,2193; F0B8,00F7;"

myList = myList & "F0DC,21D0; F0DD,21D1; F0DE,21D2; F0DF,21D3; F0A3,2264;"

myList = myList & "F0D7,22C5; F0C5,2295; F0C7,2229; F0C8,222A; F0C9,2283;"

myList = myList & "F0CA,2287; F0CB,2284; F0CC,2282; F0CD,2286; F0A5,221E;"

myList = myList & "F0B5,221D; F0B9,2260; F0BB,2248; F0CE,220A; F0CF,2209;"

myList = myList & "F0DB,21D4; F0C1,2111; F0B6,2202; F0C2,211C; F0C3,2118;"

myList = myList & "F0D6,221A; F0B4,00D7; F0A4,2265; F0B1,00B1; F0D1,2207;"

myList = myList & "F02D,2212; F0B3,2265; F0BA,2261; F022,2200; F0A2,2032;"

myList = myList & "F0B7,2022; F052,211D; "

' Ordinary characters, space, comma, etc

myList = myList & "F020,0020; F02C,002C; F07D,007D; F07B,007B;"

myList = myList & "F028,0028; F029,0029; F02B,002B; F03D,003D;"

myList = myList & "F030,0030; F031,0031; F032,0032; F033,0033;"

myList = myList & "F034,0034; F035,0035; F036,0036; F037,0037;"

myList = myList & "F038,0038; F039,0039; F02F,002F; F03D,002F;"

myList = myList & "F0B0,00B0; F03C,003C; F03E,003E; F02E,002E;"

myList = myList & "F03B,003B; F07C,007C; F02A,002A; F0D2;00AE;"

myFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

ActiveDocument.TrackRevisions = False

gotOne = False

For Each myChar In ActiveDocument.Characters

ascChar = Asc(myChar)

ascWChar = AscW(myChar)

myFontName = myChar.Font.Name

If ascChar = 40 Or ascChar = 63 Then

myChar.Select

myFontName = Selection.Font.Name

ascWChar = Dialogs(wdDialogInsertSymbol).CharNum

End If

If gotOne = False And ascWChar < 0 Then

myChar.Select

symbolCode = Replace(Hex(ascWChar), "FFFF", "")

myPos = InStr(myList, symbolCode)

If myPos > 0 And ascWChar <> 40 Then

gotOne = True

uCode = Val("&H" & Mid(myList, myPos + 5, 4))

If testRun = False Then

Selection.Font.Name = myFont

Selection.TypeText ChrW(uCode)

Selection.MoveStart , -1

Else

Selection.Collapse wdCollapseEnd

Selection.TypeText " "

Selection.MoveStart , -1

Selection.Font.Name = myFont

Selection.TypeText ChrW(uCode)

Selection.MoveStart , -1

End If

Selection.Range.HighlightColorIndex = myColourFound

If myFontName <> "Symbol" Then

Selection.Range.HighlightColorIndex = myColourWarning

Beep

End If

Else

If Not (myFontName = myFont And Asc(myChar) = 40) Then \_

Selection.Range.HighlightColorIndex = myColourNotFound

gotOne = False

Beep

End If

Else

gotOne = False

End If

Next

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub FontHighlight()**

**'** Paul Beverley - Version 03.10.11

' Highlight all fonts NOT named in the list

myBestColour = wdTurquoise

nonHighlight = 5

ReDim myFont(nonHighlight) As String

myFont(1) = "Calibri"

myFont(2) = "Times New Roman"

myFont(3) = "Garamond"

myFont(4) = "Cambria"

myFont(5) = ""

nowColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myBestColour

Set rng = ActiveDocument.Content

rng.Font.Shadow = True

For i = 1 To nonHighlight

If myFont(i) > "" Then

thisFont = myFont(i)

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = ""

.Replacement.Text = ""

.Font.Name = thisFont

.Wrap = wdFindContinue

.Replacement.Font.Shadow = False

.Execute Replace:=wdReplaceAll

End With

End If

Next i

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = ""

.Wrap = wdFindContinue

.Font.Shadow = True

.Replacement.Font.Shadow = False

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = nowColour

End Sub

**Sub FontEliminate()**

' Paul Beverley - Version 16.02.13

' Restore anything in this font to the default font

myUnwantedFont = Selection.Font.Name

myBaseStyle = ActiveDocument.Styles(wdStyleNormal)

normalFont = ActiveDocument.Styles(myBaseStyle).Font.Name

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Font.Name = myUnwantedFont

.Replacement.Text = ""

.Replacement.Font.Name = normalFont

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub FunnyFontFind()**

' Paul Beverley - Version 07.09.17

' Finds the next paragraph that has mixed fonts

Set rng = ActiveDocument.range(0, Selection.End)

myStart = rng.Paragraphs.Count

For i = myStart + 1 To ActiveDocument.Paragraphs.Count

If ActiveDocument.Paragraphs(i).range.Font.Name = "" Then

ActiveDocument.Paragraphs(i).range.Select

Exit Sub

End If

Next i

Selection.EndKey Unit:=wdStory

Beep

End Sub

**Sub FunnyFontClear()**

' Paul Beverley - Version 07.09.17

' Makes all text in the selection the same font

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = Selection.range.Duplicate

startFont = Selection.range.Characters(1).Font.Name

For i = 1 To rng.Characters.Count

Set ch = rng.Characters(i)

If ch.Font.Name <> startFont Then

myChar = Asc(ch.Text)

ch.Select

ch.Delete

Selection.Text = Chr(myChar)

End If

Next i

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub FontLister()**

' Paul Beverley - Version 07.04.21

' Lists all font names in selected text or whole file

' stopWhenFound = True

stopWhenFound = False

myFileTitle = "Fonts list"

If Selection.Start = Selection.End Then

doAll = True

Set rng = ActiveDocument.Content

Else

doAll = False

Set rng = Selection.Range.Duplicate

End If

CR = vbCr

fontList = CR

If doAll = True Then

For Each myPar In rng.Paragraphs

nm = myPar.Range.Font.Name

If nm > "" Then

If InStr(fontList, nm) = 0 Then

fontList = fontList & nm & CR

If stopWhenFound = True Then myPar.Range.Select: MsgBox nm

End If

Else

For Each wd In myPar.Range.Words

nm = wd.Font.Name

If nm > "" Then

If InStr(fontList, nm) = 0 Then

fontList = fontList & nm & CR

If stopWhenFound = True Then

wd.Select

MsgBox nm

End If

End If

Else

For Each ch In wd.Characters

nm = ch.Font.Name

If InStr(fontList, nm) = 0 Then

fontList = fontList & nm & CR

If stopWhenFound = True Then ch.Select: MsgBox nm

End If

Next ch

End If

Next wd

End If

DoEvents

Next myPar

Documents.Add

Selection.HomeKey Unit:=wdStory

Selection.TypeText CR

Selection.TypeText fontList

Selection.Start = 0

Selection.Sort SortOrder:=wdSortOrderAscending

Selection.HomeKey Unit:=wdStory

Selection.TypeText myFileTitle

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

Else

For Each ch In rng.Characters

nm = ch.Font.Name

If InStr(fontList, nm) = 0 Then fontList = fontList & nm & CR

Next ch

MsgBox fontList

End If

End Sub

**Sub FontFind()**

' Paul Beverley - Version 08.09.17

' Finds text in a given font name in selected text or whole file

nmlFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

If Selection.End - Selection.Start < 2 Then

Selection.End = Selection.Start + 1

fn = Selection.range.Font.Name

If fn = nmlFont Then

Selection.Expand wdParagraph

Selection.MoveEnd , -1

fn = Trim(Selection.Text)

Selection.MoveEnd , 1

End If

Else

If Selection.range.Font.Name = "" Then

Selection.End = Selection.Start + 1

fn = Selection.range.Font.Name

Else

fn = Trim(Selection.range.Text)

End If

End If

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Name = fn

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If Not (Selection.Find.Found) Then Beep

End Sub

**Sub FontFunniesClearThisOne()**

' Paul Beverley - Version 08.09.17

' Makes the selected text into the default font

nmlFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

If Selection.range.Font.Name = nmlFont Then

Selection.Collapse wdCollapseStart

hereNow = Selection.End

Selection.Find.Execute

If Selection.Start = hereNow Then Beep

Exit Sub

End If

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = Selection.range.Duplicate

startFont = Selection.range.Characters(1).Font.Name

For I = 1 To rng.Characters.Count

Set ch = rng.Characters(I)

If ch.Font.Name <> nmlFont Then

myChar = Asc(ch.Text)

myCharW = AscW(ch.Text)

isSuper = (ch.Font.Superscript = True)

isSub = (ch.Font.Subscript = True)

ch.Select

ch.Delete

If myCharW < 0 Then

Selection.Text = Chr(myChar)

Else

Selection.Text = ChrW(myCharW)

End If

If isSuper = False And isSub = False Then

Selection.Font.Superscript = False

Selection.Font.Subscript = False

End If

If isSuper Then Selection.Font.Superscript = True

If isSub Then Selection.Font.Subscript = True

End If

Next I

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub FontFunniesClearAll()**

**'** Paul Beverley - Version 22.07.22

' Changes all text in this specific font into the default font

highlightChanges = True

myColour = wdBrightGreen

badFont = Selection.Font.Name

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Application.ScreenUpdating = False

On Error GoTo ReportIt

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^?"

.Font.Name = badFont

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myCount = 0

Do While Selection.Find.Found = True

myCount = myCount + 1

endNow = Selection.End

myChar = Asc(Selection.Text)

myCharW = AscW(Selection.Text)

isSuper = (Selection.Font.Superscript = True)

isSub = (Selection.Font.Subscript = True)

Selection.Delete

If myCharW < 0 Then

Selection.Text = Chr(myChar)

Else

Selection.Text = ChrW(myCharW)

End If

If isSuper = False And isSub = False Then

If Selection.Font.Superscript = True Then Selection.Font.Superscript = False

If Selection.Font.Subscript = True Then Selection.Font.Subscript = False

End If

If isSuper Then Selection.Font.Superscript = True

If isSub Then Selection.Font.Subscript = True

If highlightChanges = True Then Selection.Range.HighlightColorIndex = myColour

Selection.Start = endNow

Selection.Find.Execute

Loop

MsgBox "Changed: " & myCount

ActiveDocument.TrackRevisions = myTrack

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub UnitSpacer()**

' Paul Beverley - Version 07.12.15

' Find units with numbers and add a thin space

'myColour = wdNoHighlight

myColour = wdBrightGreen

'mySpace = "<spc>"

'mySpace = ChrW(160) ' non-breaking space

mySpace = ChrW(8201) ' thin space

' one- and two-letter words to ignore

ignoreThese = ",a,b,c,d,e,f,g,h,i,j,k,n,o,p,q,r,t,u,v,w,x,y,z," & \_

",D,E,I,O,P,X,Y,Z" & \_

",AD,An,an,as,As,at,BC,be,by,do,en,gh,IC,ie,if,If," & \_

"in,In,is,Is,it,It,nd,of,no,No,on,On," & \_

",or,pp,rd,Re,re,so,So,st,th,to,To,UK,US,vs,we,"

' three-or-more letter words to include

includeThese = "kWh,MPa,kHz,MHz"

' three-or-more letter words to ignore

excludeThese = "exp,"

' Avoid things like "Fig. 2.3 A view..."

notAfterThese = "Fig,Figure,Table,Box,Section,Chapter"

' Mask off special constructions, e.g. H2O, BS5261C

maskMixedNumbers = True

mySymbols = ChrW(176) & ChrW(181) ' degree, mu (micro)

myLanguage = wdEnglishUK

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

ignoreThese = "," & ignoreThese & "," & excludeThese & ","

includeThese = "," & includeThese & ","

For hit = 1 To 4

Select Case hit

Case 1: goes = ActiveDocument.Footnotes.Count

If goes > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Set rng2 = ActiveDocument.StoryRanges(wdFootnotesStory)

Set rng3 = ActiveDocument.StoryRanges(wdFootnotesStory)

goes = 1

End If

Case 2: goes = ActiveDocument.Endnotes.Count

If goes > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Set rng2 = ActiveDocument.StoryRanges(wdEndnotesStory)

Set rng3 = ActiveDocument.StoryRanges(wdEndnotesStory)

goes = 1

End If

Case 3: goes = 1: Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

Set rng3 = ActiveDocument.Content

Case 4: goes = ActiveDocument.Shapes.Count

End Select

i = goes

Do While i > 0

doIt = True

If hit = 4 Then

If shp.Type <> 24 And shp.Type <> 3 Then

doIt = ActiveDocument.Shapes(i).TextFrame.HasText

If doIt Then

Set rng = ActiveDocument.Shapes(i).TextFrame.TextRange

Set rng2 = ActiveDocument.Shapes(i).TextFrame.TextRange

Set rng3 = ActiveDocument.Shapes(i).TextFrame.TextRange

End If

End If

End If

If doIt = True Then

If maskMixedNumbers = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-zA-Z\-]{1,}[0-9]{1,}[\-a-zA-Z0-9]{1,}"

.Wrap = wdFindContinue

.Replacement.Text = "^&"

.Replacement.Font.DoubleStrikeThrough = True

.Forward = True

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.Text = "[0-9]{1,}[\-a-zA-Z]{1,}[0-9]{1,}"

.Replacement.Font.DoubleStrikeThrough = True

.Execute Replace:=wdReplaceAll

End With

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]{1,}[ a-zA-Z" & mySymbols & "]{1,}"

.Font.Superscript = False

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute

End With

myEnd = rng.End

Do While rng.Find.Found = True

gotOne = True

rng2.Start = rng.Start

rng2.End = rng.Start

rng2.MoveStart wdWord, -1

allBefore = rng2.Text

Do

rng2.MoveStart wdWord, -1

rng2.MoveEnd wdWord, -1

allBefore = rng2.Text & allBefore

Loop Until LCase(rng2) <> UCase(rng2) Or rng2.Start < 50

rng.Collapse wdCollapseStart

rng.Find.Text = "[0-9]{1,}"

rng.Find.Execute

numFound = rng.Text

myColNum = rng.Information(wdStartOfRangeColumnNumber)

myRowNum = rng.Information(wdStartOfRangeRowNumber)

rng.Collapse wdCollapseEnd

startUnit = rng.Start

rng.MoveEnd , 1

gotSpc = (Asc(rng.Text) = 32)

If gotSpc Then

rng.Collapse wdCollapseEnd

startUnit = rng.Start

Else

rng.Collapse wdCollapseStart

End If

rng.Find.Text = "[a-zA-Z" & mySymbols & "]{1,}"

rng.Find.Execute

If rng.Start > myEnd Then rng.Collapse wdCollapseStart

If rng.Information(wdStartOfRangeColumnNumber) <> myColNum \_

Then gotOne = False

If rng.Information(wdStartOfRangeRowNumber) <> myRowNum \_

Then gotOne = False

wd = rng.Text

If wd > "" Then isDegree = (Asc(rng.Text) = 176)

If Len(wd) > 3 Then gotOne = False

If InStr(ignoreThese, "," & wd & ",") > 0 Then gotOne = False

If Len(wd) = 3 Then

If Application.CheckSpelling(wd, MainDictionary:= \_

Languages(myLanguage).NameLocal) = True Then

gotOne = False

End If

End If

If InStr(includeThese, "," & wd & ",") > 0 Then gotOne = True

If Len(wd) = 0 Then gotOne = False

num = Val(numFound)

' Trap "In the 1980s..."

If num > 1700 And num < 2100 Then

If wd = "s" Then gotOne = False

End If

' Check the previous word

If InStr(allBefore, Chr(13)) > 0 Then gotOne = False

numFPs = Len(allBefore) - Len(Replace(allBefore, ".", ""))

If numFPs > 1 Then gotOne = False

If InStr(notAfterThese, Trim(rng2.Text)) > 0 Then gotOne = False

If rng.Font.DoubleStrikeThrough = True Then gotOne = False

If rng.Font.StrikeThrough = True Then gotOne = False

If gotOne = True Then

' add the <whatever> space

noHighlight = False

If gotSpc Then

rng.Collapse wdCollapseStart

rng.Start = rng.Start - 1

' For Symbol font degree

If Asc(rng) = 40 Then

rng.Start = rng.Start - 1

rng.End = rng.Start + 1

rng.Delete

noHighlight = True

Else

If isDegree = True Then

rng.Delete

noHighlight = True

Else

rng.Text = mySpace

DoEvents ' Debug.Print wd

End If

End If

Else

If Asc(rng) <> 176 Then

rng.InsertBefore Text:=mySpace

Else

noHighlight = True

End If

rng.End = rng.Start + Len(mySpace)

End If

If noHighlight = False And myColour > 0 Then

rng.HighlightColorIndex = myColour

End If

j = j + 1

If j Mod 10 = 2 Then rng.Select

End If

rng.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

rng.Find.Text = "[0-9]{1,}[ a-zA-Z]{1,}"

rng.Find.Execute

myEnd = rng.End

Loop

End If

i = i - 1

If i > 0 Then

DoEvents ' Debug.Print "Textbox: " & Str(i)

StatusBar = "Textbox: " & Str(i)

End If

rng3.Font.DoubleStrikeThrough = False

Loop

Next hit

Beep

Selection.HomeKey Unit:=wdStory

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub LanguageSetUK()**

' Paul Beverley - Version 09.04.23

' Sets language as UK English

myLanguage = wdEnglishUK

' If you want all the styles to be language-set as well,

' make it true:

setStyleLanguage = False

' setStyleLanguage = True

ActiveDocument.Content.LanguageID = myLanguage

For Each aStory In ActiveDocument.StoryRanges

Do

aStory.LanguageID = myLanguage

If aStory.NextStoryRange Is Nothing Then

MoreStoryRanges = False

Else

MoreStoryRanges = True

Set aStory = aStory.NextStoryRange

End If

Loop While MoreStoryRanges

Next aStory

If ActiveDocument.Shapes.Count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

shp.TextFrame.TextRange.LanguageID = myLanguage

End If

End If

Next

End If

If setStyleLanguage = True Then

For i = 1 To ActiveDocument.Styles.Count

Set sty = ActiveDocument.Styles(i)

If sty.Type = wdStyleTypeParagraph Then \_

sty.LanguageID = myLanguage

Next i

End If

ActiveDocument.Styles(wdStyleNormal).LanguageID = myLanguage

ActiveDocument.Styles(wdStyleCommentText).LanguageID = myLanguage

ActiveDocument.Styles("Balloon Text").LanguageID = myLanguage

End Sub

**Sub LanguageSetUS()**

' Paul Beverley - Version 09.04.23

' Sets language as US English

' If you want all the styles to be language-set as well,

' make it true:

setStyleLanguage = False

' setStyleLanguage = True

myLanguage = wdEnglishUS

ActiveDocument.Content.LanguageID = myLanguage

For Each aStory In ActiveDocument.StoryRanges

Do

aStory.LanguageID = myLanguage

If aStory.NextStoryRange Is Nothing Then

MoreStoryRanges = False

Else

MoreStoryRanges = True

Set aStory = aStory.NextStoryRange

End If

Loop While MoreStoryRanges

Next aStory

If setStyleLanguage = True Then

For i = 1 To ActiveDocument.Styles.Count

Set sty = ActiveDocument.Styles(i)

If sty.Type = wdStyleTypeParagraph Then \_

sty.LanguageID = myLanguage

Next i

End If

If ActiveDocument.Shapes.Count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

shp.TextFrame.TextRange.LanguageID = myLanguage

End If

End If

Next

End If

ActiveDocument.Styles(wdStyleNormal).LanguageID = myLanguage

ActiveDocument.Styles(wdStyleCommentText).LanguageID = myLanguage

ActiveDocument.Styles("Balloon Text").LanguageID = myLanguage

End Sub

**Sub LanguageSetCanada()**

' Paul Beverley - Version 09.04.23

' Sets language as Canadian English

' If you want all the styles to be language-set as well,

' make it true:

setStyleLanguage = False

' setStyleLanguage = True

myLanguage = wdEnglishCanadian

ActiveDocument.Content.LanguageID = myLanguage

For Each aStory In ActiveDocument.StoryRanges

Do

aStory.LanguageID = myLanguage

If aStory.NextStoryRange Is Nothing Then

MoreStoryRanges = False

Else

MoreStoryRanges = True

Set aStory = aStory.NextStoryRange

End If

Loop While MoreStoryRanges

Next aStory

If setStyleLanguage = True Then

For i = 1 To ActiveDocument.Styles.count

Set sty = ActiveDocument.Styles(i)

If sty.Type = wdStyleTypeParagraph Then \_

sty.LanguageID = myLanguage

Next i

End If

If ActiveDocument.Shapes.count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

shp.TextFrame.TextRange.LanguageID = myLanguage

End If

End If

Next

End If

ActiveDocument.Styles(wdStyleNormal).LanguageID = myLanguage

ActiveDocument.Styles(wdStyleCommentText).LanguageID = myLanguage

ActiveDocument.Styles("Balloon Text").LanguageID = myLanguage

End Sub

**Sub LanguageUSUKswitch()**

**'** Paul Beverley - Version 22.04.24

' Switches language between UK and US English

' If you want all the styles to be language-set as well,

' make it true:

setStyleLanguage = False

' setStyleLanguage = True

langNow = Selection.LanguageID

If langNow = wdEnglishUK Then

nowLanguage = "UK English"

newLanguage = "US English"

Else

nowLanguage = "US English"

newLanguage = "UK English"

End If

myResponse = MsgBox("Currently " & nowLanguage & vbCr \_

& vbCr & "Switch to " & newLanguage & "?", \_

vbQuestion + vbYesNo, "LanguageUSUKswitch")

If myResponse <> vbYes Then Exit Sub

If langNow = wdEnglishUK Then

myLanguage = wdEnglishUS

Else

myLanguage = wdEnglishUK

End If

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

ActiveDocument.Content.LanguageID = myLanguage

For Each aStory In ActiveDocument.StoryRanges

Do

aStory.LanguageID = myLanguage

If aStory.NextStoryRange Is Nothing Then

MoreStoryRanges = False

Else

MoreStoryRanges = True

Set aStory = aStory.NextStoryRange

End If

Loop While MoreStoryRanges

Next aStory

If ActiveDocument.Shapes.Count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

shp.TextFrame.TextRange.LanguageID = myLanguage

End If

End If

Next

End If

If setStyleLanguage = True Then

For i = 1 To ActiveDocument.Styles.Count

Set sty = ActiveDocument.Styles(i)

If sty.Type = wdStyleTypeParagraph Then \_

sty.LanguageID = myLanguage

Next i

End If

ActiveDocument.Styles(wdStyleNormal).LanguageID = myLanguage

ActiveDocument.Styles(wdStyleCommentText).LanguageID = myLanguage

ActiveDocument.SpellingChecked = False

ActiveDocument.GrammarChecked = False

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub SpracheDE()**

' Paul Beverley - Version 09.04.23

' Sets language as German

' myLanguage = wdGermanLiechtenstein

' myLanguage = wdGermanAustria

' myLanguage = wdGermanLuxembourg

myLanguage = wdGerman

ActiveDocument.Content.LanguageID = myLanguage

For Each aStory In ActiveDocument.StoryRanges

aStory.LanguageID = myLanguage

Next aStory

If ActiveDocument.Shapes.Count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.hasText Then

shp.TextFrame.TextRange.LanguageID = myLanguage

End If

End If

Next

End If

ActiveDocument.Styles(wdStyleNormal).LanguageID = myLanguage

ActiveDocument.Styles(wdStyleCommentText).LanguageID = myLanguage

ActiveDocument.Styles("Balloon Text").LanguageID = myLanguage

End Sub

**Sub SpracheCH()**

' Paul Beverley - Version 09.04.23

' Sets language as SwissGerman

myLanguage = wdSwissGerman

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

ActiveDocument.Content.LanguageID = myLanguage

For Each aStory In ActiveDocument.StoryRanges

aStory.LanguageID = myLanguage

Next aStory

If ActiveDocument.Shapes.Count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.hasText Then

shp.TextFrame.TextRange.LanguageID = myLanguage

End If

End If

Next

End If

ActiveDocument.Styles(wdStyleNormal).LanguageID = myLanguage

ActiveDocument.Styles(wdStyleCommentText).LanguageID = myLanguage

ActiveDocument.Styles("Balloon Text").LanguageID = myLanguage

End Sub

**Sub LanguageHighlight()**

**'** Paul Beverley - Version 05.12.11

' Highlight all text not in main language

myColour = wdGray25

mainLanguage = Selection.LanguageID

For Each myPara In ActiveDocument.Paragraphs

If myPara.Range.LanguageID <> mainLanguage Then

If myPara.Range.LanguageID = 9999999 Then

For Each myWord In myPara.Range.Words

If myWord.LanguageID <> mainLanguage Then

myWord.HighlightColorIndex = myColour

End If

Next myWord

Else

myPara.Range.HighlightColorIndex = myColour

End If

End If

Next myPara

End Sub

**Sub LongSentenceCheck()**

**'** Paul Beverley - Version 06.01.11

' Colour long sentences

myCol1 = wdColorBlue

myCol2 = wdColorSeaGreen

minLen = 40

' nowTrack = ActiveDocument.TrackRevisions

' ActiveDocument.TrackRevisions = False

myCol = myCol1

For Each snt In ActiveDocument.Sentences

allWords = snt

wordNum = Len(allWords) - Len(Replace(allWords, " ", ""))

If wordNum > minLen Then

snt.Font.Color = myCol

' Swap between the two colours ready for next long sentence

If myCol = myCol1 Then

myCol = myCol2

Else

myCol = myCol1

End If

End If

Next snt

' ActiveDocument.TrackRevisions = nowTrack

End Sub

**Sub DoubleSpaceAfterSentence()**

' Paul Beverley - Version 28.02.18

' Ensures that every sentence has TWO spaces after it

For Each sn In ActiveDocument.Sentences

sn.Select

If Len(sn) > 5 Then

lastTwo = Right(sn.Text, 2)

lastChar = Right(sn.Text, 1)

If Right(sn.Text, 2) <> " " And lastChar <> vbCr And \_

sn.Information(wdWithInTable) = False And \_

lastChar <> vbTab Then

sn.InsertAfter " "

i = i + 1

StatusBar = "Sentences done: " & i

End If

End If

Next sn

Beep

End Sub

**Sub HighlightAllQuestions()**

**'** Paul Beverley - Version 15.02.11

' Highlights all questions

qnColour = wdBrightGreen

For Each sntce In ActiveDocument.Sentences

myWords = sntce

If Asc(Right(myWords, 2)) = Asc("?") Then

sntce.HighlightColorIndex = qnColour

End If

Next

End Sub

**Sub HighlightLongQuotesDouble()**

**'** Paul Beverley - Version 11.08.11

' Highlight all long quotes (double)

' Alt-Ctrl-Num-9

myColour = wdBrightGreen

wordLimit = 50

Set rng = ActiveDocument.Content

Do

stopNow = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8220)

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

If rng.Find.Found = True Then

quoteStart = rng.Start

rng.Start = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8221)

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

' Highlight it

rng.Start = quoteStart

myText = rng

myText = Replace(myText, " ", " ")

myText = Replace(myText, ChrW(8211) & " ", "")

myText = Replace(myText, " -", "")

myText = Replace(myText, ChrW(8212), " ")

myLen = Len(myText) - Len(Replace(myText, " ", "")) + 1

If myLen >= wordLimit Then

rng.HighlightColorIndex = myColour

End If

Else

stopNow = True

End If

rng.Start = rng.End

Loop Until stopNow = True

Beep

End Sub

**Sub HighlightLongQuotesSingle()**

**'** Paul Beverley - Version 11.08.11

' Highlight all long quotes (single)

myColour = wdPink

wordLimit = 60

Set rng = ActiveDocument.Content

Do

stopNow = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8216)

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

If rng.Find.Found = True Then

quoteStart = rng.Start

rng.Start = rng.End

Do

stopNow = False

gotOne = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8217)

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

If rng.Find.Found = True Then

' You've found a close quote/apostrophe

rng.Start = rng.Start - 1

rng.End = rng.Start + 1

ch1 = rng

rng.Start = rng.Start + 2

rng.End = rng.Start + 1

ch2 = rng

rng.Start = rng.End

gotOne = True

If LCase(ch2) <> UCase(ch2) Then gotOne = False

' i.e. there's a letter after the apostrophe

If gotOne = True And ch1 = "s" Then

' This could be an s-apostrophe, so test it

hereNow = rng.Start

With rng.Find

.Text = ChrW(8216)

.Wrap = False

.Forward = True

.MatchWildcards = False

.Execute

End With

nextOpen = rng.Start

rng.Start = hereNow

rng.End = hereNow

With rng.Find

.Text = ChrW(8217) & "[!a-z]"

.Wrap = False

.Forward = True

.MatchWildcards = True

.Execute

End With

nextClosed = rng.Start

rng.Start = hereNow

rng.End = hereNow

If nextOpen > nextClosed Then gotOne = False

End If

If gotOne = True Then

' Highlight it

rng.Start = quoteStart

rng.End = rng.End - 1

myText = rng

myText = Replace(myText, " ", " ")

myText = Replace(myText, ChrW(8211) & " ", "")

myText = Replace(myText, " -", "")

myText = Replace(myText, ChrW(8212), " ")

myLen = Len(myText) - Len(Replace(myText, " ", "")) + 1

If myLen >= wordLimit Then

rng.HighlightColorIndex = myColour

End If

End If

Else

stopNow = True

End If

Loop Until stopNow = True Or gotOne = True

Else

stopNow = True

End If

rng.Start = rng.End

Loop Until stopNow = True

Set rng = ActiveDocument.Content

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "s" & ChrW(8217) & "^?"

.Highlight = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

gotOne = rng.Find.Found

foundColour = rng.HighlightColorIndex

If foundColour = myColour Then rng.HighlightColorIndex = 0

rng.Start = rng.End

Loop Until gotOne = False

Set rng = ActiveDocument.Content

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^?" & ChrW(8216) & "^?"

.Highlight = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

gotOne = rng.Find.Found

foundColour = rng.HighlightColorIndex

If foundColour = myColour Then rng.HighlightColorIndex = 0

rng.Start = rng.End

Loop Until gotOne = False

Beep

End Sub

**Sub DisplayQuote()**

**'** Paul Beverley - Version 14.12.23

' Finds or displays a long quote

displayedQuoteStyle = "Quote"

' nextParaStyle = "Body No Indent"

nextParaStyle = "Normal"

' Or if you don't want to change next paragraph style

' nextParaStyle = ""

removeQuotes = True

minWords = 20

singleQuotes = False

addTags = False

startTag = "<DQ>"

endTag = "<\DQ>"

tagOnNewLine = False

If addTags = False Then endTag = ""

If tagOnNewLine = True Then

endText = vbCr & endTag

Else

endText = endTag

End If

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If singleQuotes = True Then

openQ = ChrW(8216)

closeQ = ChrW(8217)

Else

openQ = ChrW(8220)

closeQ = ChrW(8221)

End If

If Selection.Start = Selection.End Then

Do

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = openQ

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

If Selection.Find.Found = False Then

Selection.Collapse wdCollapseEnd

Beep

Exit Sub

End If

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

paraEnd = rng.End

rng.Start = Selection.Start

rng.End = Selection.Start

foundEnd = False

wdCount = 1

Do While rng.End < paraEnd And foundEnd = False

rng.MoveEnd wdWord, 1

wdCount = wdCount + 1

DoEvents

If Len(rng.Text) > 20 Then Debug.Print Right(rng, 20)

If InStr(Right(rng, 3), closeQ) > 0 Then foundEnd = True

Loop

Loop Until wdCount > minWords

rng.Select

If foundEnd = False Then

rng.Collapse wdCollapseStart

rng.Start = rng.Start + 1

ActiveDocument.ActiveWindow.LargeScroll down:=1

rng.Select

ActiveDocument.ActiveWindow.SmallScroll down:=1

Beep

Exit Sub

End If

End If

' Delete redundant space

qtStart = Selection.Start

Set rng = ActiveDocument.Range(qtStart - 1, qtStart)

If rng.Text = " " Then

rng.Delete

Selection.Start = qtStart - 1

End If

' Add newline unless it's already a para start.

qtStart = Selection.Start

Set rng = ActiveDocument.Range(qtStart - 1, qtStart)

If rng.Text <> vbCr Then

Selection.InsertBefore vbCr

Selection.MoveStart , 1

End If

' Remove open quote if necessary

qtStart = Selection.Start

Set rng = ActiveDocument.Range(qtStart, qtStart + 1)

If rng.Text = openQ And removeQuotes = True Then

Selection.MoveStart , 1

rng.Delete

End If

' Add start tag

qtStart = Selection.Start

Set rng = ActiveDocument.Range(qtStart, qtStart + 1)

If addTags = True Then Selection.InsertBefore startTag

' Selection.Expand wdParagraph

' Delete redundant space

qtEnd = Selection.End

Set rng = ActiveDocument.Range(qtEnd - 1, qtEnd)

spaceRemoved = False

If rng.Text = " " Then

Selection.MoveEnd , -1

rng.Delete

spaceRemoved = True

End If

' Remove close quote if necessary

qtEnd = Selection.End

Set rng = ActiveDocument.Range(qtEnd - 2, qtEnd)

Debug.Print rng.Text

If InStr(rng.Text, closeQ) > 0 And removeQuotes = True Then

rng.Text = Replace(rng.Text, closeQ, "")

Selection.MoveEnd , 1

End If

If spaceRemoved = True Then endText = endText & vbCr

Selection.InsertAfter endText

Selection.Expand wdParagraph

Selection.Range.Style = displayedQuoteStyle

Selection.Collapse wdCollapseEnd

If nextParaStyle > "" Then

Selection.Expand wdParagraph

Selection.Range.Style = nextParaStyle

Selection.Collapse wdCollapseEnd

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub HighlightIndentedParas()**

**'** Paul Beverley - Version 01.03.11

' Highlight all indented paragraphs

myColour = wdBrightGreen

' myStyleName = "Displayed Quote"

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

For Each myPara In ActiveDocument.Paragraphs

If myPara.Range.ParagraphFormat.LeftIndent > 0 Then

myPara.Range.HighlightColorIndex = myColour

' myPara.Range.Style = ActiveDocument.Styles(myStyleName)

End If

Next

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub** **FirstLineIndentToTab()**

' Paul Beverley - Version 07.09.20

' Changes all first-line indents to a tab

tabSize = 0.25

useInches = True

' For cm, use, e.g.:

' tabSize = 0.8

' useInches = False

For Each myPara In ActiveDocument.Paragraphs

If myPara.Range.ParagraphFormat.FirstLineIndent > 0 And \_

myPara.Range.Characters.Count > 1 Then

With myPara.Range

.Select

.InsertBefore Text:=vbTab

.ParagraphFormat.FirstLineIndent = 0

If useInches Then

' In inches

.ParagraphFormat.TabStops.Add Position:=InchesToPoints(tabSize) \_

, Alignment:=wdAlignTabLeft

.ParagraphFormat.TabStops(InchesToPoints(0.1)).Clear

Else

' In cm

.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(tabSize) \_

, Alignment:=wdAlignTabLeft

.ParagraphFormat.TabStops(CentimetersToPoints(0.3)).Clear

End If

End With

End If

Next

Beep

End Sub

**Sub StyleBodyIndent()**

' Paul Beverley - Version 20.01.17

' Adds body style generally, plus No indent after headings

bodyStyle = "Body text"

indentStyle = "No indent"

noIndent = False

For Each myPara In ActiveDocument.Paragraphs

If InStr(myPara.range.Style, "Heading") > 0 Then

noIndent = True

Else

If noIndent = False Then

myPara.range.Style = bodyStyle

Else

myPara.range.Style = indentStyle

noIndent = False

End If

End If

Next myPara

End Sub

**Sub IndentChanger()**

**'** Paul Beverley - Version 07.07.11

' Changes paras of one indent value to another

wasIndent = 1.25

newIndent = 0

wasPoints = CentimetersToPoints(wasIndent)

For Each myPara In ActiveDocument.Paragraphs

Set rng = myPara.Range

rng.Select

StatusBar = (rng.ParagraphFormat.LeftIndent)

myGuess = rng.ParagraphFormat.LeftIndent

If myGuess < wasPoints + 1 And myGuess > wasPoints - 1 Then

rng.ParagraphFormat.LeftIndent = CentimetersToPoints(newIndent)

End If

Next myPara

Beep

End Sub

**Sub** **FormatHeadwords()**

' Paul Beverley - Version 22.11.20

' Adds a character style to the first word of every para in a given style

If Selection.End = Selection.Start Then

myResponse = MsgBox("Do this to the WHOLE file?", \_

vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

Set rng = ActiveDocument.Content

Else

Set rng = Selection.Range.Duplicate

End If

For Each myPara In rng.Paragraphs

If Len(myPara.Range) > 1 Then

If myPara.Range.Style = ActiveDocument.Styles("Normal,Normal full left") Then

' myPara.Range.Words(1).Font.Bold = True

' myPara.Range.Words(1).Style = ActiveDocument.Styles("EntryBib")

myPara.Range.Words(1).Style = ActiveDocument.Styles("Emphasis")

End If

End If

DoEvents

Next myPara

Beep

End Sub

**Sub CountHighlightColour()**

**'** Paul Beverley - Version 04.03.11

' Count how many times a highlight colour occurs

selColour = Selection.Range.HighlightColorIndex

hereNow = Selection.Start

If selColour = 0 Then

MsgBox ("Select colour to be counted")

Exit Sub

End If

i = 0

Set rng = ActiveDocument.Content

theEnd = rng.End

Do

If rng.Start > 0 Then rng.Start = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

If rng.Find.Found = True Then

If rng.HighlightColorIndex = selColour Then i = i + 1

If i Mod 10 = 0 Then StatusBar = "To go: " \_

& Str(Int((theEnd - rng.Start) / 100))

End If

Loop Until rng.End = rng.Start Or rng.End = ActiveDocument.Range.End

StatusBar = ""

MsgBox ("Found: " & Str(i))

Selection.Start = hereNow

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = oldFind

.Replacement.Text = oldReplace

.Wrap = wdFindContinue

End With

End Sub

**Sub BodyTexter()**

**'** Paul Beverley - Version 02.02.11

' Apply 'Body Text' to every paragraph in 'Normal'

fromStyle = "Normal"

toStyle = "Body Text"

' Or use two other styles:

' fromStyle = "Caption"

' toStyle = "Figure Heading"

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

i = ActiveDocument.Paragraphs.Count

For Each para In ActiveDocument.Paragraphs

thisStyle = para.Style

If thisStyle = ActiveDocument.Styles(fromStyle) Then

para.Style = ActiveDocument.Styles(toStyle)

End If

i = i - 1

StatusBar = "Paragraphs to go: " & Str(i)

Next para

Beep

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub CodeBoldParas()**

**'** Paul Beverley - Version 13.01.11

' Tag/code every bold heading

myCode1 = "<B>"

myCode2 = "<A>"

' If you don't want to change the numbered headings

' to code 2, use myCode2 = ""

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

For Each para In ActiveDocument.Paragraphs

Set rng = para.Range

rng.End = rng.End - 1

If rng.Font.Bold = True Then

rng.End = rng.Start

rng.InsertAfter myCode1

rng.Select

End If

Next para

' And take out any examples of the code

' being applied to a blank line

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myCode1 & "^p"

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Make sure there are no spaces at the beginnings

' of coded lines

With Selection.Find

.Replacement.ClearFormatting

.ClearFormatting

.Text = myCode1 & " "

.Replacement.Text = myCode1

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Optionally change the coding of any heading

' that starts with a number

If myCode2 > "" Then

mCod1 = Replace(myCode1, "<", "\<")

mCod1 = Replace(mCod1, ">", "\>")

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Text = mCod1 + "([0-9])"

.Replacement.Text = myCode2 & "\1"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

' Now clear up

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchCase = False

End With

ActiveDocument.TrackRevisions = nowTrack

End Sub

**Sub AutoTagger()**

' Paul Beverley - Version 30.09.19

' Tag/code automatically

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

For Each para In ActiveDocument.Paragraphs

Set rng = para.range.Duplicate

startText = "": endText = ""

styleTitle = rng.Style

Select Case styleTitle

' List your styles and tags here

' ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Case "Heading 1"

startText = "<A>"

Case "Heading 2"

startText = "<B>"

startText = "<CH>"

Case "Heading 3"

startText = "<C>"

startText = "<Part>"

Case "Definition"

startText = "<DF>": endText = "</DF>"

Case "Caption"

startText = "<Cap>"

Case "citation"

startText = "<Cap>"

Case "Table3"

startText = "<Cap>"

Case "Level A"

startText = "<A>"

End Select

' For any "funny" styles, select part of the name

' ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

If InStr(styleTitle, "eading4") > 0 Then startText = "<A>"

If InStr(styleTitle, "evel4") > 0 Then startText = "<A>"

If InStr(styleTitle, "eading 4") > 0 Then startText = "<A>"

If InStr(styleTitle, "evel A") > 0 Then startText = "<A>"

If InStr(styleTitle, "Heading 5") > 0 Then startText = "<B>"

If InStr(styleTitle, "Table") > 0 Then startText = "<Cap>"

If rng.Characters(1) <> "<" And startText > "" And \_

rng.Information(wdWithInTable) = False And \_

Len(rng) > 3 Then

rng.InsertBefore startText

rng.End = rng.End - 1

rng.InsertAfter endText

End If

Next para

ActiveDocument.TrackRevisions = myTrack

Beep

End Sub

**Sub TableSpaceBeforeHeading()**

' Paul Beverley - Version 04.11.17

' Adds a blank line where a numbered heading follows a table

For Each myTable In ActiveDocument.Tables

Set rng = myTable.range.Duplicate

rng.Collapse wdCollapseEnd

rng.MoveEnd , 1

If Val(rng.Text) > 0 Then rng.InsertBefore Text:=vbCr

rng.Select

Next myTable

Selection.HomeKey Unit:=wdStory

End Sub

**Sub CodeIndentedParas()**

**'** Paul Beverley - Version 07.07.11

' Add codes to indented paragraphs

beforeText = "<disp>"

afterText = "</disp>"

For Each para In ActiveDocument.Paragraphs

Set rng = para.Range

If rng.ParagraphFormat.LeftIndent <> CentimetersToPoints(0) \_

And Len(rng) > 10 Then

rng.Select

Selection.InsertBefore Text:=beforeText

Selection.Start = Selection.End - 1

Selection.End = Selection.Start

Selection.TypeText Text:=afterText

End If

Next para

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = afterText & "^p" & beforeText

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub TagBulletLists()**

' Paul Beverley - Version 30.04.12

' Add tags to all bullet lists

For Each myPara In ActiveDocument.Paragraphs

If myPara.FirstLineIndent < 0 Then

myPara.Range.InsertBefore "<BL>"

myPara.Range.InsertAfter "</BL>"

End If

Next myPara

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<BL></BL>"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub ListItemNumberFormatter()**

' Paul Beverley - Version 24.10.16

' Format the numbering of current list item

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

If Selection = " " Then Selection.Delete

If Selection = "(" Then

Selection.Delete

End If

Selection.MoveEnd , 4

parenPos = InStr(Selection, ")")

Selection.Collapse wdCollapseStart

If parenPos > 0 Then

Selection.MoveStartUntil cset:=")"

Selection.MoveEnd , 1

Selection.TypeText Text:="."

End If

Selection.Expand wdParagraph

ActiveDocument.TrackRevisions = myTrack

Selection.Collapse wdCollapseEnd

End Sub

**Sub VerseListFormat()**

' Paul Beverley - Version 09.11.18

' Formats list(s) or poem verse(s) with manual linebreaks

doOneVerse = (Selection.Start = Selection.End)

If doOneVerse Then

keepGoing = True

Set rng = Selection.range.Duplicate

Do While keepGoing = True

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Wrap = False

.Replacement.Text = "^11"

.Forward = True

.MatchWildcards = False

.Execute

End With

rng.Select

Selection.MoveRight , 1

If Asc(Selection) = 13 Then

keepGoing = False

Else

rng.Text = Chr(11)

rng.Start = rng.End + 1

End If

DoEvents

Loop

Selection.MoveEnd , 1

Selection.Delete

Else

Set rng = Selection.range.Duplicate

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

If Len(Selection) > 1 Then

Selection.Collapse wdCollapseEnd

Selection.MoveRight , 1

End If

Selection.Collapse wdCollapseEnd

rng.End = Selection.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p^p"

.Wrap = False

.Replacement.Text = "zczc"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = "^p"

.Wrap = False

.Replacement.Text = "^11"

.Execute Replace:=wdReplaceAll

.Text = "zczc"

.Wrap = False

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

End If

End Sub

**Sub ItalicBoldTagger()**

' Paul Beverley - Version 02.05.16

' Adds red tags to all italic and/or bold text

doItalic = True

doBold = True

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

myCount = 0

' Go and find the first italic bit

If doItalic = True Then

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While Selection.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the start of the found item is

startNow = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:="</i>"

endNow = Selection.End

Selection.MoveStart , -4

Selection.Font.Color = wdColorRed

Selection.End = startNow

Selection.TypeText Text:="<i>"

Selection.MoveStart , -3

Selection.Font.Color = wdColorRed

Selection.Start = endNow + 3

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

myItalic = myCount

End If

' Go and find the first bold bit

If doBold = True Then

Selection.HomeKey Unit:=wdStory

myCount = 0

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Bold = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While Selection.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the start of the found item is

startNow = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:="</b>"

endNow = Selection.End

Selection.MoveStart , -4

Selection.Font.Color = wdColorRed

Selection.End = startNow

Selection.TypeText Text:="<b>"

Selection.MoveStart , -3

Selection.Font.Color = wdColorRed

Selection.Start = endNow + 3

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

myBold = myCount

End If

myMsg = "Changed: "

If doItalic = True Then myMsg = myMsg & myItalic & " Italic" & vbCr

If doBold = True Then myMsg = myMsg & myBold & " Bold"

MsgBox myMsg

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub TagVariousAttributes()**

' Paul Beverley - Version 13.07.17

' Add red tags to all italic text

myColour = wdColorRed

' for no colour, use myColour = wdColorBlack

doItalic = True

italON = "<em>"

doBold = True

boldON = "<strong>"

doSBSS = True

sbON = "<sub>"

ssON = "<sup>"

doUL = True

ulON = "<u>"

doStrike = True

strikeON = "<del>"

offMarker = "/"

' i.e. the italic markers will be <em> and </em>

italOFF = Replace(italON, "<", "<" & offMarker)

boldOFF = Replace(boldON, "<", "<" & offMarker)

ssOFF = Replace(ssON, "<", "<" & offMarker)

sbOFF = Replace(sbON, "<", "<" & offMarker)

ulOFF = Replace(ulON, "<", "<" & offMarker)

strikeOFF = Replace(strikeON, "<", "<" & offMarker)

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

myCount = 0

' Go and find the first italic bit

If doItalic = True Then

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While Selection.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the start of the found item is

startNow = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=italOFF

endNow = Selection.End

Selection.MoveStart , -Len(italOFF)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.End = startNow

Selection.TypeText Text:=italON

Selection.MoveStart , -Len(italON)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.Start = endNow + Len(italON)

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

myItalic = myCount

End If

myCount = 0

' Go and find the first bold bit

If doBold = True Then

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Bold = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While Selection.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the start of the found item is

startNow = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=boldOFF

endNow = Selection.End

Selection.MoveStart , -Len(boldOFF)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.End = startNow

Selection.TypeText Text:=boldON

Selection.MoveStart , -Len(boldON)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.Start = endNow + Len(boldON)

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

myBold = myCount

End If

myCount = 0

' Go and find the first underline bit

If doUL = True Then

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While Selection.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the start of the found item is

startNow = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=ulOFF

endNow = Selection.End

Selection.MoveStart , -Len(ulOFF)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.End = startNow

Selection.TypeText Text:=ulON

Selection.MoveStart , -Len(ulON)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.Start = endNow + Len(ulON)

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

myUL = myCount

End If

myCount = 0

' Go and find the first StrikeThrough bit

If doStrike = True Then

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While Selection.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the start of the found item is

startNow = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=strikeOFF

endNow = Selection.End

Selection.MoveStart , -Len(strikeOFF)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.End = startNow

Selection.TypeText Text:=strikeON

Selection.MoveStart , -Len(strikeON)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.Start = endNow + Len(strikeON)

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

myStrike = myCount

End If

myCount = 0

' Go and find the first subscript bit

If doSBSS = True Then

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Subscript = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While Selection.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the start of the found item is

startNow = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=sbOFF

endNow = Selection.End

Selection.MoveStart , -Len(sbOFF)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.Font.Subscript = False

Selection.End = startNow

Selection.TypeText Text:=sbON

Selection.MoveStart , -Len(sbON)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.Start = endNow + Len(sbON)

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

mySB = myCount

' Go and find the first superscript bit

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Superscript = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While Selection.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the start of the found item is

startNow = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=ssOFF

endNow = Selection.End

Selection.MoveStart , -Len(ssOFF)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.Font.Superscript = False

Selection.End = startNow

Selection.TypeText Text:=ssON

Selection.MoveStart , -Len(ssON)

If myColour > 0 Then Selection.Font.Color = myColour

Selection.Start = endNow + Len(ssON)

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

mySS = myCount

End If

myMsg = "Changed:" & vbCr

If doItalic = True Then myMsg = myMsg & myItalic & " Italic" & vbCr

If doBold = True Then myMsg = myMsg & myBold & " Bold" & vbCr

If doSBSS = True Then

myMsg = myMsg & mySB & " Subscript" & vbCr

myMsg = myMsg & mySS & " Superscript" & vbCr

End If

If doUL = True Then myMsg = myMsg & myUL & " Underline" & vbCr

If doStrike = True Then myMsg = myMsg & myStrike & " StrikeThrough" & vbCr

MsgBox myMsg

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub TagSelectedOrBold()**

' Paul Beverley - Version 02.05.16

' Add red tags to the currently selected text or italic text

tagTextSelected = "<xx></xx>"

tagTextBold = "<b></b>"

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If Selection.Start = Selection.End Then

tagStop = InStr(tagTextBold, "><")

startTag = Left(tagTextBold, tagStop)

endTag = Mid(tagTextBold, tagStop + 1)

' Go and find the first occurrence

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Bold = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Else

startTag = Left(tagTextSelected, InStr(tagTextSelected, "><"))

endTag = Mid(tagTextSelected, InStr(tagTextSelected, "><") + 1)

End If

startNow = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=endTag

endNow = Selection.End

Selection.MoveStart , -(Len(endTag))

' Here's where the colour is added

Selection.Font.Color = wdColorRed

Selection.End = startNow

Selection.TypeText Text:=startTag

Selection.MoveStart , -(Len(startTag))

' And here

Selection.Font.Color = wdColorRed

Selection.Start = endNow + Len(startTag)

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub TagNI()**

' Paul Beverley - Version 04.11.17

' Adds an <ni> tag after every heading

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\<[ABC]\>"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While Selection.Find.Found = True

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:="<ni>"

Selection.MoveEnd , 1

If Asc(Selection) = 13 Then Selection.Delete

Selection.Collapse wdCollapseEnd

Selection.Find.Execute

Loop

ActiveDocument.TrackRevisions = myTrack

Selection.HomeKey Unit:=wdStory

End Sub

**Sub ListOfTaggedHeadings()**

' Paul Beverley - Version 21.09.18

' Lists all tagged headings, <A>, <B>, etc

withFormatting = True

Set nowDoc = ActiveDocument

Set rng = ActiveDocument.Content

Documents.Add

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\<[A-E]\>"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rng.Expand wdParagraph

Selection.EndKey Unit:=wdStory

If withFormatting = True Then

Selection.FormattedText = rng.FormattedText

Else

Selection.Text = rng.Text

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

nowDoc.Activate

End Sub

**Sub FullPointOnCaptions()**

' Paul Beverley - Version 09.06.17

' Finds captions and ensures that they have a full point

myTag = "<Cap>"

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myTag

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

rng.Expand wdParagraph

lastChar = Left(Right(rng.Text, 2), 1)

If lastChar = "." Or lastChar = "?" Or lastChar = "!" Then

' No nothing!

Else

rng.Start = rng.End - 1

rng.Select

rng.InsertBefore Text:="."

End If

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

rng.Find.Execute

Loop

End Sub

**Sub HeadingStyler()**

**'** Paul Beverley - Version 02.02.11

' Styles all headings by depth of section number

delimiter = Chr(9)

' delimiter = " "

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

For Each para In ActiveDocument.Paragraphs

myText = para

tabPos = InStr(myText, delimiter) - 1

If tabPos > 2 Then

myText = Left(myText, tabPos)

' this should now hold just the heading number, e.g. 2.4.5.1

' but let's just check there are no alpha characters in there

If LCase(myText) = UCase(myText) Then

noPeriods = Replace(myText, ".", "")

headLevel = Len(myText) - Len(noPeriods)

' No of periods = heading level

Select Case headLevel

Case 1: para.Style = ActiveDocument.Styles("Heading 1")

Case 2: para.Style = ActiveDocument.Styles("Heading 2")

Case 3: para.Style = ActiveDocument.Styles("Heading 3")

Case 4: para.Style = ActiveDocument.Styles("Heading 4")

Case 5: para.Style = ActiveDocument.Styles("Heading 5")

Case 6: para.Style = ActiveDocument.Styles("Heading 6")

Case Else

para.Range.Select

Beep

Exit Sub

End Select

End If

End If

Next para

Beep

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub CodeFirstLines()**

' Paul Beverley - Version 05.12.12

' Prepares the file by adding codes, etc.

code1 = "<CN>"

code2 = "<CH>"

code3 = "<CA>"

code4 = "<A>"

myLine = 1

For Each myPara In ActiveDocument.Paragraphs

If Len(myPara.Range) > 1 Then

If myLine = 4 Then

myPara.Range.InsertBefore Text:=code4

Exit For

End If

If myLine = 3 Then

myPara.Range.InsertBefore Text:=code3

myLine = 4

End If

If myLine = 2 Then

myPara.Range.InsertBefore Text:=code2

myLine = 3

End If

If myLine = 1 Then

myPara.Range.InsertBefore Text:=code1

myLine = 2

End If

End If

Next myPara

' Set 1.15 spacing

Set rng = ActiveDocument.Content

rng.ParagraphFormat.LineSpacing = ActiveDocument.Styles(wdStyleNormal).Font.Size \* 1.15

' Set language UK

ActiveDocument.TrackRevisions = False

Set rng = ActiveDocument.Content

rng.LanguageID = wdEnglishUK

rng.NoProofing = False

ActiveDocument.Styles(wdStyleNormal).LanguageID = wdEnglishUK

' Switch on track changes

ActiveDocument.TrackRevisions = True

End Sub

**Sub ShowStyles()**

' Paul Beverley - Version 06.05.12

' Show style names as <A>-type codes in the text

noShow = ",N,Normal,TOC 1,TOC 2,TOC 3,,,,"

noShow = noShow & ",Table of Figures,P1,,,"

abbrvs = ",MTDisplayEquation,Disp,Heading 1,A,Heading 2,B,,,"

abbrvs = abbrvs & ",Heading 3,C,Heading 4,D,Normal,N,"

removePads = False

doTables = True

' Merged cells generate an error, so ignore it

' and carry on regardless!

If doTables = True Then On Error Resume Next

' Find return <|

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<["

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

gotCode = rng.Find.Found

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

'If the codes are there, remove them

If gotCode Then

Set rng = ActiveDocument.Content

With rng.Find

.Text = "\<\[\*\]\>"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Else

' If no codes then add codes

i = ActiveDocument.Paragraphs.Count

abbrvs = abbrvs & ","

For Each myPara In ActiveDocument.Paragraphs

thisStyle = myPara.Style

typeIt = True

If InStr(noShow, "," & thisStyle & ",") > 0 Then typeIt = False

If doTables = False And myPara.Range.Information(wdWithInTable) = True Then typeIt = False

If typeIt = True Then

myPos = InStr(abbrvs, thisStyle)

If myPos > 0 Then

thisStyle = Mid(abbrvs, myPos + Len(thisStyle) + 1)

thisStyle = Left(thisStyle, InStr(thisStyle, ",") - 1)

End If

myPara.Range.InsertBefore Text:="<[" & thisStyle & "]>"

End If

i = i - 1

StatusBar = "Paragraphs to go: " & Str(i)

Next myPara

End If

If removePads = True And gotCode = False Then

' Check whether the user wants to remove the pads

myResponse = MsgBox("Remove vertical bars?", vbQuestion + vbYesNo)

If myResponse = vbYes Then

Set rng = ActiveDocument.Content

With rng.Find

.Text = "<["

.Wrap = wdFindContinue

.Replacement.Text = "<"

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.Text = "]>"

.Wrap = wdFindContinue

.Replacement.Text = ">"

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

End If

StatusBar = ""

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub NumberSequenceCheckerSimple()**

' Paul Beverley - Version 24.09.18

' Checks consecutivity of numbering

dateStart = 1800

Selection.MoveStartUntil cset:="0123456789", Count:=wdForward

Selection.MoveEndWhile cset:="0123456789", Count:=wdForward

myNumWas = Val(Selection)

Selection.Collapse wdCollapseEnd

Do

Selection.MoveStartUntil cset:="0123456789", Count:=wdForward

Selection.MoveEndWhile cset:="0123456789", Count:=wdForward

myNum = Val(Selection)

DoEvents

If (myNum <> myNumWas + 1) And (myNum < dateStart) Then

Beep

Exit Sub

End If

Selection.Collapse wdCollapseEnd

If myNum < dateStart Then myNumWas = myNum

Loop Until 0

End Sub

**Sub NumberSequenceCheckerDecimal()**

' Paul Beverley - Version 24.09.18

' Checks consecutivity of numbering containing decimal point

tooFar = 3000

Do

Selection.MoveStartUntil cset:="0123456789", Count:=wdForward

Selection.MoveEndWhile cset:="0123456789.", Count:=wdForward

dotPos = InStr(Selection, ".")

DoEvents

If dotPos = 0 Then Selection.Collapse wdCollapseEnd

Loop Until dotPos > 0

myText = Selection

dotPos = InStr(Selection, ".")

chNumWas = Val(Left(myText, dotPos - 1))

myNumWas = Val(Mid(myText, dotPos + 1))

Set wasRange = Selection.range

Selection.Collapse wdCollapseEnd

Do

Do

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^#"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

DoEvents

If Not (Selection.Find.Found) Or \_

Selection.Start > wasRange.Start + tooFar Then

wasRange.Select

ActiveWindow.ScrollIntoView wasRange, True

Selection.MoveStartWhile cset:="0123456789.", Count:=wdBackward

Beep

Exit Sub

End If

Selection.MoveEndWhile cset:="0123456789.", Count:=wdForward

DoEvents

dotPos = InStr(Selection, ".")

If dotPos = 0 Then Selection.Collapse wdCollapseEnd

Loop Until dotPos > 0

myText = Selection

chNum = Val(Left(myText, dotPos - 1))

myNum = Val(Mid(myText, dotPos + 1))

If (myNum = myNumWas + 1 And chNum = chNumWas) Or \_

(chNum = chNumWas + 1 And myNum = 1) Then

' carry on!

Else

If dotPos < Len(myText) Then

Beep

ActiveWindow.ScrollIntoView Selection.range, True

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

End If

Selection.Collapse wdCollapseEnd

If dotPos < Len(myText) Then

myNumWas = myNum

chNumWas = chNum

End If

Set wasRange = Selection.range

ActiveWindow.ScrollIntoView wasRange, True

Loop Until 0

End Sub

**Sub NumberSequenceCheckerHierarchical()**

**'** Paul Beverley - Version 06.02.12

' Check the sequence of section numbers

allowSingleNumbers = False

highlightError = True

captionWordsMax = 80

endAtLastRightItem = True

showProgress = True

Dim numText() As String

Dim num(9) As Integer

' Select line to be the model

If Selection.Start = Selection.End Then

Selection.Paragraphs(1).Range.Select

End If

If Len(Selection) < 10 Then

myResponse = MsgBox("Is this the starting line?", vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

End If

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

' Look at the current heading line

Set rng = Selection.Range

If Asc(rng.Text) < 32 Then rng.MoveStart 1

rng.MoveEnd , -1

Selection.Collapse

lineText = rng.Text

' See if there's any text before the number

Set rng2 = rng.Duplicate

rng2.MoveStartUntil cset:="123456789", Count:=wdForward

posStart = InStr(rng.Text, Left(rng2.Text, 3)) - 1

leftText = Left(lineText, posStart)

' Find what's after the numbers: tab? space?

rng2.MoveStartWhile cset:="1234567890.", Count:=wdForward

posEnd = InStr(rng.Text, Left(rng2.Text, 3)) - 1

stopper = Mid(lineText, posEnd + 1, 1)

If Asc(stopper) = 9 Then stopper = "^t"

If AscW(stopper) = 8195 Then stopper = ChrW(8195)

If stopper = ")" Then stopper = "\)"

If stopper = "]" Then stopper = "\]"

startNum = Mid(lineText, posStart + 1, posEnd - posStart)

If Right(startNum, 1) = "." Then startNum = Left(startNum, Len(startNum) - 1)

' Analyse the number, splitting it into sections

numText = Split(startNum, ".")

startDepth = UBound(numText) + 1

For i = 1 To startDepth

num(i) = numText(i - 1)

Next i

Set rng = rng2.Duplicate

leftTextSearch = Replace(leftText, "(", "\(")

leftTextSearch = Replace(leftTextSearch, "[", "\[")

Do

Set rng2 = rng.Duplicate

If allowSingleNumbers = False Then

findThis = leftTextSearch & "[0-9.]{3,}" & stopper

Else

findThis = leftTextSearch & "[0-9.]@" & stopper

End If

Do

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = findThis

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Set rng3 = rng.Duplicate

rng3.MoveStart , -1

If showProgress = True Then rng.Select

rng3.End = rng3.Start + 1

If rng3.Text > "" Then crTest = Asc(rng3.Text)

rng3.MoveEnd wdParagraph, 1

' Check if this really is likely to be a heading/caption

If rng.Find.Found = False Then

lookForAnother = False

Else

If (rng3.Words.Count) < captionWordsMax And (crTest = 12 \_

Or crTest = 13) And Right(newNum, 1) <> "." Then

lookForAnother = False

If InStr(rng.Text, ".") = 0 And allowSingleNumbers = False \_

Then lookForAnother = True

Else

lookForAnother = True

End If

End If

Loop Until lookForAnother = False

If rng.Find.Found = False Then

rng.Move , 1

rng.Select

Beep

endAtLastRightItem = False

GoTo theEnd

End If

rng.MoveEnd , -1

If showProgress = True Then rng.Select

newNum = rng.Text

newNum = Replace(newNum, leftText, "")

If Right(newNum, 1) = "." Then newNum = Left(newNum, Len(newNum) - 1)

keepGoing = True

newDepth = Len(newNum) - Len(Replace(newNum, ".", "")) + 1

depthChange = newDepth - startDepth

Select Case (depthChange)

Case Is > 1

' This must be an error!

keepGoing = False

Case 1:

' Test for, e.g. 3.4.5.6 -> 3.4.5.6.1

If newNum <> startNum & ".1" Then keepGoing = False

startDepth = startDepth + 1

num(startDepth) = 1

Case 0:

' Test for, e.g. 3.4.5.6 -> 3.4.5.7

keepGoing = False

num(startDepth) = num(startDepth) + 1

testNum = ""

For i = 1 To startDepth

testNum = testNum & Trim(Str(num(i))) & "."

Next i

If Replace(testNum, newNum, "") = "." Then

keepGoing = True

Else

If allowSingleNumbers = False Then

If newNum = Trim(Str(num(1) + 1)) & ".1" Then

keepGoing = True

num(1) = num(1) + 1: num(2) = 1

End If

End If

End If

Case Is < 0:

' Test for, e.g. 3.4.5.6 -> 3.4.6 (or 3.5) (or 4)

num(newDepth) = num(newDepth) + 1

testNum = ""

For i = 1 To newDepth

testNum = testNum & Trim(Str(num(i))) & "."

Next i

If Replace(testNum, newNum, "") = "." Then

keepGoing = True

Else

If allowSingleNumbers = False Then

If newNum = Trim(Str(num(1) + 1)) & ".1" Then

keepGoing = True

num(1) = num(1) + 1: num(2) = 1

End If

End If

End If

End Select

startNum = newNum

startDepth = newDepth

Loop Until keepGoing = False

Beep

If highlightError = True Then

rng2.HighlightColorIndex = wdTurquoise

rng.HighlightColorIndex = wdRed

End If

' Second beep to show end

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

theEnd:

If endAtLastRightItem = True Then

rng2.Collapse

rng2.Select

startHere = Selection.Start - 1

Selection.MoveDown Unit:=wdScreen, Count:=2

' Make sure that you've not dropped into a footnote ...

Do While Selection.Information(wdInFootnote) = True

Selection.MoveDown Unit:=wdLine, Count:=2

Loop

Else

rng.Collapse

rng.Select

startHere = Selection.Start - 1

Selection.MoveUp Unit:=wdScreen, Count:=2

' Make sure that you've not dropped into a footnote ...

Do While Selection.Information(wdInFootnote) = True

Selection.MoveUp Unit:=wdLine, Count:=2

Loop

End If

Selection.End = startHere

Selection.MoveRight Unit:=wdCharacter, Count:=1

With Selection.Find

.Text = "^13" & findThis

.MatchWildcards = True

End With

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub ContentsListerByStyle()**

**'** Paul Beverley - Version 23.06.23

' Create a contents list from numbered heading style

bigFontSize = 12

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

For Each myTable In ActiveDocument.Tables

myTable.Delete

Next myTable

For Each myPara In ActiveDocument.Paragraphs

cutIt = True

mySize = myPara.Range.Font.Size

If mySize > bigFontSize And mySize < 100 Then cutIt = False

sfdhlsgk = myPara.Range.Style

myStyle = Left(myPara.Range.Style, 4)

myText = myPara.Range.Text

If Left(myPara.Range.Style, 4) = "Head" Then cutIt = False

' If myPara.Range.Style = "Heading 2" Then cutIt = True

' If myPara.Range.Style = "Heading 3" Then cutIt = True

If cutIt = True Then myPara.Range.Font.Color = wdColorBlue

DoEvents

Next myPara

Beep

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = wdColorBlue

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Beep

End Sub

**Sub ContentsListerByNumber()**

' Paul Beverley - Version 25.09.15

' Create a contents list from numbered headings

Selection.WholeStory

Selection.Copy

Selection.HomeKey Unit:=wdStory

Documents.Add

Selection.Paste

ActiveDocument.ConvertNumbersToText

For Each myTable In ActiveDocument.Tables

myTable.Delete

Next myTable

For Each myPara In ActiveDocument.Paragraphs

fstChar = Left(myPara.range.Text, 1)

If InStr("123456789", fstChar) = 0 Then myPara.range.Delete

Next myPara

For Each myPara In ActiveDocument.Paragraphs

chop = False

myText = myPara.range.Text

allLen = Len(myText)

If allLen - Len(Replace(myText, Chr(9), "")) > 1 Then chop = True

If InStr(myText, "<book>") > 0 Then chop = True

If InStr(myText, "<other>") > 0 Then chop = True

wrdNum = myPara.range.Words.Count

If wrdNum > 40 Then chop = True

If InStr(myText, "<journal>") > 0 Then chop = True

If chop = True Then myPara.range.Delete

Next myPara

Beep

End Sub

**Sub ContentsListerByTag()**

' Paul Beverley - Version 17.09.18

' Creates a contents list from tags, <A>, <B>, etc.

bringMainToFront = True

removeTags = True

myText = "B"

Dim fnd(5) As String

Dim rpl(5) As String

' two levels

fnd(2) = "<CHN>,<CH>,<A>,<B>"

rpl(2) = "^p^p<CHN>^t,<CH>^p,<A>,^t<B>"

' three levels

fnd(3) = "<CHN>,<CH>,<A>,<B>,<C>"

rpl(3) = "^p^p<CHN>^t,<CH>^p,<A>,^t<B>,^t^t<C>"

' four levels

fnd(4) = "<CHN>,<CH>,<A>,<B>,<C>,<D>"

rpl(4) = "^p^p<CHN>^t,<CH>^p,<A>,^t<B>,^t^t<C>,^t^t^t<D>"

' five levels

fnd(5) = "<CHN>,<CH>,<A>,<B>,<C>,<D>,<E>"

rpl(5) = "^p^p<CHN>^t,<CH>^p,<A>,^t<B>,^t^t<C>,^t^t^t<D>,^t^t^t^t<E>"

Do

myResponse = InputBox("Level? (B-E):", "Content Lister", myText)

myLevel = Asc(UCase(myResponse)) - 64

Loop Until myLevel > 1 And myLevel < 6

myColour = wdTurquoise

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

Set thisDoc = ActiveDocument

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

Selection.EndKey unit:=wdStory

myF = Split(fnd(myLevel), ",")

myR = Split(rpl(myLevel), ",")

For i = 0 To UBound(myF)

myFind = Replace(myF(i), "<", "\<")

myFind = Replace(myFind, ">", "\>")

If Right(myR(i), 1) = "#" Then

myCR = "^t"

myR(i) = Replace(myR(i), "#", "")

Else

myCR = "^p"

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind & "(\*)^13"

.Wrap = wdFindContinue

.Replacement.Text = myR(i) & "\1" & myCR

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

StatusBar = " " & myR(i)

Next i

StatusBar = " Editing the list. Please wait..."

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\*"

.Highlight = False

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

If removeTags = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\<\*\>"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

' Pull chapter titles to chapter numbers

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^t^p"

.Replacement.Text = "^t"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = wdNoHighlight

Selection.HomeKey unit:=wdStory

Options.DefaultHighlightColorIndex = oldColour

If bringMainToFront = True Then thisDoc.Activate

Beep

End Sub

**Sub MatchDoubleQuotes()**

' Paul Beverley - Version 07.12.12

' Check whether double quotes match up

For Each myPara In ActiveDocument.Paragraphs

myText = myPara.Range.Text

L = Len(myText)

L1 = Len(Replace(myText, Chr(34), ""))

Lopen = Len(Replace(myText, ChrW(8220), ""))

Lclose = Len(Replace(myText, ChrW(8221), ""))

If (L - L1) Mod 2 <> 0 Or Lopen <> Lclose Then

myPara.Range.Font.Underline = True

myCount = myCount + 1

StatusBar = "Found: " & myCount

End If

Next

StatusBar = ""

If myCount = 0 Then

MsgBox ("All clear!")

Else

MsgBox ("Number of suspect paragraphs: " & Trim(myCount))

End If

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Replacement.Text = ""

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

End Sub

**Sub MatchSingleQuotes()**

' Paul Beverley - Version 18.03.21

' Check whether single quotes match up

Dim myList As String

myList = "'s,s','t,'v,'r,'l,'m,'d,'y,'c" ' UK list

' myList = "'i,'k,'m,'n,'s,'t,'r" ' Dutch list

myColour = wdYellow

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

myList = myList & "," & Replace(myList, "'", ChrW(8217))

myCode = Split(myList, ",")

numCodes = UBound(myCode)

useExplorer = False

For Each myPara In ActiveDocument.Paragraphs

myText = LCase(myPara.Range.Text)

'Strip out all the apostrophe-s and s-apostrophe

For i = 0 To numCodes

myText = Replace(myText, myCode(i), "")

Next i

L = Len(myText)

qts = L - Len(Replace(myText, Chr(39), ""))

opens = L - Len(Replace(myText, ChrW(8216), ""))

closes = L - Len(Replace(myText, ChrW(8217), ""))

If qts Mod 2 <> 0 Or opens <> closes Then

myPara.Range.Font.Underline = True

myCount = myCount + 1

StatusBar = "Found: " & myCount

DoEvents

End If

Next

StatusBar = ""

If myCount = 0 Then

MsgBox ("All clear!")

Else

MsgBox ("Number of suspect paragraphs: " & Trim(myCount))

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "s'"

.Font.Underline = True

.Wrap = wdFindContinue

.Forward = True

.Replacement.Font.StrikeThrough = True

.Replacement.Highlight = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Wrap = wdFindStop

.Forward = True

.MatchWildcards = False

.Execute

DoEvents

End With

Do While rng.Find.Found = True

If rng.Font.StrikeThrough <> 9999999 Then

rng.HighlightColorIndex = wdYellow

End If

rng.Font.StrikeThrough = False

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Replacement.Text = ""

.Execute

End With

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.MoveRight , 1

ActiveDocument.TrackRevisions = myTrack

Options.DefaultHighlightColorIndex = oldColour

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub QuoteMarkEmbedder()**

' Paul Beverley - Version 04.11.17

' Changes double quotes inside double quotes to singles

For Each myPara In ActiveDocument.Paragraphs

If InStr(myPara.range.Text, ChrW(8220)) > 0 Or \_

InStr(myPara.range.Text, ChrW(8216)) > 0 Then

dbl = True

sgl = True

For Each ch In myPara.range.Characters

Select Case ch

Case ChrW(8220)

dbl = Not (dbl)

If dbl = True Then

ch.Text = ChrW(8216)

ch.HighlightColorIndex = wdYellow

End If

Case ChrW(8221)

dbl = Not (dbl)

If dbl = False Then

ch.Text = ChrW(8217)

ch.HighlightColorIndex = wdGray25

End If

Case ChrW(8216): sgl = Not (sgl)

Case ChrW(8217): sgl = Not (sgl)

End Select

Next ch

End If

Next myPara

End Sub

**Sub AddSectionNumber()**

**'** Paul Beverley - Version 07.02.11

' Add indexed section number

afterText = Chr(9): ' a tab

Dim v As Variable, i As Integer

varsExist = False

For Each v In ActiveDocument.Variables

If v.Name = "indexNum" Then varsExist = True: Exit For

Next v

If varsExist = False Then

ActiveDocument.Variables.Add "indexNum", 1

End If

' If no text is selected ...

If Selection.Start = Selection.End Then

' Add an index number

i = ActiveDocument.Variables("indexNum")

Selection.HomeKey Unit:=wdLine

' Selection.PasteAndFormat (wdFormatPlainText)

Selection.PasteSpecial DataType:=wdPasteText

Selection.TypeText Text:="." & Trim(Str(i)) + afterText

Selection.HomeKey Unit:=wdLine

If Selection = Chr(9) Then Selection.Delete Unit:=wdCharacter, Count:=1

ActiveDocument.Variables("indexNum") = i + 1

Else: ' if some text is selected ...

' set up a new index number

i = Val(Selection)

If i = 0 Then i = 1

myInput = InputBox("Start number?", "Section numberer", Trim(Str(i)))

If Len(myInput) = 0 Then Exit Sub

i = Val(myInput)

ActiveDocument.Variables("indexNum") = i

End If

Selection.End = Selection.Start

End Sub

**Sub FindAndDo()**

' Paul Beverley - Version 08.08.22

' Finds something specific and does things to each one

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "e"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

myCount = 0

Do While Selection.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the end of the found item is

Set rng = Selection.Range.Duplicate

' and make sure you're past it

rng.MoveEnd , 1

rng.Collapse wdCollapseEnd

' Do something with the thing you found

Selection.Font.Italic = True

' Restart search from after the previous occurrence

rng.Select

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

DoEvents

Loop

MsgBox "Changed: " & myCount

End Sub

**Sub FindAndDoRng()**

' Paul Beverley - Version 08.08.22

' Finds something specific and does things to each one

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "o"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the end of the found item is

Set rngWas = rng.Duplicate

' make sure you're past

rngWas.MoveEnd , 1

rngWas.Collapse wdCollapseEnd

' rng.Select

If myCount Mod 20 = 0 Then rng.Select

rng.Text = "ooo"

rng.Font.Italic = True

' Restart searching AFTER the previous occurrence

rng.End = rngWas.End

rng.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

rng.Find.Execute

DoEvents

Loop

MsgBox "Changed: " & myCount

End Sub

**Sub SampleFindAndDo()**

**'** Paul Beverley - Version 08.02.12

' Find something specific and do things to each one

' Start from the top

Selection.HomeKey Unit:=wdStory

' Go and find the first occurrence

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "([0-9])-([0-9])"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While Selection.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the end of the found item is

endNow = Selection.End

' Move the start of the selection one space right

' i.e. past the first number and in front of the hyphen

Selection.Start = Selection.Start + 1

' Put the end of the selection one place to the right of this

' i.e. select the hyphen

Selection.End = Selection.Start + 1

' Type a dash, which will replace the selected hyphen

Selection.TypeText Text:=ChrW(8211)

' Be sure you're past the previous occurrence

Selection.End = endNow

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

MsgBox "Changed: " & myCount

End Sub

**Sub MultiChoiceTidierGlobal()**

**'** Paul Beverley - Version 26.01.12

' Lower case first word and remove end spaces and punctuation

myCol = wdYellow

' For no highlighting

' myCol = 0

Set rng = ActiveDocument.Content

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

' List here the possible item indicators

.Text = "^13[ABCDE].[ " & Chr(9) & "]"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

wasEnd = rng.End

rng.Collapse wdCollapseEnd

If rng.Find.Found = True Then

rng.MoveEnd wdCharacter, 1

rng.Case = wdLowerCase

If myCol > 0 Then rng.HighlightColorIndex = myCol

rng.Collapse wdCollapseEnd

With rng.Find

.Text = "^p"

.MatchWildcards = False

.Execute

End With

rng.MoveEnd wdCharacter, -1

' List here possible erroneous characters

rng.MoveStartWhile cset:=". :;!?", Count:=wdBackward

If rng.Start <> rng.End Then rng.Delete

stopNow = False

Else

stopNow = True

End If

Selection.Start = Selection.End

Loop Until stopNow = True

End Sub

**Sub MultiChoiceTidierSingle()**

**'** Paul Beverley - Version 28.01.11

' Lowercases initial char of answer + remove end spaces + punct

myMarker = "." & " "

'myMarker = "." & Chr(9)

'myMarker = ")" & " "

'myMarker = ")" & Chr(9)

myChars = "ABCDE"

' Find a line that looks like an answer line

isAnAnswer = False

' Find an answer line

Do

Selection.Paragraphs(1).Range.Select

startHere = Selection.Start

endHere = Selection.End - 1

myText = Selection.Text

Selection.Collapse wdCollapseEnd

If Len(myText) > 2 Then

startLetter = Left(myText, 1)

If InStr(myText, myMarker) = 2 And InStr(myChars, \_

startLetter) > 0 Then isAnAnswer = True

End If

isTheEnd = (ActiveDocument.Content.End - Selection.End < 2)

Loop Until isAnAnswer = True Or isTheEnd

Do

If isAnAnswer = True Then

Set rng = ActiveDocument.Content

rng.Start = endHere

rng.End = endHere

' Strip off spaces and punctuation

rng.MoveStartWhile cset:=". :;!?", Count:=wdBackward

If Len(rng.Text) > 0 Then rng.Delete

rng.Start = startHere + 3

rng.End = startHere + 4

rng.Case = wdLowerCase

End If

isAnAnswer = False

' Check the following line

Selection.Paragraphs(1).Range.Select

startHere = Selection.Start

endHere = Selection.End - 1

myText = Selection.Text

Selection.Collapse wdCollapseEnd

If Len(myText) > 2 Then

startLetter = Left(myText, 1)

If InStr(myText, myMarker) = 2 And InStr(myChars, \_

startLetter) > 0 Then isAnAnswer = True

End If

Loop Until isAnAnswer = False

End Sub

**Sub SerialCommaHighlight()**

**'** Paul Beverley - Version 08.06.24

' Highlight or underline text that appears to have a serial comma

maxWords = 7

doUnderline = False

doHighlight = False

doColour = True

myColour = wdYellow

myFontColour = wdColorBlue

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9a-zA-Z'" & ChrW(8217) & "\-]@, [0-9a-zA-Z'" & ChrW(8217) & "\- ]@, and "

.Replacement.Text = ""

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

While rng.Find.Found

If rng.Words.Count < maxWords Then

If doUnderline = True Then rng.Font.Underline = True

If doHighlight = True Then rng.HighlightColorIndex = myColour

If doColour = True Then rng.Font.Color = myFontColour

End If

rng.Start = rng.End

rng.Find.Execute

DoEvents

If rng.End < rng.Start Then

rng.End = rng.Start + 2

rng.Start = rng.End

End If

Wend

Set rng = ActiveDocument.Content

With rng.Find

.Text = "[0-9a-zA-Z'" & ChrW(8217) & "\-]@, [0-9a-zA-Z'" & ChrW(8217) & "\- ]@, or "

.Replacement.Text = ""

.Execute

End With

While rng.Find.Found

If rng.Words.Count < maxWords Then

If doUnderline = True Then rng.Font.Underline = True

If doHighlight = True Then rng.HighlightColorIndex = myColour

If doColour = True Then rng.Font.Color = myFontColour

End If

rng.Start = rng.End

rng.Find.Execute

DoEvents

If rng.End < rng.Start Then

rng.End = rng.Start + 2

rng.Start = rng.End

End If

Wend

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Execute

End With

If Selection.Find.Found = False Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Execute

End With

End If

Beep

End Sub

**Sub** **SerialNotCommaHighlight()**

**'** Paul Beverley - Version 08.06.24

' Highlights or underlines text that appears not to have a serial comma

maxWords = 7

doUnderline = False

doHighlight = False

doColour = True

myColour = wdYellow

myFontColour = wdColorBlue

For i = 1 To 3

Select Case i

Case 1:

Num = ActiveDocument.Footnotes.Count

If Num > 0 Then Set rng0 = \_

ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2:

Num = ActiveDocument.Endnotes.Count

If Num > 0 Then Set rng0 = \_

ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3:

Set rng0 = ActiveDocument.Content

Num = 1

End Select

If Num > 0 Then

Set rng = rng0.Duplicate

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9a-zA-Z'" & ChrW(8217) & "\-]@, [0-9a-zA-Z'" & ChrW(8217) & "\- ]@ and "

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

While rng.Find.Found

If rng.Words.Count < maxWords Then

If doUnderline = True Then rng.Font.Underline = True

If doHighlight = True Then rng.HighlightColorIndex = myColour

If doColour = True Then rng.Font.Color = myFontColour

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

If rng.End < rng.Start Then

rng.End = rng.Start + 2

rng.Start = rng.End

End If

Wend

Set rng = rng0.Duplicate

With rng.Find

.Text = "[0-9a-zA-Z'" & ChrW(8217) & "\-]@, [0-9a-zA-Z'" & ChrW(8217) & "\- ]@ or "

.MatchWildcards = True

.Replacement.Text = ""

.Wrap = wdFindStop

.Execute

End With

While rng.Find.Found

If rng.Words.Count < maxWords Then

If doUnderline = True Then rng.Font.Underline = True

If doHighlight = True Then rng.HighlightColorIndex = myColour

If doColour = True Then rng.Font.Color = myFontColour

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

If rng.End < rng.Start Then

rng.End = rng.Start + 2

rng.Start = rng.End

End If

Wend

End If

Next i

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Text = ""

.Font.Underline = True

.Execute

End With

If Selection.Find.Found = False Then

With Selection.Find

.ClearFormatting

.Text = ""

.Highlight = True

.Execute

End With

End If

Beep

End Sub

**Sub SerialCommaCounter()**

' Paul Beverley - Version 12.11.19

' Counts serial (or not) commas in lists

maxWords = 7

serialColour = wdBrightGreen

notSerialColour = wdYellow

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " and "

.Font.Underline = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Execute

End With

If rng.Find.Found = False Then

serCount = 0

notCount = 0

myResponse = MsgBox("Count serial commas?", \_

vbQuestion + vbYesNoCancel, "SerialCommaCounter")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-zA-Z\-]@, [a-zA-Z\- ]@> and "

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

While rng.Find.Found

If rng.Words.Count < maxWords + 1 Then

rng.Underline = True

notCount = notCount + 1

StatusBar = "Serial: " & Str(serCount) & \_

" NO serial: " & Str(notCount)

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

If rng.End < rng.Start Then

rng.End = rng.Start + 2

rng.Start = rng.End

End If

Wend

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-zA-Z\-]@, [a-zA-Z\- ]@, and "

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

While rng.Find.Found

If rng.Words.Count < maxWords + 1 Then

rng.Underline = True

serCount = serCount + 1

StatusBar = "Serial: " & Str(serCount) & \_

" NO serial: " & Str(notCount)

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

If rng.End < rng.Start Then

rng.End = rng.Start + 2

rng.Start = rng.End

End If

Wend

Set rng = ActiveDocument.Content

With rng.Find

.Text = "[a-zA-Z\-]@, [a-zA-Z\- ]@> or "

.MatchWildcards = True

.Replacement.Text = ""

.Wrap = wdFindStop

.Execute

End With

While rng.Find.Found

If rng.Words.Count < maxWords + 1 Then

rng.Underline = True

notCount = notCount + 1

StatusBar = "Serial: " & Str(serCount) & \_

" NO serial: " & Str(notCount)

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

If rng.End < rng.Start Then

rng.End = rng.Start + 2

rng.Start = rng.End

End If

Wend

Set rng = ActiveDocument.Content

With rng.Find

.Text = "[a-zA-Z\-]@, [a-zA-Z\- ]@, or "

.MatchWildcards = True

.Replacement.Text = ""

.Wrap = wdFindStop

.Execute

End With

While rng.Find.Found

If rng.Words.Count < maxWords + 1 Then

rng.Underline = True

serCount = serCount + 1

StatusBar = "Serial: " & Str(serCount) & \_

" NO serial: " & Str(notCount)

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

If rng.End < rng.Start Then

rng.End = rng.Start + 2

rng.Start = rng.End

End If

Wend

Beep

End If

myResponse = MsgBox("Check which items really are lists?", \_

vbQuestion + vbYesNoCancel, "SerialCommaCounter")

If myResponse <> vbYes Then Exit Sub

serCount = 0

notCount = 0

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Execute

End With

While rng.Find.Found

colourHere = rng.HighlightColorIndex

myResponse = vbYes

If colourHere <> wdNoHighlight Then

If colourHere = serialColour Then

serCount = serCount + 1

Else

notCount = notCount + 1

End If

Else

rng.Select

ActiveDocument.ActiveWindow.LargeScroll Down:=1

ActiveDocument.ActiveWindow.SmallScroll Down:=1

rng.Select

myResponse = MsgBox("Is this a list?", \_

vbQuestion + vbYesNoCancel, "SerialCommaCounter")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then rng.Font.Underline = False

End If

If myResponse = vbYes Then

If InStr(rng, ", and ") > 0 Or InStr(rng, ", or ") > 0 Then

Selection.range.HighlightColorIndex = serialColour

serCount = serCount + 1

Else

Selection.range.HighlightColorIndex = notSerialColour

notCount = notCount + 1

End If

End If

StatusBar = "Serial: " & Str(serCount) & \_

" NO serial: " & Str(notCount)

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Wend

Beep

MsgBox "Finished!" & vbCr & vbCr & "Serial: " & Str(serCount) & \_

" NO serial: " & Str(notCount)

End Sub

**Sub HighlightDuplicateSentences()**

' Paul Beverley - Version 01.05.16

' Highlight pairs of identical sentences

minWords = 10

numSent = ActiveDocument.Sentences.Count

For i = 1 To numSent - 1

If Not (ActiveDocument.Sentences(i).Words.Count < minWords) Then

For j = (i + 1) To numSent

If ActiveDocument.Sentences(i).Text = \_

ActiveDocument.Sentences(j).Text Then

ActiveDocument.Sentences(i).HighlightColorIndex = wdGray25

ActiveDocument.Sentences(j).HighlightColorIndex = wdYellow

ActiveDocument.Sentences(i).Select

End If

Next j

End If

DoEvents

StatusBar = "To go: " & Str(numSent - i)

Next i

Selection.HomeKey Unit:=wdStory

End Sub

**Sub HighlightDuplicateSentences2()**

' Paul Beverley - Version 01.05.16

' Highlight pairs of identical sentences

minWords = 5

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = wdNoHighlight

numSent = ActiveDocument.Sentences.Count

For i = 1 To numSent - 1

Set rngS = ActiveDocument.Sentences(i)

If rngS.Words.Count > minWords Then

Set rng = rngS.Duplicate

rng.Collapse wdCollapseEnd

rngS.Select

numChars = rngS.Characters.Count

mySearch = rngS.Text

longSentence = False

If numChars > 254 Then

longSentence = True

mySearch = Left(rngS.Text, 254)

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySearch

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If rng.Find.Found Then

rng.Select

If longSentence = True Then

rngS.HighlightColorIndex = wdGray25

rng.HighlightColorIndex = wdGray25

Else

rngS.HighlightColorIndex = wdYellow

rng.HighlightColorIndex = wdYellow

End If

rngS.Select

End If

End If

DoEvents

StatusBar = "To go: " & Str(numSent - i)

Next i

End Sub

**Sub EnclosureFixer()**

' Paul Beverley - Version 09.02.16

' Checks and corrects the order of enclosures - brackets etc.

'myOrder = "{([])}"

'myOrder = "([{}])"

myOrder = "{([])}"

addHighlight = True

hiColour = wdBrightGreen

errorColour = wdTurquoise

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[\[\]\{\}\(\)]"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Font.StrikeThrough = False

.Execute

End With

Dim o(3) As String

Dim c(3) As String

Dim myPos(1000) As Long

o(1) = Left(myOrder, 1)

o(2) = Mid(myOrder, 2, 1)

o(3) = Mid(myOrder, 3, 1)

c(3) = Mid(myOrder, 4, 1)

c(2) = Mid(myOrder, 5, 1)

c(1) = Right(myOrder, 1)

myTrack = ActiveDocument.TrackRevisions

myLevel = 0

myMaxLevel = 0

encloseNum = 0

Do While rng.Find.Found = True

' Number characters until end of set

encloseNum = encloseNum + 1

' Record position of each character in the set

myPos(encloseNum) = rng.Start

myCharNumber = InStr(myOrder, rng)

If myCharNumber < 4 Then

' It's an open enclosure

myLevel = myLevel + 1

If myLevel > myMaxLevel Then myMaxLevel = myLevel

If myMaxLevel = 4 Then

Beep

rng.Select

rng.Start = myPos(1)

rng.HighlightColorIndex = errorColour

MsgBox "Too many opens! This is level four."

Exit Sub

End If

Else

' It's a close enclosure

myLevel = myLevel - 1

If myLevel = -1 Then

Beep

rng.Select

rng.Start = myPosOld

rng.HighlightColorIndex = errorColour

MsgBox "Too many closes!"

Exit Sub

End If

End If

' Debug.Print encloseNum, myLevel, myMaxLevel

If myLevel = 0 Then

' We've found the final enclosure item

' So check all the characters in the set are right

Select Case myMaxLevel

Case 1: myCharsOpen = o(3): myCharsClose = c(3)

Case 2: myCharsOpen = o(2) & o(3): myCharsClose = c(2) & c(3)

Case 3: myCharsOpen = o(1) & o(2) & o(3)

myCharsClose = c(1) & c(2) & c(3)

End Select

actualLevel = 0

TCcorrection = 0

For i = 1 To encloseNum

rng.Start = myPos(i) + TCcorrection

rng.End = rng.Start + 1

nowChar = rng.Text

myCharNumber = InStr(myOrder, nowChar)

If myCharNumber < 4 Then

actualLevel = actualLevel + 1

thisChar = Mid(myCharsOpen, actualLevel, 1)

Else

thisChar = Mid(myCharsClose, actualLevel, 1)

actualLevel = actualLevel - 1

End If

If rng.Text <> thisChar Then

If myTrack = True Then

TCcorrection = TCcorrection + 1

ActiveDocument.TrackRevisions = True

End If

rng.Delete

rng.InsertAfter thisChar

ActiveDocument.TrackRevisions = False

rng.Start = rng.Start - 1

rng.HighlightColorIndex = hiColour

rng.Collapse wdCollapseEnd

End If

Next i

' Then reset the counters

myMaxLevel = 0

encloseNum = 0

End If

rng.Select

rng.Start = rng.End

rng.Find.Execute

myPosOld = myPos(1)

Loop

Beep

rng.Select

If myLevel > 0 Then

rng.Start = myPos(1)

rng.HighlightColorIndex = errorColour

MsgBox "Final group not closed."

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub FormatRemoveNotURLs()**

' Paul Beverley - Version 01.09.16

' Removes all styles and formatting except URLs

Set rng = ActiveDocument.Content

rng.Style = ActiveDocument.Styles(wdStyleNormal)

rng.Font.Reset

For Each myURL In ActiveDocument.Hyperlinks

myURL.range.Font.Underline = True

myURL.range.Font.Color = wdColorBlue

Next myURL

End Sub

**Sub HighlightWithTrackChange()**

' Paul Beverley - Version 03.04.19

' Use allcaps, smallcaps, underline for tracking highlighting

myAllCaps = wdYellow

mySmallCaps = wdBrightGreen

myUnderline = wdNoHighlight

If myAllCaps > 0 Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.AllCaps = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

EndNow = rng.End

ActiveDocument.TrackRevisions = False

rng.Font.AllCaps = False

ActiveDocument.TrackRevisions = True

rng.HighlightColorIndex = myAllCaps

rng.Start = EndNow

rng.Find.Execute

Loop

End If

If mySmallCaps > 0 Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.SmallCaps = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

EndNow = rng.End

ActiveDocument.TrackRevisions = False

rng.Font.SmallCaps = False

ActiveDocument.TrackRevisions = True

rng.HighlightColorIndex = mySmallCaps

rng.Start = EndNow

rng.Find.Execute

Loop

End If

If myUnderline > 0 Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

EndNow = rng.End

ActiveDocument.TrackRevisions = False

rng.Font.Underline = False

ActiveDocument.TrackRevisions = True

rng.HighlightColorIndex = myUnderline

rng.Start = EndNow

rng.Find.Execute

Loop

End If

End Sub

**Sub LinesToParagraphs()**

' Paul Beverley - Version 16.08.19

' Converts lots of individual lines of text into paragraphs

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p^p"

.Wrap = wdFindStop

.Replacement.Text = "zczc"

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = "^p"

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

.Text = "zczc"

.Replacement.Text = "^p^p"

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub Confusables()**

' Paul Beverley - Version 17.06.22

' Highlights/colours list of words in confusables file

onlyColourWholeWords = False

closeConfusablesFile = False

' Address where Confusables file is held

' On Windows, it will need to be something like:

myFile = "C:\Documents and Settings\Paul\My Documents\Macro stuff\Confusables.docx"

' On a Mac, it will need to be something like:

myFile = "/Users/Paul/My Documents/Macro stuff/Confusables.docx"

For i = 4 To 100

myTest = Left(Right(myFile, i), 1)

If myTest = "\" Or myTest = "/" Then

confusablesFile = Right(myFile, i - 1)

Exit For

End If

Next i

dotPos = InStr(confusablesFile, ".")

confusablesFile = Left(confusablesFile, dotPos - 1)

myScreenOff = True

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

' Start of main program

Set mainDoc = ActiveDocument

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

gottadoc = False

For Each thisDoc In Application.Documents

thisName = thisDoc.Name

If InStr(thisName, confusablesFile) > 0 Then

gottadoc = True

closeExceptionsFile = False

thisDoc.Activate

Exit For

End If

Next thisDoc

On Error Resume Next

If gottadoc = False Then

Documents.Open myFile

If Err.Number = 5174 Then

MsgBox ("Please open the confusables file")

Err.Clear

Exit Sub

Else

On Error GoTo ReportIt

End If

End If

num = ActiveDocument.Paragraphs.Count

ReDim wd(num) As String

ReDim Hi(num) As Integer

ReDim col(num) As Integer

numWds = 0

Set rng = ActiveDocument.Content

If rng.HighlightColorIndex = 0 Then

Beep

myResponse = MsgBox("The words in your confusables file" & \_

vbCr & "need to be highlighted!", \_

vbOKOnly, "Confusables")

rng.HighlightColorIndex = wdYellow

Exit Sub

End If

For Each pa In ActiveDocument.Paragraphs

Set rng = pa.Range.Duplicate

rng.End = rng.End - 1

thisWord = rng.Text

Debug.Print thisWord & "|"

If Len(thisWord) > 2 Then

numWds = numWds + 1

wd(numWds) = thisWord

Hi(numWds) = rng.HighlightColorIndex

col(numWds) = rng.Font.ColorIndex

End If

Next pa

If closeConfusablesFile = True Then

ActiveDocument.Close SaveChanges:=False

End If

mainDoc.Activate

Selection.HomeKey Unit:=wdStory

oldColour = Options.DefaultHighlightColorIndex

For hit = 1 To 4

goes = 0

If hit = 1 Then

thisMany = ActiveDocument.Endnotes.Count

If thisMany > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

End If

End If

If hit = 2 Then

thisMany = ActiveDocument.Footnotes.Count

If thisMany > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

End If

End If

If hit = 3 Then

Set rng = ActiveDocument.Content

thisMany = 1

goes = 1

End If

goes = 1

someText = True

If hit = 4 Then

thisMany = ActiveDocument.Shapes.Count

goes = thisMany

End If

If goes > 0 And thisMany > 0 Then

For myGo = 1 To goes

If hit = 4 Then

Do

someText = False

If ActiveDocument.Shapes(myGo).Type <> 24 And \_

ActiveDocument.Shapes(myGo).Type <> 3 Then

someText = ActiveDocument.Shapes(myGo).TextFrame.HasText

If someText Then

Set rng = ActiveDocument.Shapes(myGo).TextFrame.TextRange

Else

myGo = myGo + 1

End If

End If

Loop Until someText Or myGo > goes

End If

theEnd = rng.End

If someText = True Then

For i = 1 To numWds

Options.DefaultHighlightColorIndex = Hi(i)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Format = True

.Text = wd(i)

.Wrap = wdFindContinue

.Replacement.Text = ""

If col(i) > 0 Then

.Replacement.Font.ColorIndex = col(i)

End If

If Hi(i) > 0 Then

.Replacement.Highlight = True

End If

.MatchWildcards = False

.MatchWholeWord = onlyColourWholeWords

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

End If

Next myGo

End If

Next hit

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Options.DefaultHighlightColorIndex = oldColour

ActiveDocument.TrackRevisions = myTrack

Beep

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub CapitaliseUndoubler()**

' Paul Beverley - Version 13.09.19

' Finds doubled capital letters and corrects them

myColour = wdBlue

' myColour = 0

' myHighlight = wdYellow

myHighlight = 0

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z]{2}[a-z" & ChrW(8217) & "]{2,}>"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

If myColour > 0 Then rng.Font.ColorIndex = myColour

If myHighlight > 0 Then rng.HighlightColorIndex = myHighlight

EndNow = rng.End

rng.Start = rng.Start + 1

rng.End = rng.Start + 1

rng.Text = LCase(rng.Text)

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

MsgBox "Double capitals changed: " & myCount

End Sub

**Sub SupercriptNumberFormatter()**

' Paul Beverley - Version 28.10.19

' Corrects spaces + punctuation on superscripted numbers

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]{1,}"

.Font.Superscript = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

rng.Select

Do While rng.Find.Found = True

EndNow = rng.End

Set rng2 = ActiveDocument.range(EndNow, EndNow + 1)

If InStr(vbCr & " ", rng2.Text) = 0 Then

myPunct = rng2.Text

rng2.Delete

Else

myPunct = ""

End If

rng.Collapse wdCollapseStart

rng.Start = rng.End - 1

If rng.Text = " " Then

rng.Delete

End If

rng.Collapse wdCollapseEnd

If myPunct > "" Then rng.InsertAfter Text:=myPunct

rng.Start = EndNow + 1

rng.End = EndNow + 1

rng.Find.Execute

DoEvents

Loop

rng.Select

Beep

End Sub

**Sub CaseNextChar()**

**'** Paul Beverley - Version 13.05.24

' Changes case of the next character/selection

trackIt = True

trackIfSelected = False

myShow = ActiveWindow.View.ShowRevisionsAndComments

myView = ActiveWindow.View.RevisionsView

myTrack = ActiveDocument.TrackRevisions

' Don't track case change if TC is OFF

If myTrack = False Then trackIt = False

' If an area of text is NOT selected ...

If Asc(Selection) = 32 Then

Selection.MoveRight , 1

Exit Sub

End If

If Selection.End = Selection.Start Then

If trackIt = False Then \_

ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

rng.MoveEnd , 1

If UCase(rng) = rng Then

myNewChar = LCase(rng)

Else

myNewChar = UCase(rng)

End If

rng.InsertAfter Text:=myNewChar

rng.Select

Selection.Collapse wdCollapseEnd

rng.MoveEnd , -1

rng.Delete

Else

' If an area of text is selected ...

If trackIfSelected = False Then trackIt = False

If Selection.Information(wdWithInTable) = True Then

If Selection.Range.Case = wdLowerCase Then

Selection.Range.Case = wdUpperCase

Else

Selection.Range.Case = wdLowerCase

End If

Else

myText = Selection

voteUpper = 0

voteLower = 0

For myCount = 1 To Len(myText)

myChar = Asc(Mid(myText, myCount, 1))

If myChar > 96 And myChar < 123 Then voteLower = voteLower + 1

If myChar > 64 And myChar < 91 Then voteUpper = voteUpper + 1

Next myCount

myUpper = (voteUpper > voteLower)

If voteLower = 0 Then myUpper = False

If trackIt = False Then

ActiveDocument.TrackRevisions = False

If myUpper = True Then

Selection.Range.Case = wdUpperCase

Else

Set rng = Selection

For Each myWd In rng.Words

If Len(myWd) > 1 Then

myCh1 = Left(myWd, 1)

myCh2 = Mid(myWd, 2, 1)

If LCase(myCh1) <> myCh1 And UCase(myCh2) <> myCh2 And \_

myWd.Start >= rng.Start Then \_

myWd.Case = wdLowerCase

End If

Next myWd

End If

If Selection = myText Then Selection.Range.Case = wdUpperCase

If Selection = myText Then Selection.Range.Case = wdLowerCase

Else

startWas = Selection.Start

If myUpper = True Then

myTextNew = UCase(myText)

Else

myTextNew = LCase(myText)

End If

If myTextNew = myText Then myTextNew = UCase(myText)

If myTextNew = myText Then myTextNew = LCase(myText)

wasBold = Selection.Font.Bold

wasItalic = Selection.Font.Italic

Selection.Delete

Selection.TypeText Text:=myTextNew

Selection.Start = startWas

If wasBold Then Selection.Font.Bold = True

If wasItalic Then Selection.Font.Italic = True

End If

End If

End If

ActiveDocument.TrackRevisions = myTrack

ActiveWindow.View.ShowRevisionsAndComments = myShow

End Sub

**Sub CaseNextWord()**

**'** Paul Beverley - Version 16.04.24

' Changes case of initial letter of next word or selection

trackIt = True

showSingleTrack = True

' List of lowercase words \*not\* to be uppercased

lclist = " a an and as at by for from if in is it into of "

lclist = lclist & " on or that the to with "

' Or if you don't want this feature, use:

' lclist = ""

' If an area of text is selected

If Selection.End > Selection.Start Then

Set rng = Selection.Range.Duplicate

endNow = rng.End

rng.Collapse wdCollapseStart

rng.Expand wdWord

startNow = rng.Start

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

rng.Start = startNow

rng.Select

Set rng = Selection.Range.Duplicate

If LCase(rng.Text) = rng.Text Then

For wd = 1 To rng.Words.Count

myWd = rng.Words(wd)

checkWd = " " & Trim(myWd) & " "

If InStr(lclist, checkWd) = 0 Then

ch = rng.Words(wd).Characters(1)

rng.Words(wd).Characters(1) = UCase(ch)

End If

Next wd

Else

For wd = 1 To rng.Words.Count

myWd = rng.Words(wd)

If UCase(myWd) <> myWd Then

ch = rng.Words(wd).Characters(1)

rng.Words(wd).Characters(1) = LCase(ch)

End If

Next wd

End If

If trackIt = False Then

Selection.Range.Revisions.AcceptAll

End If

If showSingleTrack = True Then

myText = Selection

Set rng = Selection.Range.Duplicate

Selection.Range.Revisions.RejectAll

rng.Text = myText

End If

Else

' If no text is selected

Selection.MoveStart wdWord

Selection.MoveEnd , 1

If LCase(Selection) = UCase(Selection) Then

Selection.MoveStart wdWord

Selection.MoveEnd , 1

End If

If LCase(Selection) = UCase(Selection) Then

Selection.MoveStart wdWord

Selection.MoveEnd , 1

End If

If LCase(Selection) = UCase(Selection) Then

Selection.MoveStart wdWord

Selection.MoveEnd , 1

End If

If trackIt = False Then

Selection.Range.Case = wdToggleCase

Selection.MoveRight Unit:=wdCharacter, Count:=1

Else

m = Selection.Text

If UCase(m) = m Then

Selection.Text = LCase(m)

Else

Selection.Text = UCase(m)

End If

Selection.Collapse wdCollapseEnd

End If

End If

End Sub

**Sub CaseThisWord()**

**'** Paul Beverley Version 23.05.24

' Changes initial case of current word and jumps to next word

showSingleTrack = False

trackIt = True

' List of lowercase words \*not\* to be uppercased

lclist = " a an and as at by for from if in is it into of "

lclist = lclist & " on or that the to with "

' Or if you don't want this feature, use:

' lclist = ""

' If an area of text is selected

If Selection.End > Selection.Start Then

Set rng = Selection.Range.Duplicate

endNow = rng.End

rng.Collapse wdCollapseStart

rng.Expand wdWord

startNow = rng.Start

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

rng.Start = startNow

rng.Select

Set rng = Selection.Range.Duplicate

If LCase(rng.Text) = rng.Text Then

For wd = 1 To rng.Words.Count

myWd = rng.Words(wd)

checkWd = " " & Trim(myWd) & " "

Debug.Print "|" & checkWd & "|"

If InStr(lclist, checkWd) = 0 Then

ch = rng.Words(wd).Characters(1)

rng.Words(wd).Characters(1) = UCase(ch)

End If

DoEvents

Next wd

Else

For wd = 1 To rng.Words.Count

myWd = rng.Words(wd)

checkWd = " " & Trim(myWd) & " "

If InStr(lclist, checkWd) = 0 And UCase(myWd) <> myWd Then

ch = rng.Words(wd).Characters(1)

rng.Words(wd).Characters(1) = LCase(ch)

End If

DoEvents

Next wd

End If

If trackIt = False Then

Selection.Range.Revisions.AcceptAll

End If

If showSingleTrack = True Then

myText = Selection

Set rng = Selection.Range.Duplicate

Selection.Range.Revisions.RejectAll

rng.Text = myText

End If

Else

' If no text is selected

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

Selection.MoveEnd , 1

If LCase(Selection) = UCase(Selection) Then

Selection.MoveStart wdWord

Selection.MoveEnd , 1

End If

If trackIt = False Then

Selection.Range.Case = wdToggleCase

Selection.MoveRight Unit:=wdCharacter, Count:=1

Else

m = Selection.Text

If UCase(m) = m Then

Selection.Text = LCase(m)

Else

Selection.Text = UCase(m)

End If

Selection.Collapse wdCollapseEnd

End If

End If

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

rng.Collapse wdCollapseEnd

rng.Select

End Sub

**Sub CaseNextPara()**

**'** Paul Beverley - Version 12.08.10

' Changes case of initial letter of the paragraph

trackIt = True

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

Selection.HomeKey Unit:=wdLine

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-zA-Z]"

.Replacement.Text = ""

.Wrap = wdFindContinue

.Forward = True

.MatchWildcards = True

.Execute

End With

If trackIt = False Then

Selection.Range.Case = wdToggleCase

Else

myChar = Selection

If Asc(myChar) > 96 Then

myChar = UCase(myChar)

Else

myChar = LCase(myChar)

End If

Selection.Delete

Selection.TypeText Text:=myChar

End If

Selection.MoveDown Unit:=wdParagraph, Count:=1

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

End Sub

**Sub SemicolonEndPara()**

**'** Paul Beverley - Version 09.02.11

' Lowercase first character + add semicolon at end

Selection.Paragraphs(1).Range.Select

Selection.End = Selection.Start

lineStart = Selection.Start

startChar = Selection

If LCase(startChar) <> startChar Then

Selection.Delete Unit:=wdCharacter, Count:=1

Selection.TypeText Text:=LCase(startChar)

End If

Selection.End = Selection.End + 3

Selection.Start = Selection.Start + 2

startChar = Selection

If LCase(startChar) <> startChar Then

Selection.Delete Unit:=wdCharacter, Count:=1

Selection.TypeText Text:=LCase(startChar)

End If

Selection.Paragraphs(1).Range.Select

Selection.Start = Selection.End - 2

Selection.End = Selection.End - 1

lastChar = Selection

If lastChar = " " Then

Selection.Delete

Selection.Start = Selection.Start - 1

lastChar = Selection

End If

If LCase(lastChar) = UCase(lastChar) And lastChar <> ";" \_

And lastChar <> ")" Then

Selection.Delete

Selection.TypeText Text:=";"

Else

Selection.MoveRight Unit:=wdCharacter, Count:=1

Selection.TypeText Text:=";"

End If

Selection.MoveRight Unit:=wdCharacter, Count:=2

End Sub

**Sub ListLowercaseSemicolon()**

' Paul Beverley - Version 17.12.14

' Add semicolons to bulletted list + lowercase initial char

' Shift-Alt-L

trackThis = True

addAnd = False

checkLength = 10

notThese = ";,. "

nowTrack = ActiveDocument.TrackRevisions

If trackThis = False Then ActiveDocument.TrackRevisions = False

Selection.Expand wdParagraph

startStyle = Selection.Style

startColour = Selection.Font.Color

i = 0

Do

endItem = Selection.End - 1

' select LH end until you meet an alpha character

Selection.End = Selection.Start + 1

If UCase(Selection) = LCase(Selection) Then Selection.MoveEnd , 1

If UCase(Selection) = LCase(Selection) Then Selection.MoveEnd , 1

Selection.Range.Case = wdLowerCase

' Get rid of any stray characters you don't want

Selection.Start = endItem

Do

gotOne = False

Selection.MoveStart , -1

If InStr(notThese, Selection) > 0 Then

Selection.Delete

gotOne = True

End If

Loop Until gotOne = False

Selection.InsertAfter ";"

' Now select the next item/para

Selection.MoveStart , 3

Selection.Expand wdParagraph

nowStyle = Selection.Style

nowColour = Selection.Font.Color

i = i + 1

If i = checkLength Then

myResponse = MsgBox("Continue?", vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

i = 0

End If

Loop Until nowStyle <> startStyle Or nowColour <> startColour

Selection.End = Selection.Start - 1

Selection.Start = Selection.End - 1

If Selection = ";" Then Selection.Delete

Selection.MoveStart wdCharacter, -1

If Selection <> "." Then Selection.InsertAfter "."

If addAnd = True Then

' Go back to penultimate

Selection.Expand wdParagraph

Selection.Start = Selection.Start - 1

Selection.End = Selection.Start

Selection.TypeText Text:=" and"

Selection.MoveStart wdCharacter, -10

If Selection = "; and; and" Then

Selection.MoveStart wdCharacter, 5

Selection.Delete

End If

Selection.MoveStart , 1

If Selection = " and; and" Then

Selection.MoveEnd , -5

Selection.Delete

End If

End If

Selection.Start = Selection.End

ActiveDocument.TrackRevisions = nowTrack

End Sub

**Sub ListSemicolon()**

**'** Paul Beverley - Version 15.02.11

' Add semicolons to bulleted list

addAnd = True

checkLength = 10

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.Paragraphs(1).Range.Select

startStyle = Selection.Style

i = 0

Do

Selection.End = Selection.End - 1

Selection.Start = Selection.End

notThese = ";,. "

Do

gotOne = False

Selection.MoveStart , -1

If InStr(notThese, Selection) > 0 Then

Selection.Delete

gotOne = True

End If

Loop Until gotOne = False

Selection.InsertAfter ";"

' Now select the next item/para

Selection.MoveStart , 3

Selection.Paragraphs(1).Range.Select

nowStyle = Selection.Style

i = i + 1

If i = checkLength Then

myResponse = MsgBox("Continue?", vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

i = 0

End If

Loop Until nowStyle <> startStyle

Selection.End = Selection.Start - 1

Selection.Start = Selection.End - 1

If Selection = ";" Then Selection.Delete

Selection.MoveStart wdCharacter, -1

If Selection <> "." Then Selection.InsertAfter "."

If addAnd = True Then

' Go back to penultimate

Selection.Paragraphs(1).Range.Select

Selection.Start = Selection.Start - 1

Selection.End = Selection.Start

Selection.TypeText Text:=" and"

Selection.MoveStart wdCharacter, -10

If Selection = "; and; and" Then

Selection.MoveStart wdCharacter, 5

Selection.Delete

End If

Selection.MoveStart , 1

If Selection = " and; and" Then

Selection.MoveEnd , -5

Selection.Delete

End If

End If

Selection.Start = Selection.End

ActiveDocument.TrackRevisions = nowTrack

End Sub

**Sub ListLowercaseNoPunct()**

' Paul Beverley - Version 03.10.17

' Lowercases the initial char + removes end punctuation

trackThis = False

doLowerCase = True

nowTrack = ActiveDocument.TrackRevisions

If trackThis = False Then ActiveDocument.TrackRevisions = False

Selection.Expand wdParagraph

nowStyle = Selection.range.Style

nowList = Selection.range.ListFormat.ListType

startHere = Selection.Start

endHere = Selection.End - 1

Selection.Collapse wdCollapseEnd

' Check next paragraph

Selection.Expand wdParagraph

nextStyle = Selection.range.Style

nextList = Selection.range.ListFormat.ListType

nextStart = Left(Selection, 5)

isLastItem = False

If nowStyle <> nextStyle Then isLastItem = True: Beep

If nowList > 0 And nextList = 0 Then isLastItem = True: Beep

Selection.End = Selection.Start + 5

nextPosTab = InStr(Selection, Chr(9))

If nextPosTab = 0 And nowList = 0 Then

nextPosTab = InStr(Selection, " ")

End If

' Check beginning of nowParagraph

Selection.Start = startHere

Selection.End = endHere + 5

posTab = InStr(Selection, Chr(9))

If posTab = 0 And nowList = 0 Then

posTab = InStr(Selection, " ")

End If

' select first character

Selection.End = startHere + 1

' If there's a bullet + a tab, go past it

If posTab > 0 Then

If nextPosTab = 0 Then isLastItem = True: Beep

Selection.MoveStart wdCharacter, posTab

Selection.Collapse wdCollapseStart

End If

If doLowerCase = True Then

If trackThis = False Then

Selection.range.Case = wdLowerCase

Else

newText = LCase(Selection)

Selection.Delete

Selection.TypeText newText

End If

End If

If trackThis = True Then endHere = endHere + 1

Selection.Start = endHere - 1

Selection.End = endHere

If Asc(Selection) = 13 Then

Selection.MoveStart , -1

Selection.MoveEnd , -1

End If

If Selection = " " Then

Selection.Delete

Selection.MoveStart wdCharacter, -1

End If

If Selection = " " Then

Selection.Delete

Selection.MoveStart wdCharacter, -1

End If

If Selection = "." And isLastItem = False Then Selection.Delete

If Selection = ";" Then Selection.Delete

If Selection = "," Then Selection.Delete

If Asc(Selection) = 2 Then

Selection.MoveLeft Unit:=wdCharacter, Count:=2

Selection.Delete

End If

Selection.Start = Selection.End - 5

If Selection = "; and" Then Selection.Delete

Selection.Start = endHere

If isLastItem = False Then Selection.MoveRight , 1

ActiveDocument.TrackRevisions = nowTrack

End Sub

**Sub ListUppercaseNoPunct()**

' Paul Beverley - Version 29.02.20

' Uppercases the initial character + removes end punctuation

trackThis = False

nowTrack = ActiveDocument.TrackRevisions

If trackThis = False Then ActiveDocument.TrackRevisions = False

Selection.Expand wdParagraph

nowStyle = Selection.range.Style

nowList = Selection.range.ListFormat.ListType

startHere = Selection.Start

endHere = Selection.End - 1

Selection.Collapse wdCollapseEnd

' Check next paragraph

Selection.Expand wdParagraph

nextStyle = Selection.range.Style

nextList = Selection.range.ListFormat.ListType

nextStart = Left(Selection, 5)

isLastItem = False

If nowStyle <> nextStyle Then isLastItem = True: Beep

If nowList > 0 And nextList = 0 Then isLastItem = True: Beep

Selection.End = Selection.Start + 5

nextPosTab = InStr(Selection, Chr(9))

If nextPosTab = 0 And nowList = 0 Then

nextPosTab = InStr(Selection, " ")

End If

' Check beginning of nowParagraph

Selection.Start = startHere

Selection.End = endHere + 5

posTab = InStr(Selection, Chr(9))

If posTab = 0 And nowList = 0 Then

posTab = InStr(Selection, " ")

End If

' select first character

Selection.End = startHere + 1

' If there's a bullet + a tab, go past it

If posTab > 0 Then

If nextPosTab = 0 Then isLastItem = True: Beep

Selection.MoveStart wdCharacter, posTab

Selection.Collapse wdCollapseStart

End If

If trackThis = False Then

Selection.MoveEnd , 1

Selection.range.Case = wdUpperCase

Else

newText = UCase(Selection)

Selection.Delete

Selection.TypeText newText

End If

If trackThis = True Then endHere = endHere + 1

Selection.Start = endHere - 1

Selection.End = endHere

If Asc(Selection) = 13 Then

Selection.MoveStart , -1

Selection.MoveEnd , -1

End If

If Selection = " " Then

Selection.Delete

Selection.MoveStart wdCharacter, -1

End If

If Selection = " " Then

Selection.Delete

Selection.MoveStart wdCharacter, -1

End If

If Selection = "." And isLastItem = False Then Selection.Delete

If Selection = "." Then Selection.Delete

If Selection = ";" Then Selection.Delete

If Selection = "," Then Selection.Delete

If Asc(Selection) = 2 Then

Selection.MoveLeft Unit:=wdCharacter, Count:=2

Selection.Delete

End If

Selection.Start = Selection.End - 5

If Selection = "; and" Then Selection.Delete

If Selection = ", and" Then Selection.Delete

Selection.Start = endHere

If isLastItem = False Then Selection.MoveRight , 1

ActiveDocument.TrackRevisions = nowTrack

End Sub

**Sub FullPointOnBullets()**

' Paul Beverley - Version 09.06.17

' Finds bullet items and ensures they have a full point

' myBullet = AscW(8226) ' Mac safe code

myBullet = "•"

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myBullet

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

rng.Expand wdParagraph

lastChar = Left(Right(rng.Text, 2), 1)

If lastChar = "." Or lastChar = "?" Or lastChar = "!" Then

' No nothing!

Else

rng.Start = rng.End - 1

rng.Select

rng.InsertBefore Text:="."

End If

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

rng.Find.Execute

Loop

End Sub

**Sub TypeThat()**

**'** Paul Beverley - Version 20.07.24

' Types 'that' after the current word.

Set rng = Selection.Range.Duplicate

rng.MoveStart , -1

If Left(rng.Text, 1) = " " Then Selection.MoveLeft , 1

Selection.Expand wdWord

Selection.Collapse wdCollapseEnd

Selection.TypeText "that "

End Sub

**Sub CharToAcute()**

' Paul Beverley - Version 01.02.21

' Adds an acute accent to the next vowel

myChars = "aeiouAEIOU"

myAccents = ChrW(225) & ChrW(233) & ChrW(237) & ChrW(243) \_

& ChrW(250) & ChrW(193) & ChrW(201) & ChrW(205) \_

& ChrW(211) & ChrW(218)

Selection.MoveUntil cset:=myChars, Count:=wdForward

Selection.MoveEnd , 1

charPos = InStr(myChars, Selection)

Selection.TypeText Mid(myAccents, charPos, 1)

End Sub

**Sub CharToGrave()**

' Paul Beverley - Version 01.02.21

' Adds a grave acute accent to the next vowel

myChars = "aeiouAEIOU"

myAccents = ChrW(224) & ChrW(232) & ChrW(236) & ChrW(242) \_

& ChrW(249) & ChrW(192) & ChrW(200) & ChrW(204) \_

& ChrW(210) & ChrW(217)

' For French?

' myChars = "aeuAEU"

' myAccents = ChrW(224) & ChrW(232) & ChrW(249) & ChrW(192) \_

' & ChrW(200) & ChrW(217)

Selection.MoveUntil cset:=myChars, Count:=wdForward

Selection.MoveEnd , 1

charPos = InStr(myChars, Selection)

Selection.TypeText Mid(myAccents, charPos, 1)

End Sub

**Sub CharToCircumflex()**

' Paul Beverley - Version 01.02.21

' Adds a circumflex to the next vowel

myChars = "aeouAEOU"

myAccents = ChrW(226) & ChrW(234) & ChrW(244) & ChrW(251) \_

& ChrW(194) & ChrW(202) & ChrW(212) & ChrW(219)

Selection.MoveUntil cset:=myChars, Count:=wdForward

Selection.MoveEnd , 1

charPos = InStr(myChars, Selection)

Selection.TypeText Mid(myAccents, charPos, 1)

End Sub

**Sub CharToUmlaut()**

' Paul Beverley - Version 01.02.21

' Adds various accents

myChars = "aeiouAEIOU"

myAccents = ChrW(228) & ChrW(235) & ChrW(239) & ChrW(246) \_

& ChrW(252) & ChrW(196) & ChrW(203) & ChrW(207) \_

& ChrW(214) & ChrW(220)

Selection.MoveUntil cset:=myChars, Count:=wdForward

Selection.MoveEnd , 1

charPos = InStr(myChars, Selection)

Selection.TypeText Mid(myAccents, charPos, 1)

End Sub

**Sub CharToVariousAccents()**

' Paul Beverley - Version 01.02.21

' Adds various accents

myChars = "ncNC"

myAccents = ChrW(241) & ChrW(231) & ChrW(209) & ChrW(199)

Selection.MoveUntil cset:=myChars, Count:=wdForward

Selection.MoveEnd , 1

charPos = InStr(myChars, Selection)

Selection.TypeText Mid(myAccents, charPos, 1)

End Sub

**Sub ArticleChanger()**

' Paul Beverley - Version 28.10.17

' Types, deletes or switches article 'the'/'a'/'an'.

If Selection = " " Then Selection.MoveRight , 1

useA = (Selection.Start <> Selection.End)

Set wd = Selection.range.Duplicate

wd.Expand wdWord

Set prevWd = wd.Duplicate

prevWd.End = wd.Start - 1

prevWd.Expand wdWord

If InStr(" a an the ", " " & Trim(wd.Text) & " ") > 0 Then

wd.Delete

Exit Sub

End If

If InStr(" A An The ", " " & Trim(wd.Text) & " ") > 0 Then

wd.Delete

Selection.MoveEnd 1

newChar = UCase(Selection)

If newChar <> Selection Then

Selection.Delete

Selection.TypeText Text:=newChar

Else

Selection.Collapse wdCollapseEnd

End If

Exit Sub

End If

If InStr(" a an ", " " & Trim(prevWd.Text) & " ") > 0 Then

prevWd.Select

Selection.Delete

Selection.TypeText Text:="the "

Exit Sub

End If

If prevWd = "the " Then

prevWd.Select

Selection.Delete

nextchar = Left(wd, 1)

If InStr("aeiouAEIOU", nextchar) > 0 Then

Selection.TypeText Text:="an "

Else

Selection.TypeText Text:="a "

End If

Exit Sub

End If

If InStr(" A An ", " " & Trim(prevWd.Text) & " ") > 0 Then

prevWd.Select

Selection.Delete

Selection.TypeText Text:="The "

Exit Sub

End If

If prevWd = "The " Then

prevWd.Select

Selection.Delete

nextchar = Left(wd, 1)

If InStr("aeiouAEIOU", nextchar) > 0 Then

Selection.TypeText Text:="An "

Else

Selection.TypeText Text:="A "

End If

Exit Sub

End If

firstChar = Left(wd, 1)

nextchar = Mid(wd, 2, 1)

cd = Asc(prevWd)

isABreak = (cd > 10 And cd < 15)

If cd = 149 Then isABreak = True

If InStr(prevWd, ".") > 0 Then isABreak = True

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

needCap = False

If prevWd = ". " Or isABreak = True Then

Selection.Expand wdCharacter

newInitial = LCase(Selection)

If UCase(nextchar) <> nextchar Then

If Selection <> newInitial Then

Selection.Delete

Selection.TypeText Text:=newInitial

Selection.MoveLeft , 1

End If

End If

Selection.Collapse wdCollapseStart

needCap = True

End If

If useA = False Then

If needCap = True Then

Selection.TypeText Text:="The "

Else

Selection.TypeText Text:="the "

End If

Else

If InStr("aeiouAEIOU", firstChar) > 0 Then

If needCap = True Then

Selection.TypeText Text:="An "

Else

Selection.TypeText Text:="an "

End If

Else

If needCap = True Then

Selection.TypeText Text:="A "

Else

Selection.TypeText Text:="a "

End If

End If

End If

End Sub

**Sub TypeA()**

' Paul Beverley - Version 15.01.21

' Types 'a' or 'A', (or 'an' or 'An') between two words

If Len(Selection) > 2 Then

Selection.Expand wdWord

cap = Left(Selection, 1)

If LCase(cap) = cap Then

If InStr("aeiou", cap) > 0 Then

Selection.TypeText "an "

Else

Selection.TypeText "a "

End If

Else

Selection.TypeText "A "

End If

Selection.MoveLeft , 1

Exit Sub

End If

Set rng = ActiveDocument.Content

rng.Start = Selection.Start - 1

rng.End = Selection.Start

prevChar = Asc(rng)

If prevChar = 13 And UCase(A) <> A Then Selection.TypeText "a ": Exit Sub

Selection.MoveStart , -2

Selection.MoveEnd , 1

newPos = InStr(Selection, " ")

If newPos = 0 Then newPos = 3

Selection.MoveStart , newPos

Selection.Expand wdWord

Do While InStr(ChrW(13) & " ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

' For the/The, sort out the capitalisation

firstChar = Left(Selection, 1)

secondChar = Mid(Selection, 2, 1)

If Len(Selection) > 2 Then

thirdChar = Mid(Selection, 3, 1)

Else

thirdChar = "a" ' not an uppercase char

End If

Selection.Collapse wdCollapseStart

' Check for sentence start indicator

Set rng = Selection.Range

rng.MoveStart wdCharacter, -5

spPos = InStr(rng, Chr(32))

crPos = InStr(rng, vbCr) - 3

tabPos = InStr(rng, Chr(9))

fpPos = InStr(rng, ".")

If fpPos = 0 Then fpPos = InStr(rng, "?")

If fpPos = 0 Then fpPos = InStr(rng, "!")

gotanFP = tabPos + fpPos + crPos

thisText = rng

If Len(thisText) - Len(Replace(thisText, vbCr, "")) > 1 Then gotanFP = 1

If Len(thisText) - Len(Replace(thisText, " ", "")) > 1 Then gotanFP = 0

If gotanFP > 0 Then

gotAnAcronym = False

If LCase(secondChar) <> secondChar Or LCase(secondChar) = \_

UCase(secondChar) Or LCase(thirdChar) <> thirdChar Or \_

LCase(thirdChar) = UCase(thirdChar) Then gotAnAcronym = True

If gotAnAcronym = False And firstChar <> LCase(firstChar) Then

Selection.Delete

Selection.TypeText LCase(firstChar)

Selection.MoveEnd wdCharacter, -1

End If

Selection.TypeText "A "

Else

Selection.TypeText "a "

End If

nextchar = Chr(Asc(Selection))

Selection.MoveLeft , 1

' A list of letters where we say 'an' rather than 'a'

If InStr("aAeEiIoOuU", nextchar) = 0 Then Exit Sub

Selection.TypeText "n"

End Sub

**Sub CharToMacron()**

' Paul Beverley - Version 13.09.19

' Adds a macron accent to the next vowel

myChars = "aeiouAEIOU"

myAccents = ChrW(257) & ChrW(275) & ChrW(299) & ChrW(333) & ChrW(363) & ChrW(256) & ChrW(274) & ChrW(298) & ChrW(332) & ChrW(362)

Selection.MoveUntil cset:=myChars, Count:=wdForward

Selection.MoveEnd , 1

charPos = InStr(myChars, Selection)

Selection.TypeText Text:=Mid(myAccents, charPos, 1)

End Sub

**Sub ListBulleter()**

' Paul Beverley - Version 09.06.12

' Add a bullet to every paragraph in a list

funnyFont = "Wingdings 2"

' If you don't want a funny font, use:

' funnyFont = ""

mySeparator = " "

' If you prefer a tab, use:

' mySeparator = Chr(9)

theEnd = Selection.End

Selection.End = Selection.Start

Do

' Select the current paragraph

Selection.Paragraphs(1).Range.Select

Selection.InsertBefore "zczc" & mySeparator

Selection.Start = Selection.End

Loop Until Selection.End > theEnd

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc"

.Wrap = wdFindContinue

.Replacement.Text = ChrW(8226)

If Len(funnyFont) > 2 Then .Replacement.Font.Name = funnyFont

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub InitialNext()**

**'** Paul Beverley - Version 29.05.10

' Find run of caps

' Alt-I

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

Selection.MoveStart wdWord, 1

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z][A-Z][A-Z][A-Z]"

.MatchWildcards = True

.Execute

End With

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

End Sub

**Sub InitialCapOnly()**

**'** Paul Beverley - Version 29.05.10

' Reduce all-capitals to initial cap

' Crtl-Alt-I

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

Selection.MoveStart wdCharacter, 1

Selection.MoveEnd wdWord, 1

Selection.MoveStart wdWord, -1

Selection.Range.Case = wdTitleWord

Selection.Start = Selection.End

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z][A-Z][A-Z][A-Z]"

.MatchWildcards = True

.Execute

End With

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

End Sub

**Sub LowercaseGlobal()**

**'** Paul Beverley - Version 10.02.11

' Lowercase this phrase throughout text

myHighlight = True

myHighlightColour = wdTurquoise

myOldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myHighlightColour

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

myText = Trim(Selection)

Selection.Start = Selection.End

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText

.Wrap = wdFindContinue

.Replacement.Text = LCase(myText)

If myHighlight = True Then .Replacement.Highlight = True

.Forward = True

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = myOldColour

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub SelectWord()**

**'** Paul Beverley - Version 24.05.24

' Selects current word, and then those before it

If Selection.Start = Selection.End Then

Selection.Expand wdWord

lenNow = Len(Selection)

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

If Right(Selection, 2) = ChrW(8217) & "s" Then

Selection.MoveEnd , -2

End If

If Len(Selection) <> lenNow Then Exit Sub

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.MoveStart , -1

preChar = rng.Text

If LCase(preChar) = UCase(preChar) Then Exit Sub

End If

Selection.MoveStart , -1

preChar = Left(Selection, 1)

Selection.MoveStart , 1

If preChar = "," Or preChar = " " Then

Selection.MoveStart , -1

Exit Sub

End If

Selection.MoveStart wdWord, -1

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End Sub

**Sub SelectSentence()**

' Paul Beverley - Version 12.11.22

' Selects the current sentence + next sentence right

If Selection.Start = Selection.End Then

Selection.Expand wdSentence

Exit Sub

End If

Set rng = Selection.Range.Duplicate

If InStr(rng, vbCr) = 0 Then

rng.Collapse wdCollapseEnd

rng.Expand wdSentence

Selection.End = rng.End

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Expand wdSentence

Selection.Start = rng.Start

Else

rng.MoveStart , -1

If Left(rng, 1) = vbCr Then

rng.Collapse wdCollapseEnd

rng.Expand wdSentence

Selection.End = rng.End

Else

rng.Collapse wdCollapseStart

rng.Expand wdSentence

Selection.Start = rng.Start

End If

End If

End Sub

**Sub SelectParagraph()**

**'** Paul Beverley - Version 24.11.10

' Selects current paragraph

Selection.Expand wdParagraph

End Sub

**Sub SelectCurrentPage()**

**'** Paul Beverley - Version 26.01.11

' Select the current page

Selection.GoTo What:=wdGoToPage, Which:=wdGoToNext, Count:=1

endHere = Selection.Start

Selection.GoTo What:=wdGoToPage, Which:=wdGoToPrevious, Count:=1

Selection.End = endHere

End Sub

**Sub SelectTheseWords()**

' Paul Beverley - Version 12.12.11

' Extend selection to whole words

Selection.MoveEnd Unit:=wdWord, Count:=1

startWas = Selection.Start

Selection.MoveStart Unit:=wdWord, Count:=-1

startNow = Selection.Start

Selection.MoveStart Unit:=wdWord, Count:=1

If Selection.Start <> startWas Then Selection.Start = startNow

Do While InStr(ChrW(8217) & " ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End Sub

**Sub DeleteOneWord()**

**'** Paul Beverley - Version 26.05.23

' Deletes current word, but no punctuation (was called DeleteWord)

If InStr(",.!;:" & vbCr, Selection) > 0 Then Selection.MoveLeft , 1

If Selection = " " Th0en Selection.MoveRight , 1

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

rng.MoveEnd , 1

nextChar = Right(rng, 1)

If nextChar <> " " Then nextChar = "."

rng.MoveStart , -1

prevChar = Left(rng, 1)

If prevChar <> " " Then prevChar = "X"

rng.MoveStart , -1

prevPrevChar = Left(rng, 1)

If prevPrevChar <> " " Then prevPrevChar = "X"

myTest = prevPrevChar & prevChar & nextChar

Select Case myTest

Case "X ": rng.MoveStart , 2

Case " ": rng.MoveStart , 1

Case "X .": rng.MoveStart , 1

rng.MoveEnd , -1

Case " X.": rng.MoveStart , 2

rng.MoveEnd , 1

Case " .": rng.MoveStart , 1

rng.MoveEnd , -1

Case " X ": rng.MoveStart , 2

rng.MoveEnd , -1

Case "XX ": rng.MoveStart , 2

Case "XX.": rng.MoveStart , 2

rng.MoveEnd , -1

Case Else: Beep: myTest = ""

End Select

If myTest > "" Then rng.Delete

If Selection = " " Then

Selection.MoveEnd , 1

Selection.Delete

End If

End Sub

**Sub DeleteRestOfSentence()**

' Paul Beverley - Version 14.01.21

' Deletes from end of current word to the end of the sentence

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

myEnd = Selection.End

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[.\!\?^13]"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

DoEvents

End With

Selection.Collapse wdCollapseStart

Selection.Start = myEnd

qt = Selection.Characters(2)

If qt = ChrW(8217) Or qt = ChrW(8221) Then

Selection.Delete

Selection.MoveRight , 1

Selection.TypeText Text:=qt

Else

Selection.Delete

End If

End Sub

**Sub DeleteRestOfSentence2()**

**'** Paul Beverley - Version 22.10.24

' Deletes to the end of the sentence, including the current word

Selection.Expand wdWord

Selection.MoveStart , -1

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.MoveStart , -1

rng.Select

If UCase(rng) = LCase(rng) Then rng.Delete

myEnd = rng.End

Selection.Collapse wdCollapseStart

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[.\!\?^13]"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

DoEvents

End With

Selection.Collapse wdCollapseStart

Selection.start = myEnd

qt = Selection.Characters(2)

If qt = ChrW(8217) Or qt = ChrW(8221) Then

Selection.Delete

Selection.MoveRight , 1

Selection.TypeText Text:=qt

Else

Selection.Delete

End If

End Sub

**Sub DeleteToNextPunctuation()**

' Paul Beverley - Version 20.07.22

' Deletes from end of current word to the next punctuation mark

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

If UCase(Selection) = LCase(Selection) Then \_

Selection.Collapse wdCollapseStart

myEnd = Selection.End

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[:;,.\!\?^13]"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

DoEvents

End With

If Selection.Start = myEnd Then

Selection.Collapse wdCollapseEnd

Selection.Find.Execute

End If

Selection.Collapse wdCollapseStart

Selection.Start = myEnd

qt = Selection.Characters(2)

If qt = ChrW(8217) Or qt = ChrW(8221) Then

Selection.Delete

Selection.MoveRight , 1

Selection.TypeText Text:=qt

Else

Selection.Delete

End If

End Sub

**Sub DeleteRestOfLine()**

' Paul Beverley - Version 09.06.17

' Deletes from the beginning of current word to the end of the line

Selection.Expand wdWord

Selection.MoveStartUntil cset:=" ", Count:=wdBackward

Selection.MoveStart , -1

Selection.Collapse wdCollapseStart

Selection.MoveEndUntil cset:=vbCr, Count:=wdForward

Selection.Delete

End Sub

**Sub FinalPhraseMoveForward()**

' Paul Beverley - Version 11.08.22

' Cuts the final phrase of a sentence and pulls it to just after the cursor-word

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Set rng = Selection.Range.Duplicate

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[!,]@."

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

rng.MoveEnd , -1

rng.Cut

rng.MoveStart , -1

rng.Delete

If Selection <> "," Then Selection.TypeText Text:=" "

Selection.Paste

End Sub

**Sub FinalCharDelete()**

' Paul Beverley - Version 24.09.22

' Removes the final character or punct off a word

Selection.Collapse wdCollapseEnd

If UCase(Selection) = LCase(Selection) Then Selection.MoveLeft , 1

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Selection.TypeBackspace

End Sub

**Sub PunctuationOffRight()**

' Paul Beverley - Version 16.01.21

' Removes the punctuation off a word end (and quote off start)

deleteLHandQuote = True

' Check first if there's a quote at the left

gotLeftChar = False

If deleteLHandQuote = True Then

Selection.Expand wdWord

If LCase(Selection) = UCase(Selection) Then

Selection.MoveLeft , 2

Selection.Expand wdWord

End If

Selection.Collapse wdCollapseStart

Selection.MoveStart , -1

If InStr(ChrW(8216) & ChrW(8220) & "('""", Selection) > 0 Then

Selection.Delete

gotLeftChar = True

End If

Selection.MoveRight , 2

End If

Selection.Expand wdWord

myLen = Len(Selection)

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

If Len(Selection) > 1 Or Len(Selection) < myLen Then

Selection.Collapse wdCollapseEnd

Selection.MoveEnd , 1

End If

If Selection = "-" And gotLeftChar = True Then

Selection.MoveRight , 2

Else

If Selection <> " " Then

Selection.Delete

Selection.Collapse wdCollapseEnd

Else

Selection.Collapse wdCollapseStart

Selection.MoveStart , -1

Selection.Delete

End If

End If

End Sub

**Sub QuotesOffBothEndsDouble()**

' Paul Beverley - Version 03.09.22

' Removes double quotations marks from both ends of a test

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8220)

.Wrap = wdFindContinue

.Forward = False

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

DoEvents

End With

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8221)

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

DoEvents

End With

rng.Select

End Sub

**Sub QuotesOffBothEndsDoubleAddItalic()**

' Paul Beverley - Version 03.02.23

' Removes double quotations marks from both ends of a text + italicises

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8220)

.Wrap = wdFindStop

.Forward = False

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

DoEvents

End With

myStart = rng.Start

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8221)

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

DoEvents

End With

rng.Start = myStart

rng.Style = "i"

rng.Collapse wdCollapseEnd

rng.Select

End Sub

**Sub QuotesOffBothEndsSingle()**

' Paul Beverley - Version 03.09.22

' Removes single quotations marks from both ends of a test

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8216)

.Wrap = wdFindContinue

.Forward = False

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

DoEvents

End With

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8217)

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

DoEvents

End With

rng.Select

End Sub

**Sub PunctuationOffNearHere()**

' Paul Beverley - Version 03.11.17

' Removes the punctuation near the cursor

ignoreThese = " )(][/0123456789"

rngForward = 3

rngBackward = 3

Selection.Collapse wdCollapseStart

Selection.MoveStart , -rngBackward

Selection.MoveEnd , rngForward

For i = 1 To Len(Selection)

ch = Selection.Characters(i)

If LCase(ch) = UCase(ch) And InStr(ignoreThese, ch) = 0 Then

gotOne = True

Exit For

Else

gotOne = False

End If

Next i

If gotOne = True Then

Selection.Characters(i).Select

Selection.Delete

Else

Beep

End If

Selection.Collapse wdCollapseEnd

End Sub

**Sub ScareQuoteAdd()**

' Paul Beverley - Version 29.05.21

' Add single quotes round a word

useGermanQuotes = False

Selection.Expand wdWord

If Right(Selection, 1) = " " Then Selection.MoveEnd , -1

If useGermanQuotes = True Then

Selection.InsertAfter Text:=ChrW(8216)

Selection.InsertBefore ChrW(8218)

Else

Selection.InsertBefore Text:=ChrW(8216)

Selection.InsertAfter Text:=ChrW(8217)

End If

Selection.Collapse wdCollapseEnd

End Sub

**Sub DoubleQuotesSingleTopical()**

**'** Paul Beverley - Version 20.07.23

' Changes double quotes around current text to singles

myRange = 60

' doUSpunctuation = False

doUSpunctuation = True

Selection.MoveStartUntil cset:=ChrW(8220) & """", count:=wdBackward

If Len(Selection) > myRange Then

Beep

Exit Sub

End If

Selection.Collapse wdCollapseStart

Selection.MoveStart , -1

Selection.TypeText Text:=ChrW(8216)

Selection.MoveEndUntil cset:=ChrW(8221) & """", count:=wdForward

If Len(Selection) > myRange Then

Beep

Exit Sub

End If

Selection.Collapse wdCollapseEnd

Selection.MoveEnd , 1

Selection.Delete

If doUSpunctuation = True Then

Selection.MoveEnd , 1

If InStr(".,", Selection.Text) > 0 Then

Selection.Collapse wdCollapseEnd

Else

Selection.Collapse wdCollapseStart

End If

End If

Selection.TypeText Text:=ChrW(8217)

End Sub

**Sub SingleQuotesDoubleTopical()**

**'** Paul Beverley - Version 20.07.23

' Changes single quotes around current text to doubles

myRange = 60

' doUSpunctuation = False

doUSpunctuation = True

Set rng = Selection.Range.Duplicate

Selection.MoveStartUntil cset:=ChrW(8216), count:=wdBackward

If Len(Selection) > myRange Then

Beep

Exit Sub

End If

Selection.Collapse wdCollapseStart

Selection.MoveStart , -1

Selection.TypeText Text:=ChrW(8220)

rng.Select

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "['" & ChrW(8217) & "][!a-z]"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

End With

Selection.MoveEnd , -1

Selection.Delete

If doUSpunctuation = True Then

Selection.MoveEnd , 1

If InStr(".,", Selection.Text) > 0 Then

Selection.Collapse wdCollapseEnd

Else

Selection.Collapse wdCollapseStart

End If

End If

Selection.TypeText Text:=ChrW(8221)

End Sub

**Sub NonCurlyApostrophe()**

**'** Paul Beverley - Version 01.01.10

' Add non-curly single quote

Options.AutoFormatReplaceQuotes = False

Selection.TypeText Text:="'"

Options.AutoFormatReplaceQuotes = True

End Sub

**Sub NonCurlyQuote()**

**'** Paul Beverley - Version 01.01.10

' Add non-curly double quote

Options.AutoFormatReplaceQuotes = False

Selection.TypeText Text:=""""

Options.AutoFormatReplaceQuotes = True

End Sub

**Sub SinglePrime()**

' Paul Beverley - Version 19.02.16

' Adds single prime

myFont = Selection.Font.Name

' myFont = "Times New Roman"

Selection.InsertSymbol Font:=myFont, CharacterNumber:=8242, unicode:=True

End Sub

**Sub DoublePrime()**

' Paul Beverley - Version 19.02.16

' Adds double prime

myFont = Selection.Font.Name

' myFont = "Times New Roman"

Selection.InsertSymbol Font:=myFont, CharacterNumber:=8243, unicode:=True

End Sub

**Sub QuoteCopier()**

' Paul Beverley - Version 24.09.18

' Copies text from one quote pair to the next

If Selection.Text = " " Then

Selection.MoveEnd , 1

Selection.Delete

End If

Selection.MoveEndUntil cset:=ChrW(8217), Count:=wdForward

Selection.MoveStartUntil cset:=ChrW(8216), Count:=wdBackward

Selection.Copy

Selection.Collapse wdCollapseEnd

Selection.MoveRight , 1

Selection.MoveStartUntil cset:=ChrW(8216), Count:=wdForward

Selection.MoveRight , 1

hereNow = Selection.Start

Selection.Paste

Selection.Start = hereNow

Selection.Collapse wdCollapseStart

End Sub

**Sub MoveToNextQuote()**

' Paul Beverley - Version 25.09.18

' Moves cursor to the next quote pair

Selection.MoveEndUntil cset:=ChrW(8216), Count:=wdForward

Selection.MoveEnd , 1

Selection.MoveEndUntil cset:=ChrW(8217), Count:=wdForward

Selection.Collapse wdCollapseEnd

End Sub

**Sub TypeTimesX()**

' Paul Beverley - Version 25.09.18

' Types '(×2)' then moves back to the number, ready to increase it

Selection.Expand wdWord

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:="(×2) "

Selection.MoveLeft , 3

End Sub

**Sub DeleteSentenceAfterQuote()**

' Paul Beverley - Version 25.09.18

' Deletes rest of sentence after current quote

Set rng = ActiveDocument.Content

rng.Start = Selection.Start - 2

rng.End = Selection.End + 2

If InStr(rng, ChrW(215)) > 0 Then

myTarget = ")"

Else

myTarget = ChrW(8217)

End If

Selection.MoveEndUntil cset:=myTarget, Count:=wdForward

Selection.Collapse wdCollapseEnd

Selection.MoveRight , 1

Selection.MoveEndUntil cset:=".", Count:=wdForward

Selection.Delete

End Sub

**Sub** **SwapCharacters()**

' Paul Beverley - Version 06.10.21

' Switches characters either side of caret

Selection.Collapse wdCollapseStart

Selection.MoveStart , -1

Selection.MoveEnd , 1

txt = Selection.Text

Selection.Text = Right(txt, 1) & Left(txt, 1)

Selection.MoveEnd , -1

Selection.MoveStart , 1

End Sub

**Sub SwapPunctuation()**

**'** Version 25.05.24

' Swaps the next pair of punctuation marks

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Start = Selection.Start

For i = 1 To Len(rng) - 1

ch1 = Mid(rng, i, 1)

If (UCase(ch1) = LCase(ch1) And ch1 <> " ") Then

ch2 = Mid(rng, i + 1, 1)

If (UCase(ch2) = LCase(ch2) And ch2 <> " ") Then Exit For

End If

Next i

Selection.MoveStart , i - 1

Selection.End = Selection.Start + 2

Selection.Delete

Selection.TypeText Text:=ch2 & ch1

End Sub

**Sub SwapPreviousCharacters()**

' Paul Beverley - Version 06.10.21

' Switches the two chars in front of the caret

Selection.Collapse wdCollapseStart

Selection.MoveStart , -2

txt = Selection.Text

Selection.Text = Right(txt, 1) & Left(txt, 1)

Selection.MoveStart , 2

End Sub

**Sub SwapWords()**

' Paul Beverley - Version 14.03.23

' Swaps adjacent words

' Turn off "smart" (ha!) cut and paste

mySmart = Options.SmartCutPaste

Options.SmartCutPaste = False

Selection.Expand wdWord

Selection.Cut

Selection.MoveRight wdWord, 1

Selection.MoveStart , -1

ch = Asc(Selection)

Selection.Collapse wdCollapseEnd

If ch = 32 Then

Selection.Paste

Else

Selection.TypeText Text:=" "

Selection.Paste

ch = Right(Selection, 1)

Selection.MoveStart , -1

Selection.Delete

End If

' turn it back on if the user wants it

Options.SmartCutPaste = mySmart

End Sub

**Sub SwapThreeWords()**

' Paul Beverley - Version 14.01.21

' Swaps three adjacent words (A and B to B and A)

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Set rng1 = Selection.Range.Duplicate

Selection.Collapse wdCollapseEnd

Selection.MoveRight wdWord, 2

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Set rng2 = Selection.Range.Duplicate

Selection.Collapse wdCollapseEnd

Selection.Range.FormattedText = rng1.FormattedText

rng1.Select

Selection.Collapse wdCollapseEnd

Selection.Range.FormattedText = rng2.FormattedText

rng2.Delete

rng1.Delete

End Sub

**Sub Pluralise()**

' Paul Beverley - Version 15.07.22

' Tries to make the current word plural

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

MyWord = Selection

Selection.Collapse wdCollapseEnd

Select Case Right(MyWord, 2)

Case "ch":

Selection.TypeText Text:="es"

Case "sh":

Selection.TypeText Text:="es"

Case "oo":

Selection.TypeText Text:="s"

Case "ey":

Selection.MoveStart , -2

Selection.TypeText Text:="ies"

Case Else

Select Case Right(MyWord, 1)

Case "o":

Selection.TypeText Text:="es"

Case "y":

Selection.MoveStart , -1

Selection.TypeText Text:="ies"

Case "s":

Selection.TypeText Text:="es"

Case Else

Selection.TypeText Text:="s"

End Select

End Select

End Sub

**Sub AbbrSwap()**

' Paul Beverley - Version 24.11.15

' Swap abbreviation into or out of brackets

addPreEditCodes = False

' Find the start ...

Selection.Expand wdWord

Selection.MoveStart , -1

If Asc(Selection) <> Asc("<") Then Selection.MoveStart , 1

startHere = Selection.Start

' ... and the end of the words (close parenthesis)

Selection.MoveEndUntil cset:=ChrW(8217) & ")", Count:=wdForward

Selection.MoveEnd , 1

allWords = Selection

' find the open parenthesis, to find the two bits of text to swap

parenPos = InStr(allWords, "(")

leftWords = Left(allWords, parenPos - 2)

rightWords = Mid(allWords, parenPos + 1, Len(allWords) - parenPos - 1)

' swap them round and type them back in

If addPreEditCodes = True And InStr(leftWords, ">") = 0 Then

If Len(rightWords) > Len(leftWords) Then

newWords = "<termDef>" & rightWords & "</termDef> (<abbrev>" & leftWords + "</abbrev>)"

Else

newWords = "<termDef>" & leftWords & "</termDef> (<abbrev>" & rightWords + "</abbrev>)"

End If

Else

newWords = rightWords + " (" + leftWords + ")"

End If

Selection.TypeText Text:=newWords

End Sub

**Sub PunctuationSwap()**

' Paul Beverley - Version 03.06.22

' Swaps the next punctuation mark

mySwaps = ".|, ,|. ;|: :|; ?|! !|? " & \_

" (|[ )|] [|( ]|) "

doQuoteSwap = True

doCaseSwap = ",.;:"

Application.ScreenUpdating = False

On Error GoTo ReportIt

If doQuoteSwap = True Then

mySwaps = mySwaps & """|' '|"" " & ChrW(8216) & "|" & ChrW(8220) \_

& " " & ChrW(8220) & "|" & ChrW(8216) & " "

mySwaps = mySwaps & " " & ChrW(8217) & "|" & ChrW(8221) \_

& " " & ChrW(8221) & "|" & ChrW(8217) & " "

End If

If Selection.Start <> Selection.End And InStr(doCaseSwap, \_

Selection) > 0 Then

Selection.Move wdWord, 1

Selection.MoveEnd , 1

If UCase(Selection) = LCase(Selection) Then

Selection.MoveEnd , 1

Selection.MoveStart , 1

End If

If Asc(Selection) > 96 Then

Selection = UCase(Selection)

Else

Selection = LCase(Selection)

End If

Selection.Collapse wdCollapseEnd

Application.ScreenUpdating = True

Exit Sub

End If

mySplit = Split(mySwaps, "|")

myTargets = ""

For i = 0 To UBound(mySplit)

myTargets = myTargets & Right(mySplit(i), 1)

Next i

myTargets = Trim(myTargets)

theEnd = ActiveDocument.Content.End

myLen = 100

If Selection.End + myLen > theEnd Then

myLen = theEnd - Selection.End

End If

Selection.End = Selection.Start + myLen

myText = Selection.Text

For i = 1 To myLen

ch = Mid(myText, i, 1)

If InStr(myTargets, ch) > 0 Then

ptr = InStr(mySwaps, ch & "|")

If ch = ChrW(8217) Then

ch2 = Mid(Selection.Text, i + 1, 1)

If InStr("tsvlr", ch2) > 0 Then ptr = 0

End If

If ptr > 0 Then

Selection.Start = Selection.Start + i - 1

Selection.End = Selection.Start + 1

newChar = Mid(mySwaps, ptr + 2, 1)

Selection.Text = newChar

Application.ScreenUpdating = True

If InStr(doCaseSwap, ch) = 0 Then \_

Selection.Collapse wdCollapseEnd

Exit Sub

End If

End If

Next i

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub PunctuationSwapInDialogue()**

' Paul Beverley - Version 03.06.22

' Swaps the next punctuation mark

mySwaps = ".|, ,|. ;|: :|; ?|! !|? " & \_

" (|[ )|] [|( ]|) "

doQuoteSwap = False

doCaseSwap = ",.;:"

Application.ScreenUpdating = False

On Error GoTo ReportIt

If doQuoteSwap = True Then

mySwaps = mySwaps & """|' '|"" " & ChrW(8216) & "|" & ChrW(8220) \_

& " " & ChrW(8220) & "|" & ChrW(8216) & " "

mySwaps = mySwaps & " " & ChrW(8217) & "|" & ChrW(8221) \_

& " " & ChrW(8221) & "|" & ChrW(8217) & " "

End If

If Selection.Start <> Selection.End And InStr(doCaseSwap, \_

Selection) > 0 Then

Selection.Move wdWord, 1

Selection.MoveEnd , 1

If UCase(Selection) = LCase(Selection) Then

Selection.MoveEnd , 1

Selection.MoveStart , 1

End If

If Asc(Selection) > 96 Then

Selection = UCase(Selection)

Else

Selection = LCase(Selection)

End If

Selection.Collapse wdCollapseEnd

Application.ScreenUpdating = True

Exit Sub

End If

mySplit = Split(mySwaps, "|")

myTargets = ""

For i = 0 To UBound(mySplit)

myTargets = myTargets & Right(mySplit(i), 1)

Next i

myTargets = Trim(myTargets)

theEnd = ActiveDocument.Content.End

myLen = 100

If Selection.End + myLen > theEnd Then

myLen = theEnd - Selection.End

End If

Selection.End = Selection.Start + myLen

myText = Selection.Text

For i = 1 To myLen

ch = Mid(myText, i, 1)

If InStr(myTargets, ch) > 0 Then

ptr = InStr(mySwaps, ch & "|")

If ch = ChrW(8217) Then

ch2 = Mid(Selection.Text, i + 1, 1)

If InStr("tsvlr", ch2) > 0 Then ptr = 0

End If

If ptr > 0 Then

Selection.Start = Selection.Start + i - 1

Selection.End = Selection.Start + 1

newChar = Mid(mySwaps, ptr + 2, 1)

Selection.TypeText Text:=newChar

Application.ScreenUpdating = True

If InStr(doCaseSwap, ch) = 0 Then \_

Selection.Collapse wdCollapseEnd

Exit Sub

End If

End If

Next i

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub FullPoint()**

**'** Paul Beverley - Version 09.11.24

' Makes adjacent words into sentence end

newBit = ". "

myQuotes = Chr(34) & Chr(39) & ChrW(8220) & ChrW(8216)

oldBits = ";:.," & ChrW(8211) & ChrW(8211)

Set rng = Selection.Range.Duplicate

rng.MoveEnd , 50

For i = 1 To rng.Characters.Count

If InStr(oldBits, rng.Characters(i)) > 0 Then Exit For

Next i

If i > 50 Then

Beep

MsgBox "No relevant punctuation found."

Exit Sub

End If

rng.Characters(i - 2).Select

While (LCase(Selection) <> UCase(Selection))

Selection.MoveRight 1

DoEvents

Wend

myStart = Selection.start

If Selection = ChrW(8217) Then myStart = myStart + 1

Do

Selection.MoveRight 1

DoEvents

If Selection.start = ActiveDocument.Content.End - 1 \_

Then Beep: Exit Sub

Loop Until LCase(Selection) <> UCase(Selection) Or Asc(Selection) = 1

myEnd = Selection.start

If UCase(Selection) <> Selection Then

' It needs uppercasing

Selection.start = Selection.start - 1

preChar = Selection

Selection.MoveStart 1

Selection.MoveEnd 1

newLetter = UCase(Selection)

If InStr(myQuotes, preChar) > 0 Then

Selection.Delete

Selection.TypeText newLetter

Selection.End = myEnd - 1

Else

newBit = newBit & newLetter

End If

Selection.start = myStart

Else

newBit = Trim(newBit)

Selection.MoveLeft 1

End If

Selection.start = myStart

If Right(Selection, 1) = " " And Len(Selection) > 1 Then

Selection.End = Selection.End - 1

newBit = Trim(newBit)

End If

If Selection.End <> myStart Then Selection.Delete

Selection.TypeText newBit

End Sub

**Sub Period()**

**'** Paul Beverley - Version 09.11.24

' Makes adjacent words into sentence end

newBit = ". "

myQuotes = Chr(34) & Chr(39) & ChrW(8220) & ChrW(8216)

oldBits = ";:.," & ChrW(8211) & ChrW(8211)

Set rng = Selection.Range.Duplicate

rng.MoveEnd , 50

For i = 1 To rng.Characters.Count

If InStr(oldBits, rng.Characters(i)) > 0 Then Exit For

Next i

If i > 50 Then

Beep

MsgBox "No relevant punctuation found."

Exit Sub

End If

rng.Characters(i - 2).Select

While (LCase(Selection) <> UCase(Selection))

Selection.MoveRight 1

DoEvents

Wend

myStart = Selection.start

If Selection = ChrW(8217) Then myStart = myStart + 1

Do

Selection.MoveRight 1

DoEvents

If Selection.start = ActiveDocument.Content.End - 1 \_

Then Beep: Exit Sub

Loop Until LCase(Selection) <> UCase(Selection) Or Asc(Selection) = 1

myEnd = Selection.start

If UCase(Selection) <> Selection Then

' It needs uppercasing

Selection.start = Selection.start - 1

preChar = Selection

Selection.MoveStart 1

Selection.MoveEnd 1

newLetter = UCase(Selection)

If InStr(myQuotes, preChar) > 0 Then

Selection.Delete

Selection.TypeText newLetter

Selection.End = myEnd - 1

Else

newBit = newBit & newLetter

End If

Selection.start = myStart

Else

newBit = Trim(newBit)

Selection.MoveLeft 1

End If

Selection.start = myStart

If Right(Selection, 1) = " " And Len(Selection) > 1 Then

Selection.End = Selection.End - 1

newBit = Trim(newBit)

End If

If Selection.End <> myStart Then Selection.Delete

Selection.TypeText newBit

End Sub

**Sub Comma()**

**'** Paul Beverley - Version 09.11.24

' Make adjacent words into comma separated

newBit = ", "

myQuotes = Chr(34) & Chr(39) & ChrW(8220) & ChrW(8216)

oldBits = ";:.," & ChrW(8211) & ChrW(8211)

Set rng = Selection.Range.Duplicate

rng.MoveEnd , 50

For i = 1 To rng.Characters.Count

If InStr(oldBits, rng.Characters(i)) > 0 Then Exit For

Next i

If i > 50 Then

Beep

MsgBox "No relevant punctuation found."

Exit Sub

End If

rng.Characters(i - 2).Select

While (LCase(Selection) <> UCase(Selection))

Selection.MoveRight 1

DoEvents

Wend

myStart = Selection.start

If Selection = ChrW(8217) Then myStart = myStart + 1

Do

Selection.MoveRight 1

DoEvents

If Selection.start = ActiveDocument.Content.End - 1 \_

Then Beep: Exit Sub

Loop Until LCase(Selection) <> UCase(Selection) Or Asc(Selection) = 1

myEnd = Selection.start

Set rng = ActiveDocument.Content

rng.End = myEnd

rng.start = myStart

wasMiddle = rng

lastChar = Right(rng, 1)

If LCase(Selection) <> Selection Then

' It needs lowercasing

Selection.start = Selection.start - 1

preChar = Selection

Selection.MoveStart 1

Selection.MoveEnd 1

newLetter = LCase(Selection)

If InStr(myQuotes, preChar) > 0 Then

Selection.Delete

Selection.TypeText newLetter

Selection.End = myEnd - 1

Else

newBit = newBit & newLetter

End If

Selection.start = myStart

Else

If lastChar = " " And Len(rng) > 1 Then newBit = Trim(newBit)

Selection.MoveLeft 1

End If

Selection.start = myStart

Selection.Delete

Selection.TypeText newBit

Selection.MoveRight Count:=1

End Sub

**Sub Semicolon()**

**'** Paul Beverley - Version 09.11.24

' Make adjacent words into semicolon separated

newBit = "; "

myQuotes = Chr(34) & Chr(39) & ChrW(8220) & ChrW(8216)

oldBits = ";:.," & ChrW(8211) & ChrW(8211)

Set rng = Selection.Range.Duplicate

rng.MoveEnd , 50

For i = 1 To rng.Characters.Count

If InStr(oldBits, rng.Characters(i)) > 0 Then Exit For

Next i

If i > 50 Then

Beep

MsgBox "No relevant punctuation found."

Exit Sub

End If

rng.Characters(i - 2).Select

While (LCase(Selection) <> UCase(Selection))

Selection.MoveRight 1

DoEvents

Wend

myStart = Selection.start

If Selection = ChrW(8217) Then myStart = myStart + 1

Do

Selection.MoveRight 1

DoEvents

Loop Until LCase(Selection) <> UCase(Selection) Or Asc(Selection) = 1

myEnd = Selection.start

Set rng = ActiveDocument.Content

rng.End = myEnd

rng.start = myStart

wasMiddle = rng

lastChar = Right(rng, 1)

If LCase(Selection) <> Selection Then

' It needs lowercasing

Selection.start = Selection.start - 1

preChar = Selection

Selection.MoveStart 1

Selection.MoveEnd 1

newLetter = LCase(Selection)

If InStr(myQuotes, preChar) > 0 Then

Selection.Delete

Selection.TypeText Text:=newLetter

Selection.End = myEnd - 1

Else

newBit = newBit & newLetter

End If

Selection.start = myStart

Else

If lastChar = " " And Len(rng) > 1 Then newBit = Trim(newBit)

Selection.MoveLeft 1

End If

Selection.start = myStart

Selection.Delete

Selection.TypeText Text:=newBit

Selection.MoveRight Count:=2

End Sub

**Sub Colon()**

**'** Paul Beverley - Version 09.11.24

' Makes adjacent words into colon separated

uppercaseAfter = False

newBit = ": "

myQuotes = Chr(34) & Chr(39) & ChrW(8220) & ChrW(8216)

oldBits = ";:.," & ChrW(8211) & ChrW(8211)

Set rng = Selection.Range.Duplicate

rng.MoveEnd , 50

For i = 1 To rng.Characters.Count

If InStr(oldBits, rng.Characters(i)) > 0 Then Exit For

Next i

If i > 50 Then

Beep

MsgBox "No relevant punctuation found."

Exit Sub

End If

rng.Characters(i - 2).Select

While (LCase(Selection) <> UCase(Selection))

Selection.MoveRight 1

DoEvents

Wend

myStart = Selection.start

If Selection = ChrW(8217) Then myStart = myStart + 1

Do

Selection.MoveRight 1

DoEvents

If Selection.start = ActiveDocument.Content.End - 1 \_

Then Beep: Exit Sub

Loop Until LCase(Selection) <> UCase(Selection) Or Asc(Selection) = 1

myEnd = Selection.start

Set rng = ActiveDocument.Content

rng.End = myEnd

rng.start = myStart

wasMiddle = rng

lastChar = Right(rng, 1)

If uppercaseAfter = False Then

If LCase(Selection) <> Selection Then

' It needs lowercasing

Selection.start = Selection.start - 1

preChar = Selection

Selection.MoveStart 1

Selection.MoveEnd 1

newLetter = LCase(Selection)

If InStr(myQuotes, preChar) > 0 Then

Selection.Delete

Selection.TypeText newLetter

Selection.End = myEnd - 1

Else

newBit = newBit & newLetter

End If

Selection.start = myStart

Else

If lastChar = " " And Len(rng) > 1 Then newBit = Trim(newBit)

Selection.MoveLeft 1

End If

Else

If UCase(Selection) <> Selection Then

' It needs uppercasing

Selection.start = Selection.start - 1

preChar = Selection

Selection.MoveStart 1

Selection.MoveEnd 1

newLetter = UCase(Selection)

If InStr(myQuotes, preChar) > 0 Then

Selection.Delete

Selection.TypeText newLetter

Selection.End = myEnd - 1

Else

newBit = newBit & newLetter

End If

Selection.start = myStart

Else

If lastChar = " " And Len(rng) > 1 Then newBit = Trim(newBit)

Selection.MoveLeft 1

End If

End If

Selection.start = myStart

Selection.Delete

Selection.TypeText newBit

Selection.MoveRight Count:=1

End Sub

**Sub Dash()**

**'** Paul Beverley - Version 09.11.24

' Makes adjacent words into sentence end

newBit = " " & ChrW(8212) & " "

newBit = ChrW(8212)

newBit = " " & ChrW(8211) & " "

myQuotes = Chr(34) & Chr(39) & ChrW(8220) & ChrW(8216)

oldBits = ";:.," & ChrW(8211) & ChrW(8211)

Set rng = Selection.Range.Duplicate

rng.MoveEnd , 50

For i = 1 To rng.Characters.Count

If InStr(oldBits, rng.Characters(i)) > 0 Then Exit For

Next i

If i > 50 Then

Beep

MsgBox "No relevant punctuation found."

Exit Sub

End If

rng.Characters(i - 2).Select

While (LCase(Selection) <> UCase(Selection))

Selection.MoveRight 1

DoEvents

Wend

myStart = Selection.start

If Selection = ChrW(8217) Then myStart = myStart + 1

Do

Selection.MoveRight 1

DoEvents

If Selection.start = ActiveDocument.Content.End - 1 \_

Then Beep: Exit Sub

Loop Until LCase(Selection) <> UCase(Selection) Or Asc(Selection) = 1

myEnd = Selection.start

If LCase(Selection) <> Selection Then

' It needs lowercasing

Selection.start = Selection.start - 1

preChar = Selection

Selection.MoveStart 1

Selection.MoveEnd 1

newLetter = LCase(Selection)

If InStr(myQuotes, preChar) > 0 Then

Selection.Delete

Selection.TypeText newLetter

Selection.End = myEnd - 1

Else

newBit = newBit & newLetter

End If

Selection.start = myStart

End If

Selection.start = myStart

If Selection.End <> myStart Then Selection.Delete

Selection.TypeText newBit

End Sub

**Sub EmDashUnspaced()**

' Paul Beverley - Version 14.06.21

' Removes punctuation, adds unspaced em dash and lowercases next char

newBit = ChrW(8212)

myPunct = "!?,.:;"

myQuotes = Chr(34) & Chr(39) & ChrW(8220) & ChrW(8216)

Selection.Collapse wdCollapseEnd

Do

Selection.MoveRight 1

DoEvents

Loop Until (InStr(myPunct, Selection) > 0) Or Selection = " "

myStart = Selection.Start

If Selection = ChrW(8217) Then myStart = myStart + 1

Do

Selection.MoveRight 1

DoEvents

Loop Until LCase(Selection) <> UCase(Selection) Or Asc(Selection) = 1

myEnd = Selection.Start

Set rng = ActiveDocument.Content

rng.End = myEnd

rng.Start = myStart

wasMiddle = rng

lastChar = Right(rng, 1)

If LCase(Selection) <> Selection Then

' It needs lowercasing

Selection.Start = Selection.Start - 1

preChar = Selection

Selection.MoveStart 1

Selection.MoveEnd 1

newLetter = LCase(Selection)

If InStr(myQuotes, preChar) > 0 Then

Selection.Delete

Selection.TypeText newLetter

Selection.End = myEnd - 1

Else

newBit = newBit & newLetter

End If

Selection.Start = myStart

End If

Selection.Start = myStart

Selection.Delete

Selection.TypeText newBit

Selection.MoveRight Count:=1

End Sub

**Sub QuestionMark()**

' Paul Beverley - Version 09.11.24

' Makes adjacent words into sentence end

newBit = "? "

myQuotes = Chr(34) & Chr(39) & ChrW(8220) & ChrW(8216)

oldBits = ";:.," & ChrW(8211) & ChrW(8211)

Set rng = Selection.Range.Duplicate

rng.MoveEnd , 50

For i = 1 To rng.Characters.Count

If InStr(oldBits, rng.Characters(i)) > 0 Then Exit For

Next i

If i > 50 Then

Beep

MsgBox "No relevant punctuation found."

Exit Sub

End If

rng.Characters(i - 2).Select

While (LCase(Selection) <> UCase(Selection))

Selection.MoveRight 1

DoEvents

Wend

myStart = Selection.Start

If Selection = ChrW(8217) Then myStart = myStart + 1

Do

Selection.MoveRight 1

DoEvents

Loop Until LCase(Selection) <> UCase(Selection) Or Asc(Selection) = 1

myEnd = Selection.Start

If UCase(Selection) <> Selection Then

' It needs uppercasing

Selection.Start = Selection.Start - 1

preChar = Selection

Selection.MoveStart 1

Selection.MoveEnd 1

newLetter = UCase(Selection)

If InStr(myQuotes, preChar) > 0 Then

Selection.Delete

Selection.TypeText Text:=newLetter

Selection.End = myEnd - 1

Else

newBit = newBit & newLetter

End If

Selection.Start = myStart

Else

newBit = Trim(newBit)

Selection.MoveLeft 1

End If

Selection.Start = myStart

If Right(Selection, 1) = " " And Len(Selection) > 1 Then

Selection.End = Selection.End - 1

newBit = Trim(newBit)

End If

If Selection.End <> myStart Then Selection.Delete

Selection.TypeText Text:=newBit

End Sub

**Sub ExclamationMark()**

' Paul Beverley - Version 09.11.24

' Make adjacent words into sentence end

newBit = "! "

myQuotes = Chr(34) & Chr(39) & ChrW(8220) & ChrW(8216)

oldBits = ";:.," & ChrW(8211) & ChrW(8211)

Set rng = Selection.Range.Duplicate

rng.MoveEnd , 50

For i = 1 To rng.Characters.Count

If InStr(oldBits, rng.Characters(i)) > 0 Then Exit For

Next i

If i > 50 Then

Beep

MsgBox "No relevant punctuation found."

Exit Sub

End If

rng.Characters(i - 2).Select

While (LCase(Selection) <> UCase(Selection))

Selection.MoveRight 1

DoEvents

Wend

myStart = Selection.Start

If Selection = ChrW(8217) Then myStart = myStart + 1

Do

Selection.MoveRight 1

DoEvents

Loop Until LCase(Selection) <> UCase(Selection) Or Asc(Selection) = 1

myEnd = Selection.Start

If UCase(Selection) <> Selection Then

' It needs uppercasing

Selection.Start = Selection.Start - 1

preChar = Selection

Selection.MoveStart 1

Selection.MoveEnd 1

newLetter = UCase(Selection)

If InStr(myQuotes, preChar) > 0 Then

Selection.Delete

Selection.TypeText Text:=newLetter

Selection.End = myEnd - 1

Else

newBit = newBit & newLetter

End If

Selection.Start = myStart

Else

newBit = Trim(newBit)

Selection.MoveLeft 1

End If

Selection.Start = myStart

If Right(Selection, 1) = " " And Len(Selection) > 1 Then

Selection.End = Selection.End - 1

newBit = Trim(newBit)

End If

If Selection.End <> myStart Then Selection.Delete

Selection.TypeText Text:=newBit

End Sub

**Sub FullPointInDialogue()**

' Paul Beverley - Version 17.02.21

' Make adjacent words into sentence end

makeCaseChange = (Selection.Start = Selection.End)

Selection.Expand wdWord

Selection.MoveEnd wdCharacter, 1

wasStart = Selection.Start

wasEnd = Selection.End

newBit = ". "

Selection.Collapse wdCollapseStart

findChars = ".:,;\!\?" & ChrW(8211) & ChrW(8222) & ChrW(8212)

Selection.MoveEndUntil cset:=findChars, Count:=wdForward

Selection.Collapse wdCollapseEnd

Selection.Start = Selection.Start - 1

Selection.End = Selection.End + 3

If Selection.Start > wasEnd Then

Selection.Start = wasStart

Selection.End = wasEnd

spacePos = InStr(Selection, " ")

Selection.Start = wasStart + spacePos - 1

Selection.InsertBefore Trim(newBit)

Selection.Start = wasStart

Selection.Collapse wdCollapseStart

Exit Sub

End If

preChar = Left(Selection, 1)

midChar = Mid(Selection, 3, 1)

postChar = Right(Selection, 1)

If midChar <> " " Then

Selection.MoveEnd wdCharacter, -1

postChar = Right(Selection, 1)

End If

If preChar <> " " Then Selection.MoveStart wdCharacter, 1

' If the middle char is a close quote go past it

If InStr(ChrW(8217) & ChrW(8221) & """'", midChar) > 0 Then

Selection.MoveEnd wdCharacter, 2

newBit = Replace(newBit, " ", postChar & " ")

postChar = Right(Selection, 1)

End If

' If the next char is an open quote go past it

If InStr(ChrW(8216) & ChrW(8220) & """'", postChar) > 0 Then

Selection.MoveEnd wdCharacter, 1

newBit = newBit & postChar

postChar = Right(Selection, 1)

End If

' If the case of the next letter needs changing

If UCase(postChar) <> postChar And makeCaseChange Then

newBit = newBit & UCase(postChar)

Else

Selection.MoveEnd wdCharacter, -1

End If

Selection.Delete

Selection.TypeText Text:=newBit

End Sub

**Sub PeriodInDialogue()**

' Paul Beverley - Version 17.02.21

' Make adjacent words into sentence end

makeCaseChange = (Selection.Start = Selection.End)

Selection.Expand wdWord

Selection.MoveEnd wdCharacter, 1

wasStart = Selection.Start

wasEnd = Selection.End

newBit = ". "

Selection.Collapse wdCollapseStart

findChars = ".:,;\!\?" & ChrW(8211) & ChrW(8222) & ChrW(8212)

Selection.MoveEndUntil cset:=findChars, Count:=wdForward

Selection.Collapse wdCollapseEnd

Selection.Start = Selection.Start - 1

Selection.End = Selection.End + 3

If Selection.Start > wasEnd Then

Selection.Start = wasStart

Selection.End = wasEnd

spacePos = InStr(Selection, " ")

Selection.Start = wasStart + spacePos - 1

Selection.InsertBefore Trim(newBit)

Selection.Start = wasStart

Selection.Collapse wdCollapseStart

Exit Sub

End If

preChar = Left(Selection, 1)

midChar = Mid(Selection, 3, 1)

postChar = Right(Selection, 1)

If midChar <> " " Then

Selection.MoveEnd wdCharacter, -1

postChar = Right(Selection, 1)

End If

If preChar <> " " Then Selection.MoveStart wdCharacter, 1

' If the middle char is a close quote go past it

If InStr(ChrW(8217) & ChrW(8221) & """'", midChar) > 0 Then

Selection.MoveEnd wdCharacter, 2

newBit = Replace(newBit, " ", postChar & " ")

postChar = Right(Selection, 1)

End If

' If the next char is an open quote go past it

If InStr(ChrW(8216) & ChrW(8220) & """'", postChar) > 0 Then

Selection.MoveEnd wdCharacter, 1

newBit = newBit & postChar

postChar = Right(Selection, 1)

End If

' If the case of the next letter needs changing

If UCase(postChar) <> postChar And makeCaseChange Then

newBit = newBit & UCase(postChar)

Else

Selection.MoveEnd wdCharacter, -1

End If

Selection.Delete

Selection.TypeText Text:=newBit

End Sub

**Sub CommaInDialogue()**

' Paul Beverley - Version 17.02.21

' Give adjacent words a comma link

makeCaseChange = (Selection.Start = Selection.End)

Selection.Words(1).Select

Selection.MoveEnd wdCharacter, 1

wasStart = Selection.Start

wasEnd = Selection.End

newBit = ", "

Selection.Collapse wdCollapseStart

findChars = ".:,;\!\?" & ChrW(8211) & ChrW(8222) & ChrW(8212)

Selection.MoveEndUntil cset:=findChars, Count:=wdForward

Selection.Collapse wdCollapseEnd

Selection.Start = Selection.Start - 1

Selection.End = Selection.End + 3

If Selection.Start > wasEnd Then

Selection.Start = wasStart

Selection.End = wasEnd

spacePos = InStr(Selection, " ")

Selection.Start = wasStart + spacePos - 1

Selection.InsertBefore Trim(newBit)

Selection.Start = wasStart

Selection.Collapse wdCollapseStart

Exit Sub

End If

preChar = Left(Selection, 1)

midChar = Mid(Selection, 3, 1)

postChar = Right(Selection, 1)

If midChar <> " " Then

Selection.MoveEnd wdCharacter, -1

postChar = Right(Selection, 1)

End If

If preChar <> " " Then Selection.MoveStart wdCharacter, 1

' If the middle char is a close quote go past it

If InStr(ChrW(8217) & ChrW(8221) & """'", midChar) > 0 Then

Selection.MoveEnd wdCharacter, 2

newBit = Replace(newBit, " ", postChar & " ")

postChar = Right(Selection, 1)

End If

' If the next char is an open quote go past it

If InStr(ChrW(8216) & ChrW(8220) & """'", postChar) > 0 Then

Selection.MoveEnd wdCharacter, 1

newBit = newBit & postChar

postChar = Right(Selection, 1)

End If

' If the case of the next letter needs changing

If LCase(postChar) <> postChar And makeCaseChange Then

newBit = newBit & LCase(postChar)

Else

Selection.MoveEnd wdCharacter, -1

End If

Selection.Delete

Selection.TypeText Text:=newBit

End Sub

**Sub ProperToPronoun()**

' Paul Beverley - Version 02.05.18

' Changes the next proper noun to a personal pronoun

sheHasPriority = True

' sheHasPriority = False

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z][a-z]"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute

End With

Selection.Expand wdWord

Selection.MoveEnd , -1

Selection.Delete

If sheHasPriority Then

Selection.InsertAfter Text:="he"

Selection.InsertBefore Text:="s"

Selection.Collapse wdCollapseEnd

Else

Selection.InsertAfter Text:="she"

Selection.Collapse wdCollapseStart

Selection.MoveEnd , 1

Selection.Delete

End If

End Sub

**Sub NumberToTextUS()**

**'** Paul Beverley - Version 13.07.23

' Converts next number into text, eg "two hundred forty-two"

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

' Find a number (six figures max)

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Text = "[0-9]{1,6}"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

' Create a field containing the digits and a special format code

Selection.Fields.Add Range:=Selection.Range, \_

Type:=wdFieldEmpty, Text:="= " + Selection + " \\* CardText", \_

PreserveFormatting:=True

' Select the field and copy it

Selection.MoveStart , -1

Set rng = Selection.Range.Duplicate

Selection.Fields(1).Unlink

DoEvents

rng.Select

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

Selection.Collapse wdCollapseEnd

Selection.MoveRight wdWord, 1

End Sub

**Sub NumberToTextCMOS()**

' Paul Beverley - Version 29.08.22

' Converts next number into text, using e.g. "fourteen hundred"

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

' Find a number (six figures max)

Selection.End = Selection.Start

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]{1,6}"

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.MatchAllWordForms = False

.Execute

End With

startNum = Selection.Start

' Create a field containing the digits and a special format code

Selection.Fields.Add Range:=Selection.Range, \_

Type:=wdFieldEmpty, Text:="= " + Selection + " \\* CardText", \_

PreserveFormatting:=True

' Select the field and copy it

Selection.MoveStart , -1

txt = Selection.Text

txt = Replace(txt, "-one thousand", "xzxz")

txt = Replace(txt, "one thousand one hundred", "eleven hundred")

txt = Replace(txt, "one thousand two hundred", "twelve hundred")

txt = Replace(txt, "one thousand three hundred", "thirteen hundred")

txt = Replace(txt, "one thousand four hundred", "fourteen hundred")

txt = Replace(txt, "one thousand five hundred", "fifteen hundred")

txt = Replace(txt, "one thousand six hundred", "sixteen hundred")

txt = Replace(txt, "one thousand seven hundred", "seventeen hundred")

txt = Replace(txt, "one thousand eight hundred", "eighteen hundred")

txt = Replace(txt, "one thousand nine hundred", "nineteen hundred")

txt = Replace(txt, "xzxz", "-one thousand")

Selection.Delete

Selection.InsertAfter Text:=txt

Selection.Cut

DoEvents

' Paste the text as unformatted, replacing the field

Selection.PasteSpecial Link:=False, DataType:=wdPasteText, \_

Placement:=wdInLine, DisplayAsIcon:=False

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

Set rng = Selection.Range.Duplicate

rng.MoveStart , -1

ch1 = rng.Text

rng.MoveStart , 1

rng.MoveEnd , 1

ch2 = rng.Text

If (UCase(ch1) <> LCase(ch1)) And (UCase(ch2) <> LCase(ch2)) Then

Selection.TypeText Text:=" "

End If

Set rng = ActiveDocument.Range(Start:=startNum - 1, End:=startNum)

If rng.Text <> " " Then rng.InsertBefore Text:=" "

End Sub

**Sub NumberToTextUK()**

**'** Paul Beverley - Version 13.07.23

' Converts next number into text, eg "two hundred and forty-two"

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

' Find a number (six figures max)

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Text = "[0-9]{1,6}"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

' Create a field containing the digits and a special format code

Selection.Fields.Add Range:=Selection.Range, \_

Type:=wdFieldEmpty, Text:="= " + Selection + " \\* CardText", \_

PreserveFormatting:=True

' Select the field and copy it

Selection.MoveStart , -1

Set rng = Selection.Range.Duplicate

Selection.Fields(1).Unlink

DoEvents

rng.Select

numWords = Selection.Text

If InStr(numWords, "dred") > 0 And Right(numWords, 4) <> "dred" Then numWords = \_

Replace(numWords, "hundred", "hundred and")

If InStr(numWords, "hundred") > 0 And InStr(numWords, "thousand") > 0 Then

numWords = Replace(numWords, "thousand", "thousand,")

Else

If Right(numWords, 4) <> "sand" Then numWords = \_

Replace(numWords, "thousand", "thousand and")

End If

If numWords <> Selection.Text Then Selection.Text = numWords

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

Selection.Collapse wdCollapseEnd

Selection.MoveRight wdWord, 1

End Sub

**Sub NumberToText2()**

**'** Paul Beverley - Version 01.06.22

' Convert next number into text

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

myOptWS = Options.AutoWordSelection

Options.AutoWordSelection = False

' Find a number (six figures max)

Selection.End = Selection.Start

With Selection.Find

.ClearFormatting

.Text = "[0-9]{1,6}"

.Replacement.Text = ""

.MatchWildcards = True

.Forward = True

.Execute

End With

Selection.Collapse Direction:=wdCollapseStart

With Selection.Find

.ClearFormatting

.Text = "[0-9 ,^0160]{1,9}"

.MatchWildcards = True

.Execute

End With

myDigits = Replace(Selection, Chr(160), "")

myDigitsNow = Replace(myDigits, " ", "")

myDigitsFinal = Replace(myDigitsNow, ",", "")

' Create a field containing the digits and a special format code

Selection.Fields.Add Range:=Selection.Range, \_

Type:=wdFieldEmpty, Text:="= " + myDigitsFinal + " \\* CardText", \_

PreserveFormatting:=True

' Select the field and copy it

Selection.MoveStart Unit:=wdCharacter, Count:=-1

Selection.Copy

' Paste the text unformatted, replacing the field

Selection.PasteSpecial Link:=False, DataType:=wdPasteText, \_

Placement:=wdInLine, DisplayAsIcon:=False

If Right(myDigitsNow, 1) = "," Then Selection.TypeText Text:=","

If Right(myDigits, 1) = " " Then Selection.TypeText Text:=" "

' Restore Find to original

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

Options.AutoWordSelection = myOptWS

End Sub

**Sub NumberToTextUK2()**

**'** Paul Beverley - Version 01.06.22

' Convert next number into text

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

myOptWS = Options.AutoWordSelection

Options.AutoWordSelection = False

' Find a number (six figures max)

Selection.End = Selection.Start

With Selection.Find

.ClearFormatting

.Text = "[0-9]{1,6}"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Selection.Collapse Direction:=wdCollapseStart

startHere = Selection.Start

Selection.End = Selection.Start

With Selection.Find

.ClearFormatting

.Text = "[0-9 ,^0160]{1,9}"

.MatchWildcards = True

.Execute

End With

myDigits = Replace(Selection, Chr(160), "")

myDigitsNow = Replace(myDigits, " ", "")

myDigitsFinal = Replace(myDigitsNow, ",", "")

' Create a field containing the digits and a special format code

Selection.Fields.Add Range:=Selection.Range, \_

Type:=wdFieldEmpty, Text:="= " + myDigitsFinal + " \\* CardText", \_

PreserveFormatting:=True

' Select the field and copy it

Selection.MoveStart Unit:=wdCharacter, Count:=-1

Selection.Copy

' Paste the text unformatted, replacing the field

Selection.PasteSpecial Link:=False, DataType:=wdPasteText, \_

Placement:=wdInLine, DisplayAsIcon:=False

Selection.Start = startHere

numWords = Selection

If Right(numWords, 4) <> "dred" Then numWords = \_

Replace(numWords, "hundred", "hundred and")

If InStr(numWords, "hundred") > 0 Then

numWords = Replace(numWords, "thousand", "thousand,")

Else

If Right(numWords, 4) <> "sand" Then numWords = \_

Replace(numWords, "thousand", "thousand and")

End If

Selection.TypeText Text:=numWords

If Right(myDigitsNow, 1) = "," Then Selection.TypeText Text:=","

If Right(myDigits, 1) = " " Then Selection.TypeText Text:=" "

' Restore Find to original

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

Options.AutoWordSelection = myOptWS

End Sub

**Sub ZifferWort()**

' Paul Beverley - Version 06.04.20

' Converts the next number (1 to 12) into German text

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

' Find a number (six figures max)

Selection.End = Selection.Start

With Selection.Find

.ClearFormatting

.Text = "[0-9]{1,5}"

.Replacement.Text = ""

.MatchWildcards = True

.Forward = True

.Execute

End With

myNum = Val(Selection)

Select Case myNum

Case 1: myText = "eins"

Case 2: myText = "zwei"

Case 3: myText = "drei"

Case 4: myText = "vier"

Case 5: myText = "fünf"

Case 6: myText = "sechs"

Case 7: myText = "sieben"

Case 8: myText = "acht"

Case 9: myText = "neun"

Case 10: myText = "zehn"

Case 11: myText = "elf"

Case 12: myText = "zwölf"

Case 13: myText = ""

Case 14: myText = ""

Case 15: myText = ""

Case 16: myText = ""

Case 17: myText = ""

Case 18: myText = ""

Case 19: myText = ""

Case 20: myText = ""

Case Else: myText = ""

End Select

If myText = "" Then

Beep

Selection.Collapse wdCollapseEnd

Else

Selection.TypeText Text:=myText

End If

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

End Sub

**Sub NumberToTextMultiSwitch()**

' Paul Beverley - Version 18.01.20

' Finds a number, then calls MultiSwitch to change it

maxDigits = 2

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[0-9]{1," & Trim(Str(maxDigits)) & "}>"

.Wrap = wdFindStop

.MatchWildcards = True

.Execute

End With

If Selection.Find.Found = True Then

Application.Run macroName:="MultiSwitch"

Else

Beep

End If

End Sub

**Sub NumberToFigure()**

' Paul Beverley - Version 16.01.21

' Converts the next number to a figure

includeTen = False

allNums = ",one,two,three,four,five,six,seven,eight,nine,"

If includeTen = True Then allNums = allNums & ",ten,"

Selection.Expand wdWord

Do

i = i + 1

Selection.MoveRight Unit:=wdWord, Count:=1

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

wd = Trim(Selection)

Loop Until i > 100 Or (Len(wd) > 2 And InStr(allNums, "," & wd & ",") > 0)

If i > 100 Then Beep: Exit Sub

Select Case wd

Case "one": fig = "1"

Case "two": fig = "2"

Case "three": fig = "3"

Case "four": fig = "4"

Case "five": fig = "5"

Case "six": fig = "6"

Case "seven": fig = "7"

Case "eight": fig = "8"

Case "nine": fig = "9"

Case "ten": fig = "10"

End Select

Selection = Replace(Selection, wd, fig)

End Sub

**Sub TextToNumber()**

**'** Paul Beverley - Version 11.03.24

' Finds numbers expressed in words + converts to figures

myUnits = ":one:two:three:four:five:six:seven:eight:nine:ten"

myTens = ":ten:twenty:thirty:forty:fifty:sixty:seventy:eighty:ninety:hundred"

myTeens = ":eleven:twelve:thirteen:fourteen:fifteen:sixteen:seventeen:eighteen:nineteen"

allNumberWords = myUnits & myTens & myTeens & ":a:and:-:"

mySmart = Options.SmartCutPaste

Options.SmartCutPaste = False

Dim wd(6) As String

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

rng.Collapse wdCollapseStart

gottaWord = False

For i = 1 To 50

rng.MoveEnd wdWord, 1

thisWord = LCase(Trim(rng.Words(rng.Words.Count)))

If InStr("aand-", thisWord) = 0 And InStr(allNumberWords, \_

":" & thisWord & ":") > 0 Then

If Right(Trim(rng.Text), 6) = "no-one" Then

gottaWord = False

Else

gottaWord = True

Exit For

End If

End If

Next i

rng.Collapse wdCollapseEnd

rng.MoveEnd wdWord, -1

If gottaWord = False Then

rng.Select

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

gotStart = False

Do While gotStart = False

rng.MoveStart wdWord, -1

If InStr(allNumberWords, ":" & LCase(Trim(rng.Words(1))) & ":") = 0 Then

gotStart = True

rng.MoveStart wdWord, 1

gotStart = True

End If

DoEvents

Loop

gotEnd = False

Do While gotEnd = False

rng.MoveEnd wdWord, 1

lastWord = LCase(Trim(rng.Words(rng.Words.Count)))

If InStr(allNumberWords, ":" & lastWord & ":") = 0 Then

gotEnd = True

rng.MoveEnd wdWord, -1

End If

DoEvents

Loop

fstWd = LCase(Trim(rng.Words(1)))

If InStr("and", fstWd) > 0 Then rng.MoveStart wdWord, 1

' To catch 'a', 'an', and 'and' as final word

lastWd = LCase(Trim(rng.Words(rng.Words.Count)))

If InStr("and", lastWd) > 0 Then rng.MoveEnd wdWord, -1

' Catch missing "and" for American usage, e.g. "two hundred fifty-two"

rText = LCase(rng.Text)

If lastWd <> "hundred" And InStr(rText, "hundred") > 0 And \_

InStr(rText, "hundred and") = 0 Then

rng.Text = Replace(rText, "hundred", "hundred and")

End If

allWords = LCase(rng.Text)

numWords = rng.Words.Count

Dim n(6) As Integer

For i = 1 To numWords

wdPos = InStr(allNumberWords, ":" & LCase(Trim(rng.Words(i))) & ":")

leftWords = Left(allNumberWords, wdPos)

n(i) = Len(leftWords) - Len(Replace(leftWords, ":", ""))

Next

If n(1) = 30 Then n(1) = 1

Select Case numWords

Case 1

myResult = n(1)

If n(1) > 10 Then myResult = 10 \* (n(1) - 10)

If n(1) > 20 Then myResult = n(1) - 10

Case 2

If n(2) = 20 Then ' "hundred"

myResult = n(1) \* 100

Else

myResult = 10 \* (n(1) - 10) + n(2)

If myResult < 21 Then

Beep

rng.Select

Exit Sub

End If

End If

Case 3

myResult = 10 \* (n(1) - 10) + n(3)

If n(2) <> 32 Then ' hyphen

If n(2) = 20 Then

myResult = n(3) + 100 \* n(1)

Else

Beep

rng.Select

Exit Sub

End If

End If

Case 4

If n(2) <> 20 Then ' "hundred"

Beep

rng.Select

Exit Sub

End If

myResult = n(4)

If n(4) > 10 Then myResult = 10 \* (n(4) - 10)

If n(4) > 20 Then myResult = n(4) - 10

myResult = myResult + 100 \* n(1)

Case 5

If n(2) <> 20 Then ' "hundred"

Beep

rng.Select

Exit Sub

End If

myResult = 100 \* n(1) + 10 \* (n(3) - 10) + n(5)

Case 6

If n(2) <> 20 Then ' "hundred"

Beep

rng.Select

Exit Sub

End If

myResult = 100 \* n(1) + 10 \* (n(4) - 10) + n(6)

Case Else

Beep

rng.Select

Exit Sub

End Select

Do While InStr(" ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

rng.Select

Selection.Delete

Selection.TypeText Trim(Str(myResult))

Options.SmartCutPaste = mySmart

End Sub

**Sub NumberIncrement()**

**'** Paul Beverley - Version 20.11.24

' Adds one to the next number after the cursor

myJump = 1

goNext = False

' goNext = True

trackit = True

searchRange = 100

If goNext = True And Selection.start <> \_

Selection.End Then \_

Selection.Collapse wdCollapseStart

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

rng.End = rng.start + searchRange

skipThese = " ." & ChrW(13)

Do While InStr(skipThese, Left(rng, 1)) > 0

rng.MoveStart , 1

Loop

' Move selection to start of any number

If Val(rng) = 0 Then

i = 0

allText = rng.Text

Do

i = i + 1

If i = searchRange Then

Beep

rng.Select

MsgBox ("No number found.")

Exit Sub

End If

myTest = Mid(allText, i)

DoEvents

Loop Until Val(myTest) > 0 And InStr(skipThese, Left(myTest, 1)) = 0

Selection.MoveStart , i - 1

rng.MoveStart , i - 1

End If

' Find end of number and increment (decrease) it

i = 0

allText = rng.Text

Do

i = i + 1

myTest = Mid(allText, i, 1)

DoEvents

Loop Until Val(myTest) = 0 And myTest <> "0"

rng.End = rng.start + i - 1

rng.Select

rng.Text = Trim(Str(Val(rng.Text) + myJump))

If goNext = True Then

Do While Asc(Selection) > 47 And Asc(Selection) < 58

Selection.MoveRight , 1

Loop

Call FindFwd

End If

End Sub

**Sub NumberDecrement()**

**'** Paul Beverley - Version 20.11.24

' Subtracts one from the next number after the cursor

myJump = -1

goNext = False

' goNext = True

trackit = True

searchRange = 100

If goNext = True And Selection.start <> \_

Selection.End Then \_

Selection.Collapse wdCollapseStart

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

rng.End = rng.start + searchRange

skipThese = " ." & ChrW(13)

Do While InStr(skipThese, Left(rng, 1)) > 0

rng.MoveStart , 1

Loop

' Move selection to start of any number

If Val(rng) = 0 Then

i = 0

allText = rng.Text

Do

i = i + 1

If i = searchRange Then

Beep

rng.Select

MsgBox ("No number found.")

Exit Sub

End If

myTest = Mid(allText, i)

DoEvents

Loop Until Val(myTest) > 0 And InStr(skipThese, Left(myTest, 1)) = 0

Selection.MoveStart , i - 1

rng.MoveStart , i - 1

End If

' Find end of number and increment (decrease) it

i = 0

allText = rng.Text

Do

i = i + 1

myTest = Mid(allText, i, 1)

DoEvents

Loop Until Val(myTest) = 0 And myTest <> "0"

rng.End = rng.start + i - 1

rng.Select

rng.Text = Trim(Str(Val(rng.Text) + myJump))

If goNext = True Then

Do While Asc(Selection) > 47 And Asc(Selection) < 58

Selection.MoveRight , 1

Loop

Call FindFwd

End If

End Sub

**Sub LetterIncrement()**

' Paul Beverley - Version 26.04.16

' Find next alpha character and decrement it alphabetically

doTrack = False

myTrack = ActiveDocument.TrackRevisions

If doTrack = False Then ActiveDocument.TrackRevisions = False

Selection.Collapse wdCollapseStart

Do While UCase(ChrW(AscW(Selection))) = LCase(ChrW(AscW(Selection)))

Selection.MoveRight , 1

DoEvents

Loop

Selection.End = Selection.Start + 1

Selection.TypeText Text:=ChrW(AscW(Selection) + 1)

Selection.MoveLeft , 1

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub LetterDecrement()**

' Paul Beverley - Version 26.04.16

' Find next alpha character and increment it alphabetically

doTrack = False

myTrack = ActiveDocument.TrackRevisions

If doTrack = False Then ActiveDocument.TrackRevisions = False

Selection.Collapse wdCollapseStart

Do While UCase(ChrW(AscW(Selection))) = LCase(ChrW(AscW(Selection)))

Selection.MoveRight , 1

DoEvents

Loop

Selection.End = Selection.Start + 1

Selection.TypeText Text:=ChrW(AscW(Selection) - 1)

Selection.MoveLeft , 1

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub Ampersand()**

' Paul Beverley - Version 03.06.22

' Changes next ampersand to 'and'

Selection.Collapse wdCollapseStart

Set rng = Selection.Range.Duplicate

If rng.End + 400 > ActiveDocument.Content.End Then

rng.End = ActiveDocument.Content.End

Else

rng.End = rng.Start + 400

End If

charPos = InStr(rng, "&")

If charPos > 0 Then

Selection.MoveStart , charPos - 1

Selection.End = Selection.Start + 1

Selection.TypeText "and"

Else

Beep

MsgBox ("Can't see an ampersand!")

End If

End Sub

**Sub PerCent()**

**'** Paul Beverley - Version 25.02.13

' Changes percent symbol to words

Do

Selection.MoveRight Unit:=wdCharacter, Count:=1

Loop Until Selection = "%"

Selection.MoveEnd wdCharacter, 1

If Selection.LanguageID = wdEnglishUK Then

Selection.TypeText Text:=" per cent"

Else

Selection.TypeText Text:=" percent"

End If

End Sub

**Sub VerbChanger()**

**'** Paul Beverley - Version 23.09.24

' Changes "(to) splodge" <-> "(of/for) splodging"

doBeep = False

preps = "to of for "

useOf = (Selection.start = Selection.End)

If UCase(Selection) = LCase(Selection) Then Selection.MoveLeft , 1

If UCase(Selection) = LCase(Selection) Then Selection.MoveLeft , 1

Selection.Expand wdWord

wasWord = Selection

' If it's a preposition, change it

If InStr(preps, wasWord) > 0 Then

If wasWord = "to " Then

If useOf = True Then

newWord = "of "

Else

newWord = "for "

End If

End If

If wasWord = "for " Or wasWord = "of " Then

newWord = "to "

End If

Selection.TypeText newWord

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & ChrW(39) & " ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

theWord = Selection

If Len(Selection) > 4 Then fiveChars = Right(Selection, 5)

If Len(fiveChars) > 2 Then twoChars = Left(fiveChars, 2)

If Right(Selection, 3) = "ing" Then

' Remove 'ing' and try adding an 'e'

Selection.start = Selection.End - 3

Selection.Delete

If theWord = "using" Then

Selection.TypeText "e"

Else

spellOK = Application.CheckSpelling(Left(theWord, Len(theWord) - 3))

If spellOK = False Then

If InStr("nn,rr,ll,tt,pp", twoChars) > 0 Then

Selection.MoveStart , -1

Selection.Delete

Else

If twoChars <> "el" Then Selection.TypeText "e"

End If

End If

End If

Else

' Remove 'e' and try adding an 'ing'

Selection.Collapse wdCollapseEnd

Selection.MoveStart , -1

If Selection = "e" Then Selection.Delete: GoTo ing

Selection.MoveStart , -2

If Selection = "ies" Then Selection.TypeText "y": GoTo ing

Selection.MoveStart , 1

If Selection = "es" Then Selection.Delete: GoTo ing

If Selection = "ed" Then

tstWd = Left(theWord, Len(theWord) - 2)

spellOK = Application.CheckSpelling(tstWd)

If spellOK = False And Len(theWord) > 4 Then Selection.Delete: GoTo ing

tstWd = tstWd & "ing"

spellOK = Application.CheckSpelling(tstWd)

If spellOK = True Then Selection.Delete: GoTo ing

End If

twoChars = Selection

Selection.MoveStart , 1

If Selection = "s" And twoChars <> "ss" Then Selection.Delete

Selection.Collapse wdCollapseEnd

ing:

Selection.TypeText "ing"

If InStr("an,ur,el,it", twoChars) > 0 Then Selection.MoveLeft , 3

End If

Set rng = Selection.Range.Duplicate

rng.MoveStart , -1

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

myWord = rng.Text

spellOK = Application.CheckSpelling(myWord)

Set suggList = Application.GetSpellingSuggestions(myWord)

If suggList.Count > 0 And Not (spellOK) Then

If doBeep = True Then Beep

rng.Text = suggList.Item(1).Name

End If

rng.Collapse wdCollapseEnd

rng.Select

End Sub

**Sub ParticipleChanger()**

**'** Paul Beverley - Version 06.04.23

' Toggles between past and present participles

If Asc(Selection) = 32 Then Selection.MoveRight , 1

fstChar = Left(Selection, 1)

If UCase(fstChar) = LCase(fstChar) Then Selection.MoveLeft , 1

Selection.Expand wdWord

Do While InStr(ChrW(8217) & ChrW(39) & " ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

' Beware being close to a comment

allCmts = ActiveDocument.Comments.count

cmt = allCmts

If allCmts > 0 Then

Set rng = Selection.Range.Duplicate

rng.Start = 0

cmt1 = rng.Comments.count

rng.End = Selection.Start

cmt0 = rng.Comments.count

If cmt0 = cmt1 Then

cmt = cmt1 + 1

Else

cmt = cmt1

End If

If cmt <= allCmts Then

' Find start and end of comment scope

cmStart = ActiveDocument.Comments(cmt).Scope.Start

cmEnd = ActiveDocument.Comments(cmt).Scope.End

myStart = Selection.Start

myEnd = Selection.End

' Warn if you're too close

doWarn = False

If myStart < cmStart And myEnd > cmStart Then doWarn = True

If myEnd - cmEnd < 3 And myEnd > cmEnd Then doWarn = True

If (myEnd > cmEnd) And (myStart < cmEnd) Then doWarn = True

If doWarn = True Then

Beep

myResponse = MsgBox("Beware: too close to a comment!", vbQuestion + vbcancelonly, \_

"ParticipleChanger")

Exit Sub

End If

End If

End If

wd = Selection

rtChars = Right(wd, 3)

If rtChars = "ing" Then

' Remove 'ing' and try adding an 'ed'

newWord = Left(wd, Len(wd) - 3) & "ed"

Else

' Remove 'ed' and try adding an 'ing'

twoChars = Right(Selection.Text, 2)

lft = Left(wd, Len(wd) - 2)

Select Case twoChars

Case "ed": newWord = lft & "ing"

Case "lt": newWord = lft & "lling"

Case "nt": newWord = lft & "ning"

Case "an": newWord = lft & "inning"

Case "un": newWord = lft & "inning"

Case Else: Beep

End Select

End If

Selection.Text = newWord

spellOK = Application.CheckSpelling(newWord)

If spellOK = False Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

End If

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

End Sub

**Sub VerbChangerNL()**

' Paul Beverley - Version 13.09.19

' Changes Dutch verbs in current sentence

Selection.Expand wdSentence

st = Selection.Start

txt = Selection

zalPosn = InStr(txt, "zal")

zullenPosn = InStr(txt, "zullen")

wordenPosn = InStr(txt, "worden")

zijnPosn = InStr(txt, "zijn")

If zalPosn \* wordenPosn > 0 Then

Selection.Start = st + wordenPosn

Selection.End = st + wordenPosn

Selection.Expand wdWord

Selection.Delete

Selection.Start = st + zalPosn - 1

Selection.End = st + zalPosn + 2

Selection.Text = "wordt"

End If

If zullenPosn \* wordenPosn > 0 Then

Selection.Start = st + wordenPosn

Selection.End = st + wordenPosn

Selection.Expand wdWord

Selection.Delete

Selection.Start = st + zullenPosn - 1

Selection.End = st + zullenPosn + 5

Selection.Text = "worden"

End If

If zalPosn \* zijnPosn > 0 Then

Selection.Start = st + zijnPosn

Selection.End = st + zijnPosn

Selection.Expand wdWord

Selection.Delete

Selection.Start = st + zalPosn - 1

Selection.End = st + zalPosn + 2

Selection.Text = "is"

End If

If zullenPosn \* zijnPosn > 0 Then

Selection.Start = st + zijnPosn

Selection.End = st + zijnPosn

Selection.Expand wdWord

Selection.Delete

Selection.Start = st + zullenPosn - 1

Selection.End = st + zullenPosn + 5

Selection.Text = "zijn"

End If

End Sub

**Sub VerbChangerNLglobal()**

' Paul Beverley - Version 13.09.19

' Changes Dutch verbs through the whole file

For i = 1 To ActiveDocument.Sentences.Count

Set sn = ActiveDocument.Sentences(i)

txt = sn.Text

st = sn.Start

zalPosn = InStr(txt, "zal")

zullenPosn = InStr(txt, "zullen")

wordenPosn = InStr(txt, "worden")

zijnPosn = InStr(txt, "zijn")

Set rng = ActiveDocument.Content

If zalPosn \* wordenPosn > 0 Then

rng.Start = st + wordenPosn

rng.End = st + wordenPosn

rng.Expand wdWord

rng.Delete

rng.Start = st + zalPosn - 1

rng.End = st + zalPosn + 2

rng.Text = "wordt"

End If

If zullenPosn \* wordenPosn > 0 Then

rng.Start = st + wordenPosn

rng.End = st + wordenPosn

rng.Expand wdWord

rng.Delete

rng.Start = st + zullenPosn - 1

rng.End = st + zullenPosn + 5

rng.Text = "worden"

End If

If zalPosn \* zijnPosn > 0 Then

rng.Start = st + zijnPosn

rng.End = st + zijnPosn

rng.Expand wdWord

rng.Delete

rng.Start = st + zalPosn - 1

rng.End = st + zalPosn + 2

rng.Text = "is"

End If

If zullenPosn \* zijnPosn > 0 Then

rng.Start = st + zijnPosn

rng.End = st + zijnPosn

rng.Expand wdWord

rng.Delete

rng.Start = st + zullenPosn - 1

rng.End = st + zullenPosn + 5

rng.Text = "zijn"

End If

Next i

End Sub

**Sub PunctuationToSpace()**

' Paul Beverley - Version 25.03.23

' Change the next punctuation item to a space

trackIt = True

searchChars = "-" & ChrW(8201) & ChrW(160) & ChrW(8211) & ChrW(8212) \_

& ChrW(8722) & Chr(30)

newChar = " "

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToComma()**

' Paul Beverley - Version 25.03.23

' Changes the next punctuation item to a comma

trackIt = True

searchChars = ":;."

newChar = ","

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToDash()**

' Paul Beverley - Version 25.03.23

' Changes next hyphen/em/en dash to an em/en dash

myDash = ChrW(8212) ' em dash

myDash = ChrW(8211) ' en dash

trackIt = True

searchChars = "-~" & ChrW(8211) & ChrW(8212) & ChrW(8722) & Chr(30)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=myDash

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToFullPoint()**

' Paul Beverley - Version 25.03.23

' Changes the next punctuation item to a full point

trackIt = True

searchChars = ":;,"

newChar = "."

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToHardSpace()**

' Paul Beverley - Version 25.03.23

' Changes the next item to a hard space

trackIt = True

searchChars = "-" & ChrW(8201) & " " & ChrW(8211) & ChrW(8212) \_

& ChrW(8722) & Chr(30)

newChar = ChrW(160)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToThinSpace()**

' Paul Beverley - Version 19.11.21

' Changes the next item to a thin space

trackIt = False

makeitColoured = False

myColour = wdYellow

makeNotSubSuper = True

searchChars = " -" & ChrW(160) & ChrW(8201) & Chr(30)

newChar = ChrW(8201)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

If Len(rng) > 1000 Then rng.End = rng.Start + 1000

For Each ch In rng.Characters

If InStr(searchChars, ch.Text) > 0 Then

ch.Select

gotChar = True

Exit For

Else

End If

Next ch

If gotChar = False Then

Beep

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

Selection.TypeText Text:=newChar

Selection.MoveStart , -1

If makeitColoured = True Then Selection.Range.HighlightColorIndex = myColour

If Selection.Font.Name = "Symbol" Then Selection.Font.Name = "Times New Roman"

If makeNotSubSuper = True Then

If Selection.Font.Subscript = True Then Selection.Font.Subscript = False

If Selection.Font.Superscript = True Then Selection.Font.Superscript = False

End If

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToSingleQuote()**

**'** Paul Beverley - Version 06.05.24

' Changes next quote mark to single curly

trackit = True

newChar = "'"

searchChars = Chr(34) & Chr(39) & ChrW(8216) & ChrW(8220) \_

& ChrW(8221) & ChrW(8249) & ChrW(8250) & ChrW(8222) \_

& ChrW(8218) & ChrW(171) & ChrW(187) & ChrW(96) \_

& ChrW(8242) & ChrW(8243)

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Exit Sub

End If

ch = Right(rng, 1)

Selection.Collapse wdCollapseStart

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

optNow = Options.AutoFormatAsYouTypeReplaceQuotes

Options.AutoFormatAsYouTypeReplaceQuotes = True

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ch

.Replacement.Text = "'"

.Execute Replace:=wdReplaceOne

End With

Selection.Collapse wdCollapseEnd

Options.AutoFormatAsYouTypeReplaceQuotes = optNow

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToDoubleQuote()**

' Paul Beverley - Version 26.03.23

' Changes next quote mark to double curly

trackIt = True

newChar = """"

searchChars = Chr(34) & Chr(39) & ChrW(8216) & ChrW(8217) \_

& ChrW(8220) & ChrW(8221) & ChrW(8249) & ChrW(8250) \_

& ChrW(8222) & ChrW(8218) & ChrW(171) & ChrW(187) & ChrW(96) \_

& ChrW(8242) & ChrW(8243)

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Exit Sub

End If

ch = Right(rng, 1)

Selection.Collapse wdCollapseStart

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

optNow = Options.AutoFormatAsYouTypeReplaceQuotes

Options.AutoFormatAsYouTypeReplaceQuotes = True

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ch

.Replacement.Text = """"

.Execute Replace:=wdReplaceOne

End With

Selection.Collapse wdCollapseEnd

Options.AutoFormatAsYouTypeReplaceQuotes = optNow

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToDoubleOpenGuillemet()**

' Paul Beverley - Version 25.03.23

' Changes next quote mark to double open guillemet

trackIt = True

searchChars = Chr(34) & Chr(39) & ChrW(8216) \_

& ChrW(8222) & ChrW(8249) & ChrW(8250) \_

& ChrW(8218) & ChrW(139) & ChrW(96)

newChar = ChrW(171)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToDoubleCloseGuillemet()**

' Paul Beverley - Version 25.03.23

' Changes next quote mark to double close guillemet

trackIt = True

searchChars = Chr(34) & Chr(39) & ChrW(8217) \_

& ChrW(8220) & ChrW(8249) & ChrW(8250) \_

& ChrW(8218) & ChrW(171) & ChrW(96)

newChar = ChrW(187)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToSingleOpenGuillemet()**

' Paul Beverley - Version 25.03.23

' Changes next quote mark to single open guillemet

trackIt = True

searchChars = Chr(34) & Chr(39) & ChrW(8216) \_

& ChrW(8222) & ChrW(8220) & ChrW(8250) \_

& ChrW(8218) & ChrW(171) & ChrW(96)

newChar = ChrW(8249)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToSingleCloseGuillemet()**

' Paul Beverley - Version 25.03.23

' Changes next quote mark to single close guillemet

trackIt = True

searchChars = Chr(34) & Chr(39) & ChrW(8217) \_

& ChrW(8221) & ChrW(8249) & ChrW(8250) \_

& ChrW(8218) & ChrW(139) & ChrW(96)

newChar = ChrW(8250)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToPrime()**

' Paul Beverley - Version 25.03.23

' Change next quote mark to single prime

trackIt = True

searchChars = Chr(34) & Chr(39) & ChrW(8216) & ChrW(8217) \_

& ChrW(8220) & ChrW(8221) & ChrW(8249) & ChrW(8250) \_

& ChrW(8222) & ChrW(8218) & ChrW(171) & ChrW(187) & ChrW(96) \_

& ChrW(8242) & ChrW(8243)

newChar = ChrW(8242)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToDoublePrime()**

' Paul Beverley - Version 25.03.23

' Change next quote mark to double prime

trackIt = True

searchChars = Chr(34) & Chr(39) & ChrW(8216) & ChrW(8217) \_

& ChrW(8220) & ChrW(8221) & ChrW(8249) & ChrW(8250) \_

& ChrW(8222) & ChrW(8218) & ChrW(171) & ChrW(187) & ChrW(96) \_

& ChrW(8242) & ChrW(8243)

newChar = ChrW(8243)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToMultiplySign()**

' Paul Beverley - Version 19.11.21

' Changes next x or X to multiplication sign

trackIt = True

searchChars = "xX"

newChar = ChrW(215)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

If Len(rng) > 1000 Then rng.End = rng.Start + 1000

For Each ch In rng.Characters

If InStr(searchChars, ch.Text) > 0 Then

ch.Select

gotChar = True

Exit For

Else

End If

Next ch

If gotChar = False Then

Beep

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

Selection.TypeText Text:=newChar

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToSingleQuoteDE()**

' Paul Beverley - Version 11.12.19

' Changes next quote mark to German single curly

trackit = True

openQuote = ChrW(8218)

closeQuote = ChrW(8216)

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

searchChars = Chr(34) & Chr(39) & ChrW(8216) & ChrW(8217) \_

& ChrW(8220) & ChrW(8221) & ChrW(8249) & ChrW(8250) \_

& ChrW(8222) & ChrW(8218) & ChrW(171) & ChrW(187) \_

& ChrW(96) & ChrW(8242) & ChrW(8243)

unText = " (,!)" & vbCr

Selection.Collapse wdCollapseEnd

startHere = Selection.Start

Set rng = ActiveDocument.Content

theEnd = rng.End

Do While InStr(searchChars, Selection) = 0

Selection.MoveRight , 1

If Selection.Start > startHere + 1000 Or \_

Selection.Start > theEnd - 2 Then

Beep

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

Loop

Selection.MoveEnd wdCharacter, 1

thisChar = Selection

myStart = Selection.Start

Set rng = ActiveDocument.range(myStart - 1, myStart)

preChar = rng.Text

Set rng = ActiveDocument.range(myStart + 1, myStart + 2)

postChar = rng.Text

myQuote = ""

If LCase(preChar) <> UCase(preChar) Then myQuote = closeQuote

If LCase(postChar) <> UCase(postChar) Then myQuote = openQuote

If myQuote = "" Then

If InStr(unText, preChar) > 0 Then myQuote = openQuote

If InStr(unText, postChar) > 0 Then myQuote = closeQuote

End If

If myQuote <> "" Then

Selection.TypeText Text:=myQuote

Else

Beep

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub CurlyQuotesToggle()**

' Paul Beverley - Version 18.02.20

' Switches on auto curly quotes on and off

If Options.AutoFormatAsYouTypeReplaceQuotes Then

Options.AutoFormatAsYouTypeReplaceQuotes = False

Options.AutoFormatReplaceQuotes = False

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Else

Options.AutoFormatAsYouTypeReplaceQuotes = True

Options.AutoFormatReplaceQuotes = True

Beep

End If

End Sub

**Sub PunctuationToDoubleQuoteDE()**

' Paul Beverley - Version 11.12.19

' Changes next quote mark to German double curly

trackit = True

openQuote = ChrW(8222)

closeQuote = ChrW(8220)

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

searchChars = Chr(34) & Chr(39) & ChrW(8216) & ChrW(8217) \_

& ChrW(8220) & ChrW(8221) & ChrW(8249) & ChrW(8250) \_

& ChrW(8222) & ChrW(8218) & ChrW(171) & ChrW(187) \_

& ChrW(96) & ChrW(8242) & ChrW(8243)

unText = " (,!)" & vbCr

Selection.Collapse wdCollapseEnd

startHere = Selection.Start

Set rng = ActiveDocument.Content

theEnd = rng.End

Do While InStr(searchChars, Selection) = 0

Selection.MoveRight , 1

If Selection.Start > startHere + 1000 Or \_

Selection.Start > theEnd - 2 Then

Beep

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

Loop

Selection.MoveEnd wdCharacter, 1

thisChar = Selection

myStart = Selection.Start

Set rng = ActiveDocument.range(myStart - 1, myStart)

preChar = rng.Text

Set rng = ActiveDocument.range(myStart + 1, myStart + 2)

postChar = rng.Text

myQuote = ""

If LCase(preChar) <> UCase(preChar) Then myQuote = closeQuote

If LCase(postChar) <> UCase(postChar) Then myQuote = openQuote

If myQuote = "" Then

If InStr(unText, preChar) > 0 Then myQuote = openQuote

If InStr(unText, postChar) > 0 Then myQuote = closeQuote

End If

If myQuote <> "" Then

Selection.TypeText Text:=myQuote

Else

Beep

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToSingleQuoteFR()**

' Paul Beverley - Version 11.12.19

' Changes next quote mark to French single curly

trackit = True

openQuote = ChrW(8249)

closeQuote = ChrW(8250)

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

searchChars = Chr(34) & Chr(39) & ChrW(8216) & ChrW(8217) \_

& ChrW(8220) & ChrW(8221) & ChrW(8249) & ChrW(8250) \_

& ChrW(8222) & ChrW(8218) & ChrW(171) & ChrW(187) \_

& ChrW(96) & ChrW(8242) & ChrW(8243)

unText = " (,!)" & vbCr

Selection.Collapse wdCollapseEnd

startHere = Selection.Start

Set rng = ActiveDocument.Content

theEnd = rng.End

Do While InStr(searchChars, Selection) = 0

Selection.MoveRight , 1

If Selection.Start > startHere + 1000 Or \_

Selection.Start > theEnd - 2 Then

Beep

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

Loop

Selection.MoveEnd wdCharacter, 1

thisChar = Selection

myStart = Selection.Start

Set rng = ActiveDocument.range(myStart - 1, myStart)

preChar = rng.Text

Set rng = ActiveDocument.range(myStart + 1, myStart + 2)

postChar = rng.Text

myQuote = ""

If LCase(preChar) <> UCase(preChar) Then myQuote = closeQuote

If LCase(postChar) <> UCase(postChar) Then myQuote = openQuote

If myQuote = "" Then

If InStr(unText, preChar) > 0 Then myQuote = openQuote

If InStr(unText, postChar) > 0 Then myQuote = closeQuote

End If

If myQuote <> "" Then

Selection.TypeText Text:=myQuote

Else

Beep

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToDoubleQuoteFR()**

' Paul Beverley - Version 11.12.19

' Changes next quote mark to French double curly

trackit = True

openQuote = ChrW(171)

closeQuote = ChrW(187)

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

searchChars = Chr(34) & Chr(39) & ChrW(8216) & ChrW(8217) \_

& ChrW(8220) & ChrW(8221) & ChrW(8249) & ChrW(8250) \_

& ChrW(8222) & ChrW(8218) & ChrW(171) & ChrW(187) \_

& ChrW(96) & ChrW(8242) & ChrW(8243)

unText = " (,!)" & vbCr

Selection.Collapse wdCollapseEnd

startHere = Selection.Start

Set rng = ActiveDocument.Content

theEnd = rng.End

Do While InStr(searchChars, Selection) = 0

Selection.MoveRight , 1

If Selection.Start > startHere + 1000 Or \_

Selection.Start > theEnd - 2 Then

Beep

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

Loop

Selection.MoveEnd wdCharacter, 1

thisChar = Selection

myStart = Selection.Start

Set rng = ActiveDocument.range(myStart - 1, myStart)

preChar = rng.Text

Set rng = ActiveDocument.range(myStart + 1, myStart + 2)

postChar = rng.Text

myQuote = ""

If LCase(preChar) <> UCase(preChar) Then myQuote = closeQuote

If LCase(postChar) <> UCase(postChar) Then myQuote = openQuote

If myQuote = "" Then

If InStr(unText, preChar) > 0 Then myQuote = openQuote

If InStr(unText, postChar) > 0 Then myQuote = closeQuote

End If

If myQuote <> "" Then

Selection.TypeText Text:=myQuote

Else

Beep

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub QuotesToGuillemets()**

' Paul Beverley - Version 08.10.22

' Changes all double and single quotes (inc. German) into guillemets

doHighlight = False

Options.AutoFormatAsYouTypeReplaceQuotes = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & ChrW(8222) & ChrW(8221) & ChrW(8220) & "]"

.Wrap = wdFindContinue

.Replacement.Text = """"

.Replacement.Highlight = doHighlight

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

.Text = "[" & ChrW(8216) & ChrW(8217) & ChrW(8218) & "]"

.Text = "'"

.Replacement.Text = "'"

.Execute Replace:=wdReplaceAll

Options.AutoFormatAsYouTypeReplaceQuotes = True

.Text = """"

.Replacement.Text = """"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = "'"

.Wrap = wdFindContinue

.Replacement.Text = "'"

.Execute Replace:=wdReplaceAll

Options.AutoFormatAsYouTypeReplaceQuotes = False

.Text = ChrW(8220)

.Replacement.Text = ChrW(171) & ChrW(160)

.Execute Replace:=wdReplaceAll

DoEvents

.Text = ChrW(8221)

.Replacement.Text = ChrW(160) & ChrW(187)

.Execute Replace:=wdReplaceAll

.Text = ChrW(8216)

.Replacement.Text = ChrW(8249) & ChrW(160)

.Execute Replace:=wdReplaceAll

DoEvents

.MatchWildcards = True

.Text = "([a-zA-Z])" & ChrW(8217) & "([!a-zA-Z])"

.Replacement.Text = "\1" & ChrW(160) & ChrW(8250) & "\2"

.Execute Replace:=wdReplaceAll

End With

Options.AutoFormatAsYouTypeReplaceQuotes = True

End Sub

**Sub QuotesAddDouble()**

**'** Paul Beverley - Version 06.07.24

' Adds quotes round a word or phrase

useUSpunctuation = True

addHighlight = False

myColour = wdBrightGreen

' doubles

myOpen = ChrW(8220)

myClose = ChrW(8221)

' Or singles

' myOpen = ChrW(8216)

' myClose = ChrW(8217)

If Selection = "." Then Selection.MoveLeft , 1

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.MoveStart , -1

If LCase(rng) = UCase(rng) Then Selection.MoveRight , 1

Set rng = Selection.Range.Duplicate

myEnd = rng.End

rng.Collapse wdCollapseStart

rng.Expand wdWord

rng.Collapse wdCollapseStart

rng.InsertBefore Text:=myOpen

If addHighlight = True Then

rng.HighlightColorIndex = myColour

End If

rng.start = myEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

If useUSpunctuation = True Then

rng.MoveEnd , 1

If InStr(",.", Right(rng.Text, 1)) = 0 Then rng.MoveEnd , -1

End If

rng.Collapse wdCollapseEnd

rng.InsertAfter Text:=myClose

If addHighlight = True Then

rng.HighlightColorIndex = myColour

End If

rng.Collapse wdCollapseEnd

rng.Select

End Sub

**Sub QuotesAddSingle()**

**'** Paul Beverley - Version 06.07.24

' Adds single quotes round a word or phrase

useUSpunctuation = False

addHighlight = False

myColour = wdBrightGreen

' singles

myOpen = ChrW(8216)

myClose = ChrW(8217)

If Selection = "." Then Selection.MoveLeft , 1

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.MoveStart , -1

If LCase(rng) = UCase(rng) Then Selection.MoveRight , 1

Set rng = Selection.Range.Duplicate

myEnd = rng.End

rng.Collapse wdCollapseStart

rng.Expand wdWord

rng.Collapse wdCollapseStart

rng.InsertBefore Text:=myOpen

If addHighlight = True Then

rng.HighlightColorIndex = myColour

End If

rng.start = myEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

If useUSpunctuation = True Then

rng.MoveEnd , 1

If InStr(",.", Right(rng.Text, 1)) = 0 Then rng.MoveEnd , -1

End If

rng.Collapse wdCollapseEnd

rng.InsertAfter Text:=myClose

If addHighlight = True Then

rng.HighlightColorIndex = myColour

End If

rng.Collapse wdCollapseEnd

rng.Select

End Sub

**Sub ParenthesesAdd()**

' Paul Beverley - Version 06.06.22

' Puts parentheses round the current word or phrase

myEnd = Selection.End

If Selection = "." Then Selection.MoveLeft , 1

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

Selection.TypeText "("

Selection.Start = myEnd

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Selection.TypeText ")"

End Sub

**Sub AddTextRoundText()**

' Paul Beverley - Version 14.01.21

' Adds text at either end of a word or phrase (can delete punctuation)

deleteExistingMarks = True

doTrack = False

' singles

myOpen = ChrW(8216)

myClose = ChrW(8217)

' doubles

myOpen = ChrW(8220)

myClose = ChrW(8221)

' parentheses

myOpen = "("

myClose = ")"

' tags

myOpen = "<it>"

myClose = "</it>"

deleteThese = "()[].,"

' And the various quotation marks

deleteThese = deleteThese & """'" & ChrW(8220) & ChrW(8221) & ChrW(8216) & ChrW(8217)

myTrack = ActiveDocument.TrackRevisions

If doTrack = False Then ActiveDocument.TrackRevisions = False

myStart = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

myEnd = Selection.End

Selection.End = myStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

myStart = Selection.End

myLength = myEnd - myStart

' Select first word

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

Selection.MoveStart , -1

If deleteExistingMarks = True And InStr(deleteThese, Selection) > 0 Then

Selection.Delete

Else

Selection.Collapse wdCollapseEnd

End If

Selection.TypeText myOpen

Selection.Start = Selection.Start + myLength

Selection.End = Selection.Start + 1

If deleteExistingMarks = True And InStr(deleteThese, Selection) > 0 \_

Then Selection.Delete

Selection.Collapse wdCollapseStart

Selection.TypeText myClose

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub ParenthesesEtcPairDelete()**

' Paul Beverley - Version 14.03.23

' Removes the following pair of parentheses or quotes, etc.

Set rng = Selection.Range.Duplicate

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[\)\]\}\>," & ChrW(8217) & ChrW(8221) & "]"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute

cl = rng.Text

.Execute Replace:=wdReplaceOne

DoEvents

End With

Select Case cl

Case "]": op = "["

Case ")": op = "("

Case "}": op = "{"

Case ">": op = "<"

Case ",": op = ","

Case ChrW(8217): op = ChrW(8216)

Case ChrW(8221): op = ChrW(8220)

End Select

Set rng = Selection.Range.Duplicate

rng.Start = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = op

.Wrap = wdFindContinue

.Forward = False

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

.Forward = True

DoEvents

End With

If rng.Find.Found = False Then

Beep

MsgBox "Couldn't find: " & op

Exit Sub

End If

End Sub

**Sub DoubleQuotesToSingle()**

' Paul Beverley - Version 31.08.16

' Changes the curly quotes on a given phrase from double to single

thisPhrase = InputBox("Phrase to change?", "DoubleQuotesToSingle")

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

' Go and find the first occurrence

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Chr(34) & thisPhrase & Chr(34)

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the end of the found item is

myStart = rng.Start

myEnd = rng.End

rng.Collapse wdCollapseEnd

rng.MoveStart , -1

' Fiddle factor for Gordon Hooper's 'problem'

If rng.Text <> Chr(34) Then

Selection.MoveStart , 1

Selection.MoveEnd , 1

End If

rng.Delete

rng.InsertAfter Text:=ChrW(8217)

rng.Start = myStart

rng.End = myStart + 1

rng.Delete

rng.InsertAfter Text:=ChrW(8216)

rng.Start = myStart + 2

rng.End = myStart + 2

rng.Find.Execute

Loop

MsgBox "Changed: " & myCount

End Sub

**Sub CommaAdd()**

**'** Paul Beverley - Version 13.05.24

' Adds a comma after the current word

spanishPunct = False

myTrack = ActiveDocument.TrackRevisions

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Set tst = rng.Duplicate

tst.MoveEnd , 1

' That's got the space

Set tst2 = tst.Duplicate

tst2.Collapse wdCollapseEnd

tst2.MoveEnd , 1

' That's got the next character

' Check if italic is switching off word to word

If tst2.Font.Italic = False Then

tst.Font.Italic = False

removeItalic = True

Else

removeItalic = False

End If

' Check if bold is switching off word to word

If tst2.Font.Bold = False Then

tst.Font.Bold = False

removeBold = True

Else

removeBold = False

End If

rng.Collapse wdCollapseEnd

rng.Select

Selection.InsertAfter Text:=","

ActiveDocument.TrackRevisions = False

If spanishPunct = False And removeBold = True Then \_

Selection.Font.Bold = False

If spanishPunct = False And removeItalic = True Then \_

Selection.Font.Italic = False

ActiveDocument.TrackRevisions = myTrack

Selection.MoveRight , 2

End Sub

**Sub CommaAddPrevious()**

' Paul Beverley - Version 19.09.18

' Adds a comma before the current word

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

Selection.TypeText Text:=", "

Selection.MoveLeft , 2

Selection.MoveStart , -1

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.Delete

ActiveDocument.TrackRevisions = myTrack

Selection.MoveRight , 1

End Sub

**Sub CommaAddUSUK()**

**'** Paul Beverley - Version 28.11.10

' Add a comma

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

Selection.End = Selection.Start

With Selection.Find

.ClearFormatting

.Text = " "

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Selection.End = Selection.Start

If Selection.LanguageID = wdEnglishUS Then

Selection.MoveEnd , -1

If Selection <> ChrW(8217) And Selection <> ChrW(8221) Then

Selection.MoveStart , 1

End If

End If

Selection.TypeText Text:=","

Selection.MoveRight Unit:=wdCharacter, Count:=1

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

End Sub

**Sub TypeThinSpace()**

' Paul Beverley - Version 25.10.13

' Types and highlights a thin space

trackIt = False

makeitColoured = False

myColour = wdYellow

makeNotSubSuper = True

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Selection.TypeText ChrW(8201)

Selection.MoveStart wdCharacter, -1

If makeitColoured = True Then Selection.Range.HighlightColorIndex = myColour

If Selection.Font.Name = "Symbol" Then Selection.Font.Name = "Times New Roman"

If makeNotSubSuper = True Then

If Selection.Font.Subscript = True Then Selection.Font.Subscript = False

If Selection.Font.Superscript = True Then Selection.Font.Superscript = False

End If

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToHyphen()**

' Paul Beverley - Version 25.03.23

' Changes the word break punctuation to a hyphen

trackIt = True

myChar = "-"

searchChars = " " & ChrW(160) & ChrW(8211) & ChrW(8212) & ChrW(8722) & Chr(30)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToEnDash()**

' Paul Beverley - Version 19.11.21

' Changes next hyphen/em/en dash to an en dash

myDash = ChrW(8211) ' en dash

trackit = True

searchChars = "-~" & ChrW(8211) & ChrW(8212) & ChrW(8722) & Chr(30)

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

If Len(rng) > 1000 Then rng.End = rng.Start + 1000

For Each ch In rng.Characters

If InStr(searchChars, ch.Text) > 0 Then

ch.Select

gotChar = True

Exit For

Else

End If

Next ch

If gotChar = False Then

Beep

Else

Selection.TypeText Text:=myDash

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToEmDash()**

' Paul Beverley - Version 19.11.21

' Changes next hyphen/em/en dash to an em dash

myDash = ChrW(8212) ' em dash

trackit = True

searchChars = "-~" & ChrW(8211) & ChrW(8212) & ChrW(8722) & Chr(30)

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

If Len(rng) > 1000 Then rng.End = rng.Start + 1000

For Each ch In rng.Characters

If InStr(searchChars, ch.Text) > 0 Then

ch.Select

gotChar = True

Exit For

Else

End If

Next ch

If gotChar = False Then

Beep

Else

Selection.TypeText Text:=myDash

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToNonBreakingEmDash()**

' Paul Beverley - Version 14.06.22

' Changes next hyphen/em/en dash to a non-breaking em dash

myDash = ChrW(8212) ' em dash

trackit = True

searchChars = "-~" & ChrW(8211) & ChrW(8212) & ChrW(8722) & Chr(30)

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

If Len(rng) > 1000 Then rng.End = rng.Start + 1000

For Each ch In rng.Characters

If InStr(searchChars, ch.Text) > 0 Then

ch.Select

gotChar = True

Exit For

Else

End If

Next ch

If gotChar = False Then

Beep

Else

Selection.TypeText Text:=myDash & ChrW(8205)

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToNonBreakingEnDash()**

' Paul Beverley - Version 25.03.23

' Changes next hyphen/em/en dash to a non-breaking em/en dash

myDash = ChrW(8211) ' en dash

' myDash = ChrW(8212) ' em dash

trackIt = True

searchChars = "-~" & ChrW(8211) & ChrW(8212) & ChrW(8722) & Chr(30)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=myDash

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToMinus()**

**'** Paul Beverley - Version 13.04.24

' Finds punctuation and changes to minus sign

trackit = False

newChar = ChrW(8722)

searchChars = "-" & ChrW(8211) & ChrW(8212) & Chr(30)

myTrack = ActiveDocument.TrackRevisions

If trackit = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Select

Selection.InsertAfter Text:=newChar

Selection.Collapse wdCollapseEnd

rng.Delete

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToNonBreakingThinSpace()**

' Paul Beverley - Version 25.03.23

' Changes next space etc to a non-breaking thin space

newChar = ChrW(8205)

trackIt = True

searchChars = " -" & ChrW(160) & ChrW(8201) & Chr(30)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationToMultiplySign()**

' Paul Beverley - Version 25.03.23

' Changes next x or X to multiplication sign

searchChars = "xX"

newChar = ChrW(215)

trackIt = True

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationOff()**

**'** Paul Beverley - Version 30.06.23

' Deletes the next punctuation item

myChars = ""

myChars = myChars + ";:.,\!\?"

' Curly quotes

myChars = myChars + ChrW(8216) & ChrW(8217) & ChrW(8220) & ChrW(8221)

' Straight quotes

myChars = myChars + "'" & Chr(34)

' Ellipsis

myChars = myChars + ChrW(8230)

' Brackets

myChars = myChars + "\(\)\[\]\{\}"

oldFind = Selection.Find.Text

Selection.End = Selection.Start

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & myChars & "]"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

With Selection.Find

.Text = oldFind

.MatchWildcards = False

End With

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

Selection.Delete

rng.Select

End Sub

**Sub PunctuationToNothing()**

' Paul Beverley - Version 25.03.23

' Changes next punctuation mark to nothing, i.e. deletes it

searchChars = ",.?!:;"

newChar = ""

trackIt = True

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Else

rng.Delete

rng.Select

Selection.TypeText Text:=newChar

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationSwitcher()**

' Paul Beverley - Version 04.10.12

' Changes order of next two punctuation marks

myChars = ""

myChars = myChars + ";:.,\!\?"

' Curly quotes

myChars = myChars + ChrW(8216) & ChrW(8217) & ChrW(8220) & ChrW(8221)

' Straight quotes

myChars = myChars + "'" & Chr(34)

' Brackets

myChars = myChars + "\(\)\[\]\{\}"

oldFind = Selection.Find.Text

Selection.End = Selection.Start

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & myChars & "]{2}"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myPair = Selection

pair = Right(myPair, 1) & Left(myPair, 1)

Selection.TypeText Text:=pair

With Selection.Find

.Text = oldFind

.MatchWildcards = False

End With

End Sub

**Sub WordPairPunctuate()**

' Paul Beverley - Version 31.08.22

' Makes word pair hyphenated or single word

trackIt = False

reverseOrder = True

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "pairWord" Then varExists = True: Exit For

Next v

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

If varExists = False Then ActiveDocument.Variables.Add \_

"pairWord", "Dummy%Dummy"

myWord = ActiveDocument.Variables("pairWord")

myFirst = ""

mySecond = ""

myBreak = "-"

For i = 1 To Len(myWord)

char = Mid(myWord, i, 1)

If LCase(char) = UCase(char) Then

myFirst = Left(myWord, i - 1)

myBreak = char

mySecond = Mid(myWord, i + 1)

End If

Next i

Selection.Expand wdWord

myStart = Selection.Start

firstWd = Trim(Selection)

endFirstWd = myStart + Len(firstWd)

Selection.Collapse wdCollapseEnd

Selection.Expand wdWord

secondWd = Trim(Selection)

isHyph = (secondWd = "-")

nowBreak = " "

If isHyph = True Then

Selection.Collapse wdCollapseEnd

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' " & vbCr, Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

secondWd = Selection

nowBreak = "-"

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

myEnd = Selection.End

If firstWd = myFirst & mySecond Then

myEnd = endFirstWd

myBreak = "|"

End If

If myBreak = "%" Or myFirst = "Dummy" Or \_

(myFirst <> firstWd And myEnd <> endFirstWd) Then

myFirst = firstWd

mySecond = secondWd

myBreak = nowBreak

End If

If reverseOrder = False Then

Select Case myBreak

Case "|": newWord = myFirst & "-" & mySecond

Case "-": newWord = myFirst & " " & mySecond

Case " ": newWord = myFirst & "|" & mySecond

End Select

Else

Select Case myBreak

Case "|": newWord = myFirst & " " & mySecond

Case " ": newWord = myFirst & "-" & mySecond

Case "-": newWord = myFirst & "|" & mySecond

End Select

End If

ActiveDocument.Variables("pairWord") = newWord

Selection.Start = myStart

Selection.End = myEnd

Selection.TypeText Replace(newWord, "|", "")

Selection.Start = myStart

Selection.Collapse wdCollapseStart

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub JoinTwoWords()**

' Paul Beverley - Version 29.05.21

' Joins two words

joinToPrevious = False

forceLowercase = False

trackIt = True

searchChars = "- " & ChrW(8211) & ChrW(8212) & ChrW(8201) & Chr(30)

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

If joinToPrevious = True Then

Do While InStr(searchChars, Left(Selection.Text, 1)) = 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Selection.End = Selection.End + 1

Else

Do While InStr(searchChars, Right(Selection.Text, 1)) = 0

Selection.MoveEnd , 1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Selection.End = Selection.End - 1

End If

Selection.Delete

If forceLowercase = True Then

Selection.MoveEnd , 1

Selection.Text = LCase(Selection)

Selection.Collapse wdCollapseEnd

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub ThatWhich()**

**'** Paul Beverley - Version 06.08.22

' Changes that to which and vice versa

Selection.MoveLeft Unit:=wdWord, count:=1

Selection.MoveRight Unit:=wdWord, count:=1, Extend:=wdExtend

thisWord = Selection

If thisWord = "which " Then

Selection.TypeText Text:="that "

thisWord = ""

Selection.MoveLeft Unit:=wdWord, count:=2

If Selection = "," Then Selection.Delete

End If

If thisWord = "which" Then

Selection.TypeText Text:="that"

thisWord = ""

End If

If thisWord = "that " Then

Selection.Start = Selection.Start - 1

Selection.Delete

Selection.TypeText Text:=", which "

thisWord = ""

End If

If thisWord <> "" Then

Selection.Start = Selection.End

Selection.TypeText Text:="that "

End If

End Sub

**Sub MultiSwitch()**

**'** Paul Beverley - Version 19.12.24

' Scripted word/phrase switching

maxWords = 5

listNameIncludes = "switch"

commentNearCheck = True

' Set min number of chars for an abbreviation

minChars = 3

includeApostrophe = True

removeHighlight = True

linksOFF = True

addMarkerInList = True

notTheseChars = ",!?.)>] "

' If you want to load the SwitchList file automatically...

' then...

' On a Mac, you will need something like this:

myList = "/Users/Paul/My Documents/Macro stuff/zzSwitchList.docx"

' On Windows, you will need something like this:

myList = "C:\Documents and Settings\Paul\My Documents\zzSwitchList.docx"

CR = vbCr

CR2 = CR & CR

Dim myStr(20) As String

Dim endStr(20) As Long

On Error GoTo ReportIt

Set theDoc = ActiveDocument

' Just in case they are using "smart" cut/paste option

mySmartOpt = Options.SmartCutPaste

Options.SmartCutPaste = False

nowSearch = Selection.Find.Text

' Read the context, noting if the word is selected

wasSelected = (Selection.End > Selection.start)

If wasSelected = True Then maxWords = 1

myTest = Selection.Text

If Asc(myTest) = 13 Then Selection.MoveLeft , 1

If InStr(notTheseChars, myTest) > 0 Then Selection.MoveLeft , 1

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

okChars = ".,;:" & Chr(11) & vbCr & ChrW(8211) & ChrW(8212) & ChrW(8221)

If InStr(okChars, Left(Selection, 1)) > 0 Then

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.Expand wdWord

End If

isAbbrev = Not (Len(Selection) > minChars)

startNow = Selection.start

Set startRng = Selection.Range.Duplicate

iMax = maxWords

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.MoveEnd Unit:=wdWord, count:=maxWords

numTCs = rng.Revisions.count

If numTCs > 0 Then

rng.Revisions.AcceptAll

rng.MoveEnd Unit:=wdWord, count:=maxWords

End If

wdsLeft = rng.Words.count

If maxWords > wdsLeft Then maxWords = wdsLeft

If Asc(rng.Words(wdsLeft)) = 13 Then maxWords = maxWords - 1

For i = 1 To maxWords

endStr(i) = rng.Words(i).End

If rng.Words(i) = vbCr Then

maxWords = i - 1

Exit For

End If

If Right(rng.Words(i), 1) = " " Then endStr(i) = endStr(i) - 1

myStr(i) = Trim(Left(rng, endStr(i) - rng.start))

DoEvents

Next i

If numTCs > 0 Then WordBasic.EditUndo

Selection.Collapse wdCollapseStart

iMax = maxWords

gottaList = False

For Each myDoc In Application.Documents

thisName = myDoc.Name

If InStr(LCase(thisName), LCase(listNameIncludes)) > 0 Then

Set theList = Documents(thisName)

gottaList = True

Exit For

End If

Next myDoc

' Find the zzSwitchList file

If gottaList = False Then

Documents.Open fileName:=myList

Set theList = ActiveDocument

theDoc.Activate

End If

Set rng = theList.Content

' Check if list has LFs

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = ChrW(11)

.Replacement.Text = ""

.Execute

DoEvents

If .Found = True Then

Beep

rng.start = rng.start - 1

rng.Collapse wdCollapseStart

rng.Expand wdWord

rng.HighlightColorIndex = wdYellow

rng.Collapse wdCollapseEnd

rng.Select

MsgBox "The switch list must use paragraphs, not line breaks!"

ActiveWindow.ActivePane.View.ShowAll = True

rng.Find.Text = nowSearch

Exit Sub

End If

End With

' Just in case the switch list has any URLs

If linksOFF = True Then

rng.Fields.Unlink

End If

lastTwo = Right(rng, 2)

If lastTwo <> CR2 Then rng.InsertAfter Text:=CR2

firstTwo = Left(rng, 2)

If firstTwo <> CR2 Then rng.InsertBefore Text:=CR2

rng.start = 0

rng.End = theList.Content.End

' Try to locate the selected text in the switch list

allWords = rng.Text

lfPos = InStr(allWords, ChrW(11))

If lfPos > 0 Then

rng.start = lfPos - 2

rng.End = lfPos + 1

rng.Select

Beep

Application.Run "ShowFormatting"

MsgBox "The switch list must use paragraphs, not line breaks!"

Options.SmartCutPaste = mySmartOpt

Selection.Find.Text = nowSearch

Exit Sub

End If

For i = iMax To 1 Step -1

myPos = InStr(allWords, CR2 & Replace(myStr(i), CR, "") & CR)

If myPos > 0 Then

allWords = Mid(allWords, myPos + 1)

gottaMatch = True

numWords = i

'Display the options

numAlts = 1

endPos = InStr(allWords, CR2)

allWords = Left(allWords, endPos + 2)

altText = Split(allWords, Chr(13))

j = 0

Do

j = j + 1

Loop Until Len(altText(j)) = 0 Or j = 20

numAlts = j - 2

' We now have j = number of options

' altText contains all numAlts texts

Exit For

Else

gottaMatch = False

End If

Next i

' If it's not found at all, give up

If gottaMatch = False Then

myWd = myStr(1)

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p^p" & myWd

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = True

.MatchWildcards = False

.Execute

.MatchCase = False

End With

If Len(myWd) > 1 Then myWd = Left(myWd, Len(myWd) - 1)

Loop Until rng.Find.Found = True Or Len(myWd) = 1

rng.Collapse wdCollapseStart

rng.Select

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Options.SmartCutPaste = mySmartOpt

Selection.Find.Text = nowSearch

Debug.Print "|" & myStr(1) & "|"

myPrompt = "Can't find """ & myStr(1) & """"

myResponse = MsgBox(myPrompt, vbOKOnly, "MultiSwitch")

theDoc.Activate

Exit Sub

End If

'Offer j optional texts to the user

If numAlts > 1 And wasSelected = False Then

For i = 1 To numAlts

myPrompt = myPrompt & i & ": " & altText(i + 1) & CR

Next

Beep

myResponse = InputBox(myPrompt, "MultiSwitch")

myChoice = Val(myResponse)

If myChoice > numAlts Or myChoice = 0 Then

Options.SmartCutPaste = mySmartOpt

Selection.Find.Text = nowSearch

Exit Sub

End If

Else

myChoice = 1

End If

rng.start = myPos + 2

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

For i = 1 To myChoice

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

Next i

' Copy chosen line (check first for format change symbol)

useFormat = False

rng.MoveEnd , -1

myNewItem = rng

Dim gotFormat As Boolean

gotFormat = rng.Font.Bold Or rng.Font.Italic Or rng.Font.SmallCaps \_

Or rng.Font.Underline Or rng.Font.Superscript Or rng.Font.Subscript \_

Or rng.InlineShapes.count > 0

If rng.Text = "" Then Beep: Exit Sub

If AscW(rng) = 172 Or gotFormat Then

useFormat = True

If AscW(rng) = 172 Then rng.MoveStart , 1

rng.Copy

End If

noTrack = (rng.Font.StrikeThrough)

startRng.Select

Selection.End = endStr(numWords)

' Check if too near to a comment

If Selection.Information(wdInCommentPane) = 0 And \_

Selection.Information(wdInFootnote) = 0 And \_

Selection.Information(wdInEndnote) = 0 Then

numCmts = ActiveDocument.Comments.count

If numCmts > 0 And commentNearCheck = True Then

' Find scope of (range covered by) the next comment

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.start = 0

cmtNumber = rng.Comments.count + 1

If cmtNumber <= numCmts Then

Set nextCmtRange = ActiveDocument.Comments(cmtNumber).Scope

' Does the scope of that comment overlap the current word?

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

If rng.End > nextCmtRange.start Then

Beep

rng.Select

myResponse = MsgBox("Selection contains a comment. Word might crash!" \_

& CR2 & "Please use a manual change.", \_

vbOKOnly, "MultiSwitch")

Options.SmartCutPaste = mySmartOpt

Selection.Find.Text = nowSearch

Exit Sub

End If

End If

End If

End If

With ActiveWindow.View.RevisionsFilter

myMarkup = .Markup

myView = .View

End With

myTrack = ActiveDocument.TrackRevisions

If noTrack Then ActiveDocument.TrackRevisions = False

If useFormat = True Then

Selection.Delete

Selection.Paste

Else

Selection.Text = myNewItem

If removeHighlight = True Then \_

Selection.Range.HighlightColorIndex = wdNoHighlight

End If

Selection.start = startNow

textEnd = Selection.End

If InStr(Selection, "^p") > 0 Then

numCRs = 0

oldFind = Selection.Find.Text

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^^p"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute

End With

Do While Selection.Find.Found = True And Selection.start < textEnd

Selection.TypeText Text:=vbCr

Selection.Find.Execute

numCRs = numCRs + 1

Loop

Selection.start = startNow

Selection.End = textEnd - numCRs

Selection.Find.Text = oldFind

End If

If InStr(Selection, "^t") > 0 Then

numTabs = 0

oldFind = Selection.Find.Text

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^^t"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute

End With

Do While Selection.Find.Found = True And Selection.start < textEnd

Selection.TypeText Text:=vbTab

Selection.Find.Execute

numTabs = numTabs + 1

Loop

Selection.start = startNow

Selection.End = textEnd - numCRs - numTabs

Selection.Find.Text = oldFind

End If

If Asc(Selection) = Asc("!") Then

myLen = Len(Selection)

Selection.End = Selection.start + 1

Selection.start = Selection.start - 1

Selection.Delete

Selection.MoveEnd , myLen - 1

End If

tildePos = InStr(Selection, "~")

Selection.Font.StrikeThrough = False

If tildePos = 0 Then

If isAbbrev = True Then

Selection.Collapse wdCollapseEnd

Else

Selection.Collapse wdCollapseStart

End If

Else

Do While InStr(Selection, "~") > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Selection.MoveEnd 1

Selection.Delete

End If

If addOriginal = True Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=" (" & myStr(1) & ")"

End If

If noTrack Then

ActiveDocument.TrackRevisions = myTrack

With ActiveWindow.View.RevisionsFilter

.Markup = myMarkup

.View = myView

End With

End If

Options.SmartCutPaste = mySmartOpt

Selection.Find.Text = "^p^p" & myStr(1)

Exit Sub

ReportIt:

myErr = Err.Number

If myErr = 4160 Then

DoEvents

gottaList = False

Resume Next

Else

If myErr = 438 Then Beep: Exit Sub

If myErr = 5174 Then

Beep

myPrompt = "Please open your switch list file: " & myListName \_

& CR2 & CR & "I did try looking for file: " & CR2 & myList

myResponse = MsgBox(myPrompt, vbQuestion + vbOKOnly, "MultiSwitch")

Exit Sub

End If

DoEvents

Resume

End If

End Sub

**Sub SearchThenMultiSwitch()**

' Paul Beverley - Version 16.01.21

' Finds one of your words, then calls MultiSwitch to change it

myWords = "apple,pair,orange,lemon"

myrange = 200

For i = 1 To myrange

Selection.MoveRight wdWord, 1

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

If InStr(myWords, "," & Selection & ",") > 0 Then Exit For

DoEvents

Next i

If i > myrange Then

Beep

Else

Application.Run MacroName:="MultiSwitch"

End If

End Sub

**Sub SearchThenChange()**

' Paul Beverley - Version 16.01.21

' Finds one of the words from a list, then changes to its alternate

myWords = "am:was, are:were, is:was, have:had, has:had, can:could,"

myWords = myWords & ",does:did, do:did, goes:went, go:went,"

myrange = 200

myWords = "," & myWords & ","

myWords = Left(myWords, Len(myWords) - 1)

myWords = Replace(myWords, " ", "")

myWords = Replace(myWords, ",,", ",")

myWords = Replace(myWords, ",,", ",")

myWords = Left(myWords, Len(myWords) - 1)

fWord = Split(myWords, ",")

rWord = Split(myWords, ",")

myFindWords = ","

For i = 1 To UBound(fWord)

colonPos = InStr(fWord(i), ":")

f = Left(fWord(i), colonPos - 1)

r = Mid(fWord(i), colonPos + 1)

fWord(i) = f

rWord(i) = r

myFindWords = myFindWords & f & ","

Next i

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

wdsLeft = rng.Words.Count

If myrange > wdsLeft Then myrange = wdsLeft

For i = 1 To myrange

wd = rng.Words(i)

If LCase(wd) <> UCase(wd) Then

wd = Replace(wd, " ", "")

Debug.Print wd

If InStr(myFindWords, "," & wd & ",") > 0 Then Exit For

End If

DoEvents

Next i

If i > myrange Then

rng.Words(i - 1).Select

Beep

Else

For j = 1 To UBound(fWord)

If wd = fWord(j) Then Exit For

Next j

newWord = rWord(j)

rng.Words(i).Select

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.TypeText Text:=newWord

End If

End Sub

**Sub ClipboardLoader()**

' Paul Beverley - Version 06.01.18

' Puts text in clipboard from a menu

m = "|"

m = m & "e - email =paul@archivepub.co.uk|"

m = m & "n - Paul Beverley =Paul Beverley|"

m = m & "a - Address =46 Nightingale Drive, Norwich NR8 6TR|"

m = m & "u - UserName =PaulBeverley66|"

m = m & "p - Password =Collywobble\_543|"

myItem = Split(m, "|")

myFullPrompt = ""

For i = 1 To UBound(myItem) - 1

j = myItem(i)

txt = Left(j, InStr(j, "=") - 1)

myFullPrompt = myFullPrompt & txt & vbCr

Next i

myResponse = InputBox(myFullPrompt & vbCr, "ClipboardLoader")

i = InStr(m, "|" & myResponse)

If i = 0 Then Beep: Exit Sub

txt = Mid(m, i)

i = InStr(txt, "=")

txt = Mid(txt, i + 1)

i = InStr(txt, "|")

txt = Left(txt, i - 1)

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=txt

Selection.MoveStart , -(Len(txt))

Selection.range.Style = ActiveDocument.Styles(wdStyleNormal)

Selection.Font.Reset

Selection.Copy

WordBasic.EditUndo

WordBasic.EditUndo

WordBasic.EditUndo

End Sub

**Sub ClipPaste()**

' Paul Beverley - Version 15.04.20

' Collects and pastes an item from a clip list

myLabel = "#"

useDedicatedFile = True

myClipFile = "zClipboard.docx"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff"

textOnly = False

Set myDoc = ActiveDocument

' Go and look for the clip file

If useDedicatedFile = False Then myClipFile = "Document"

On Error GoTo ReportIt

gottaList = False

For i = 1 To Application.Windows.Count

Set thisFile = Application.Windows(i).Document

If InStr(thisFile.Name, myClipFile) > 0 \_

And myDoc.Name <> thisFile.Name \_

And thisFile.Content.Characters(1) = myLabel Then

Set listDoc = Application.Windows(i).Document

gottaList = True

Exit For

End If

Next i

If gottaList = False Then

If useDedicatedFile = True Then

Documents.Open myFolder & myClipFile

Set listDoc = ActiveDocument

myDoc.Activate

Else

Beep

myResponse = MsgBox("Can't find a clipboard file", \_

vbOKOnly, "ClipStore")

Exit Sub

End If

End If

Set rng = listDoc.Content

txt = rng.Text

labelPos = InStr(txt, myLabel)

maxClips = 0

Do

txt = Mid(txt, labelPos + 1)

myNum = Val(txt)

labelPos = InStr(txt, myLabel)

If myNum > maxClips Then maxClips = myNum

Loop Until labelPos = 0

Do

thisNumber = InputBox("Clip number?", "ClipPaste")

myNumber = Val(thisNumber)

If thisNumber = "0" Then

Set rng = listDoc.Content

rng.Collapse wdCollapseEnd

rng.MoveStartUntil cset:="#", Count:=wdBackward

myNumber = Val(rng.Text)

asdgfas = 0

End If

If myNumber > maxClips Then Beep

If myNumber = 0 Then

Beep

Exit Sub

End If

Loop Until myNumber <= maxClips

Set rng = listDoc.Content

myStart = InStr(rng.Text, myLabel & Trim(Str(myNumber)) \_

& vbCr) + 1 + Len(myNumber)

rng.Start = myStart

endClip = InStr(rng.Text, myLabel)

If endClip > 0 Then

rng.End = myStart + endClip - 2

Else

rng.End = listDoc.Content.End - 2

End If

rng.Copy

If textOnly = True Then

Selection.PasteSpecial DataType:=wdPasteText

Else

Selection.Paste

End If

Exit Sub

ReportIt:

If Err.Number = 5174 Then

Err.Clear

Beep

myClipFile = Replace(myClipFile, ".docx", "")

myResponse = MsgBox("Can't find your clipboard file: """ \_

& myClipFile & """", vbOKOnly, "ClipStore")

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub ClipPasteTextOnly()**

' Paul Beverley - Version 15.04.20

' Collects and pastes an item from a clip list

myLabel = "#"

useDedicatedFile = True

myClipFile = "zClipboard.docx"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff"

textOnly = True

Set myDoc = ActiveDocument

' Go and look for the clip file

If useDedicatedFile = False Then myClipFile = "Document"

On Error GoTo ReportIt

gottaList = False

For i = 1 To Application.Windows.Count

Set thisFile = Application.Windows(i).Document

If InStr(thisFile.Name, myClipFile) > 0 \_

And myDoc.Name <> thisFile.Name \_

And thisFile.Content.Characters(1) = myLabel Then

Set listDoc = Application.Windows(i).Document

gottaList = True

Exit For

End If

Next i

If gottaList = False Then

If useDedicatedFile = True Then

Documents.Open myFolder & myClipFile

Set listDoc = ActiveDocument

myDoc.Activate

Else

Beep

myResponse = MsgBox("Can't find a clipboard file", \_

vbOKOnly, "ClipStore")

Exit Sub

End If

End If

Set rng = listDoc.Content

txt = rng.Text

labelPos = InStr(txt, myLabel)

maxClips = 0

Do

txt = Mid(txt, labelPos + 1)

myNum = Val(txt)

labelPos = InStr(txt, myLabel)

If myNum > maxClips Then maxClips = myNum

Loop Until labelPos = 0

Do

thisNumber = InputBox("Clip number?", "ClipPaste")

myNumber = Val(thisNumber)

If myNumber > maxClips Then Beep

If myNumber = 0 Then

Beep

Exit Sub

End If

Loop Until myNumber <= maxClips

myStart = InStr(rng.Text, myLabel & Trim(Str(myNumber)) \_

& vbCr) + 1 + Len(myNumber)

rng.Start = myStart

endClip = InStr(rng.Text, myLabel)

If endClip > 0 Then

rng.End = rng.Start + endClip - 2

Else

rng.End = listDoc.Content.End - 2

End If

rng.Copy

If textOnly = True Then

Selection.PasteSpecial DataType:=wdPasteText

Else

Selection.Paste

End If

Exit Sub

ReportIt:

If Err.Number = 5174 Then

Err.Clear

Beep

myClipFile = Replace(myClipFile, ".docx", "")

myResponse = MsgBox("Can't find your clipboard file: """ \_

& myClipFile & """", vbOKOnly, "ClipStore")

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub ClipPaste\_1()**

' Paul Beverley - Version 20.03.20

' Collects and pastes a numbered item from a clip list

myClip = 1

myLabel = "#"

textOnly = False

useDedicatedFile = True

myClipFile = "zClipboard.docx"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff"

' Go and look for the clip file

Set myDoc = ActiveDocument

If useDedicatedFile = False Then myClipFile = "Document"

On Error GoTo ReportIt

gottaList = False

For i = 1 To Application.Windows.Count

Set thisFile = Application.Windows(i).Document

If InStr(thisFile.Name, myClipFile) > 0 \_

And myDoc.Name <> thisFile.Name \_

And thisFile.Content.Characters(1) = myLabel Then

Set listDoc = Application.Windows(i).Document

gottaList = True

Exit For

End If

Next i

If gottaList = False Then

If useDedicatedFile = True Then

Documents.Open myFolder & myClipFile

Set listDoc = ActiveDocument

myDoc.Activate

Else

Beep

myResponse = MsgBox("Can't find a clipboard file", \_

vbOKOnly, "ClipStore")

Exit Sub

End If

End If

Set rng = listDoc.Content

myNumber = Trim(Str(myClip))

myStart = InStr(rng.Text, myLabel & myNumber \_

& vbCr) + 1 + Len(myNumber)

rng.Start = myStart

endClip = InStr(rng.Text, myLabel)

If endClip > 0 Then

rng.End = rng.Start + endClip - 2

Else

rng.End = listDoc.Content.End - 2

End If

rng.Copy

If textOnly = True Then

Selection.PasteSpecial DataType:=wdPasteText

Else

Selection.Paste

End If

Exit Sub

ReportIt:

If Err.Number = 5174 Then

Err.Clear

Beep

myClipFile = Replace(myClipFile, ".docx", "")

myResponse = MsgBox("Can't find your clipboard file: """ \_

& myClipFile & """", vbOKOnly, "ClipStore")

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub ClipStore()**

' Paul Beverley - Version 20.03.20

' Copies the selected text into a clip list

maxClips = 6

myLabel = "#"

useDedicatedFile = True

myClipFile = "zClipboard.docx"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff"

If useDedicatedFile = False Then myClipFile = "Document"

If Selection.Start = Selection.End Then

ss = Selection.Start

Selection.Paste

Selection.Start = ss

Selection.Copy

WordBasic.EditUndo

Else

Selection.Copy

End If

Set myDoc = ActiveDocument

Set rng = Selection.range.Duplicate

' Go and look for the clip file

On Error GoTo ReportIt

gottaList = False

For i = 1 To Application.Windows.Count

Set thisFile = Application.Windows(i).Document

If InStr(thisFile.Name, myClipFile) > 0 \_

And myDoc.Name <> thisFile.Name \_

And thisFile.Content.Characters(1) = myLabel Then

Set listDoc = Application.Windows(i).Document

gottaList = True

Exit For

End If

Next i

If gottaList = False Then

If useDedicatedFile = True Then

Documents.Open myFolder & myClipFile

Else

Documents.Add

End If

Else

listDoc.Activate

End If

If InStr(ActiveDocument.Content.Text, myLabel) = 0 Then

nowClip = 0

Else

Selection.EndKey Unit:=wdStory

Selection.MoveEndUntil cset:=myLabel, Count:=wdBackward

Selection.MoveEndUntil cset:=vbCr, Count:=wdForward

nowClip = Val(Selection)

End If

newClip = (nowClip Mod maxClips) + 1

newClipText = myLabel & Trim(Str(newClip))

oldClipPos = InStr(ActiveDocument.Content.Text, newClipText)

If oldClipPos > 0 Then

Selection.Start = oldClipPos + 2

Selection.End = ActiveDocument.Content.End

nextClipPos = InStr(Selection.range.Text, myLabel)

If nextClipPos > 0 Then

Selection.End = Selection.Start + nextClipPos - 1

Selection.Start = 0

Selection.Delete

End If

End If

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=newClipText & vbCr

Selection.Paste

Selection.TypeText Text:=vbCr

myDoc.Activate

rng.Select

Exit Sub

ReportIt:

If Err.Number = 5174 Then

Err.Clear

Beep

myClipFile = Replace(myClipFile, ".docx", "")

myResponse = MsgBox("Can't find your clipboard file: """ \_

& myClipFile & """", vbOKOnly, "ClipStore")

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub WordSwitch()**

' Paul Beverley - Version 06.02.21

' Scripted single-word switching

listName = "zzSwitchList"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff"

doaBeep = True

maxWords = 30

specChar = ">"

doAnErrorBeep = False

defaultList = myFolder & listName

On Error GoTo ReportIt

defaultLoaded = False

Set thisDoc = ActiveDocument

cursorPosn = Selection.Start

dirName = ActiveDocument.Path

' Go and look for the list file

gottaList = False

For i = 1 To Documents.Count

Set dcu = Documents(i)

If InStr(dcu.Name, listName) > 0 Then

Set listDoc = dcu

gottaList = True

End If

Next i

If gottaList = False Then

Documents.Open dirName & "\" & listName

Else

listDoc.Activate

End If

carryOn:

' Create one long string of all the words

myWds = ""

For Each myPara In ActiveDocument.Paragraphs

theLine = myPara

If InStr(theLine, specChar) > 0 Then

myWds = myWds & "|" & Replace(theLine, Chr(13), "")

myWds = Replace(myWds, "^=", ChrW(8211))

myWds = Replace(myWds, "^+", ChrW(8212))

myWds = Replace(myWds, "^32", " ")

End If

Next myPara

myWds = myWds & "|"

' ActiveDocument.ActiveWindow.WindowState = wdWindowStateMinimize

thisDoc.Activate

wNum = Application.Windows.Count

pNum = ActiveDocument.ActiveWindow.Panes.Count

If Selection.Start <> cursorPosn And wNum > 1 Then \_

Application.Windows(1).Activate

If Selection.Start <> cursorPosn And pNum > 1 Then \_

ActiveWindow.Panes(1).Activate

If defaultLoaded = True Then StatusBar = \_

">>>>>>>> Loaded from your default word list <<<<<<<<"

If Selection.LanguageID = wdEnglishUS Then

myWds = Replace(myWds, "%" & specChar & " per cent,", \_

"%" & specChar & " percent,")

End If

startWas = Selection.Start

' Set position at start of first word

Selection.MoveLeft , 1

Selection.Expand wdWord

If Len(Selection) < 4 Then

Selection.MoveRight , 1

Selection.Expand wdWord

End If

Selection.Collapse wdCollapseStart

i = 1

Do

' Select the whole of the word

Selection.MoveEnd wdWord, 1

endWord = Selection.End

If Right(Selection, 1) = " " Then Selection.MoveEnd , -1

' check if this "word" includes a close single curly quote

If Right(Selection, 1) = ChrW(8217) Then Selection.MoveEnd , -1

apoPos = InStr(Selection, ChrW(8217))

If apoPos > 0 Then Selection.End = Selection.Start + apoPos - 1

thisWd = Selection

' Check in the list of words to see if it's there

wordPos = InStr(myWds, "|" & thisWd & specChar)

If wordPos > 0 Then

' If it's in the list, find the replacement word(s)

myWds = Right(myWds, Len(myWds) - wordPos - 1 - Len(thisWd))

newWd = Left(myWds, InStr(myWds, "|") - 1)

' If it starts with "!", delete it and delete

' the previous character in the text

If Len(newWd) > 0 Then

If Asc(newWd) = 33 Then

newWd = Right(newWd, Len(newWd) - 1)

Selection.Start = Selection.Start - 1

End If

Selection.TypeText newWd

Selection.Start = Selection.End

If Len(newWd) > 1 Then Selection.MoveEnd , -1

Else

Selection.Delete

End If

' We've finished, so stop

Exit Sub

Else

' get next char, to check for %

thisStart = Selection.Start

Selection.Start = endWord

nextchar = Selection

Selection.Start = thisStart

' Is it a numeral?

If UCase(thisWd) = LCase(thisWd) And Asc(Left(thisWd, 1)) > 47 \_

And Asc(Left(thisWd, 1)) < 58 And nextchar <> "%" Then GoTo NumToWords

Selection.Start = endWord

End If

i = i + 1

Loop Until i > maxWords

' If no word found beep for warning

If doaBeep = True Then Beep

Selection.End = startWas

Exit Sub

NumToWords:

startHere = Selection.Start

If Right(Selection, 1) = " " Then Selection.MoveEnd , -1

' check if this "word" includes a close single curly quote

If Right(Selection, 1) = ChrW(8217) Then Selection.MoveEnd , -1

' or a hard space

If Right(Selection, 1) = ChrW(160) Then Selection.MoveEnd , -1

Set rng = ActiveDocument.Range

rng.Start = Selection.End

rng.End = Selection.End + 2

moreText = rng

i = Asc(Right(moreText, 1)) - 48

j = Val(Right(moreText, 1))

linkChar = Left(moreText, 1)

If (linkChar = "," Or linkChar = " " Or linkChar \_

= ChrW(160)) And i = j Then

sh = Selection.Start

Selection.Start = Selection.End + 1

Selection.TypeBackspace

Selection.MoveEnd , 3

Selection.Start = sh

End If

' Create a field containing the digits and a special format code

Selection.Fields.Add Range:=Selection.Range, \_

Type:=wdFieldEmpty, Text:="= " + Selection + " \\* CardText", \_

PreserveFormatting:=True

' Select the field and copy it

Selection.MoveStart Unit:=wdCharacter, Count:=-1

Selection.Copy

' Paste the text unformatted, replacing the field

Selection.PasteSpecial Link:=False, DataType:=wdPasteText, \_

Placement:=wdInLine, DisplayAsIcon:=False

Selection.Start = startHere

numWords = Selection

If Right(numWords, 4) <> "dred" Then numWords = \_

Replace(numWords, "hundred", "hundred and")

If InStr(numWords, "hundred") > 0 Then

numWords = Replace(numWords, "thousand", "thousand,")

Else

If Right(numWords, 4) <> "sand" Then numWords = \_

Replace(numWords, "thousand", "thousand and")

End If

Selection.TypeText numWords

Selection.MoveRight Unit:=wdCharacter, Count:=1

Exit Sub

ReportIt:

If Err.Number = 5174 Then

If doAnErrorBeep = True Then Beep

defaultLoaded = True

Documents.Open defaultList

Resume carryOn

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub CharacterSwitch()**

' Paul Beverley - Version 06.02.21

' Scripted character switching

specChar = "\_"

listName = "zzSwitchList"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff\"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff/"

doaBeep = False

maxChars = 200

doAnErrorBeep = False

On Error GoTo ReportIt

defaultList = myFolder & listName

defaultLoaded = False

Set thisDoc = ActiveDocument

cursorPosn = Selection.Start

dirName = ActiveDocument.Path

' Go and look for the list file

gottaList = False

For i = 1 To Documents.Count

Set dcu = Documents(i)

If InStr(dcu.Name, listName) > 0 Then

Set listDoc = dcu

gottaList = True

End If

Next i

If gottaList = False Then

Documents.Open dirName & "\" & listName

Else

listDoc.Activate

End If

carryOn:

' Create one long string of all the words

myWds = ""

For Each myPara In ActiveDocument.Paragraphs

theLine = myPara

If InStr(theLine, specChar) > 0 Then

myWds = myWds & "|" & Replace(theLine, Chr(13), "")

myWds = Replace(myWds, "^=", ChrW(8211))

myWds = Replace(myWds, "^+", ChrW(8212))

myWds = Replace(myWds, "^32", " ")

End If

Next myPara

myWds = myWds & "|"

' ActiveDocument.ActiveWindow.WindowState = wdWindowStateMinimize

thisDoc.Activate

wNum = Application.Windows.Count

pNum = ActiveDocument.ActiveWindow.Panes.Count

If Selection.Start <> cursorPosn And wNum > 1 Then \_

Application.Windows(1).Activate

If Selection.Start <> cursorPosn And pNum > 1 Then \_

ActiveWindow.Panes(1).Activate

If defaultLoaded = True Then StatusBar = \_

">>>>>>>> Loaded from your default word list <<<<<<<<"

If Selection.LanguageID = wdEnglishUS Then

myWds = Replace(myWds, "%" & specChar & " per cent", \_

"%" & specChar & " percent")

End If

For i = 1 To maxChars

' Select the character

Selection.End = Selection.Start + 1

thisChar = Selection

' Check in the list of words to see if it's there

wordPos = InStr(myWds, "|" & thisChar & specChar)

If wordPos > 0 Then

' If it's in the list, find the replacement text

myWds = Right(myWds, Len(myWds) - wordPos - 1 - Len(thisChar))

newWd = Left(myWds, InStr(myWds, "|") - 1)

Selection.TypeText newWd

Selection.Start = Selection.End

' We've found it, so finish

Exit Sub

Else

Selection.Start = Selection.End

End If

Next

' If no character match found beep for warning

If doaBeep = True Then Beep

Selection.End = startWas

Exit Sub

ReportIt:

If Err.Number = 5174 Then

If doAnErrorBeep = True Then Beep

defaultLoaded = True

Documents.Open defaultList

Resume carryOn

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub CentreText()**

**'** Paul Beverley - Version 07.05.10

' Centres the text

If Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter Then

Selection.ParagraphFormat.Alignment = wdAlignParagraphLeft

Else

Selection.ParagraphFormat.Alignment = wdAlignParagraphCenter

End If

End Sub

**Sub ListUcase()**

**'** Paul Beverley - Version 07.01.10

' Upper case initial letter of list item

Selection.End = Selection.Start

With Selection.Find

.Text = "^0149^t"

.Replacement.Text = ""

.Format = False

.MatchCase = False

.MatchWildcards = False

.Execute

End With

Selection.MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdExtend

Selection.Range.Case = wdUpperCase

' Now remove selection

Selection.Collapse wdCollapseEnd

End Sub

**Sub ListUcaseAll()**

**'** Paul Beverley - Version 07.01.10

' Upper case initial letter of all list items

Selection.End = Selection.Start

With Selection.Find

.Text = "^0149^t"

.Replacement.Text = ""

.Format = False

.Wrap = False

' .Wrap = wdFindContinue

.MatchCase = False

.MatchWildcards = False

End With

Do While Selection.Find.Execute

Selection.MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdExtend

Selection.Range.Case = wdUpperCase

' Now move on past the selected character

Selection.Collapse wdCollapseEnd

Loop

End Sub

**Sub AutoListLcaseAll()**

**'** Paul Beverley - Version 30.01.12

' Lowercase initial letter of all auto-bulleted/numbered list items

If Selection.End - Selection.Start < 3 Then

myResponse = MsgBox("Whole document?!", vbQuestion + vbYesNo, "Lowercase initial letters")

If myResponse = vbNo Then Exit Sub

Set rng = ActiveDocument.Content

Else

Set rng = Selection.Range

End If

For Each myPara In rng.Paragraphs

If Len(myPara) > 1 Then

Set myChar = myPara.Range

myChar.End = myChar.Start + 1

myChar.Case = wdLowerCase

End If

Next myPara

End Sub

**Sub AutoListLcaseOne()**

**'** Paul Beverley - Version 30.01.12

' Lowercase initial letter of one auto-bulleted/numbered list item

Selection.HomeKey Unit:=wdLine

Set rng = Selection.Range

rng.End = rng.Start + 1

rng.Case = wdLowerCase

Selection.MoveDown Unit:=wdLine, Count:=1

End Sub

**Sub ParaWordLengthHighlighter()**

' Paul Beverley - Version 08.05.18

' Highlights all paragraphs of a range of word lengths

myMinWords = 50

myMaxWords = 20000

myColour = wdBrightGreen

showAsYouGo = True

For Each myPara In ActiveDocument.Paragraphs

If Len(myPara.range) > 3 Then

paraText = Trim(myPara.range.Text)

paraText = Replace(paraText, " ", " ")

paraText = Replace(paraText, ChrW(8211) & " ", "")

If Asc(paraText) = Asc(vbTab) Then paraText = Mid(paraText, 2)

totChars = Len(paraText)

paraText = Replace(paraText, " ", "")

paraText = Replace(paraText, vbCr, "")

paraText = Replace(paraText, vbTab, "")

paraWords = totChars - Len(paraText)

If paraWords < myMaxWords + 1 And paraWords > myMinWords Then

If showAsYouGo = True Then myPara.range.Select

myPara.range.HighlightColorIndex = myColour

End If

End If

Next

Beep

End Sub

**Sub TitleHeadingCapper()**

' Paul Beverley - Version 18.02.20

' Uppercase initial letter of all major words (title case)

trackIt = True

maxWordsInTitle = 40

' If the headings are all in caps, say True

allIsInCaps = True

' Do you want an initial cap after a colon?

colonCap = True

' Do you want an initial cap after a hyphen?

hyphenCap = True

' List of lowercase words, each surrounded by spaces

lclist = " a an and as at by for from hoc if in is it into of "

lclist = lclist & " on or s that the to with "

Selection.Collapse wdCollapseStart

Selection.Expand wdParagraph

If Len(Selection) < 3 Then

Selection.Collapse wdCollapseEnd

Beep

Exit Sub

End If

Selection.MoveEnd , -1

If Selection.range.Words.Count > maxWordsInTitle Then

MsgBox "Too long!"

Exit Sub

End If

Set rng = Selection.range

wasText = rng.Text

'If a word is in an acronym, e.g. BBC, ignore it

If allIsInCaps = False Then

For Each wd In rng.Words

If UCase(wd) = wd And Len(Trim(wd)) > 1 Then wd.Font.Shadow = True

Next

End If

' If there's a tag or a number, jump past it

Set rng2 = rng.Duplicate

charOne = Left(rng2, 1)

If charOne = "<" Then

rng2.Start = rng2.Start + InStr(rng2, ">") + 1

charOne = Left(rng2, 1)

End If

If LCase(charOne) = UCase(charOne) Then

Do

rng2.MoveStart , 1

sdgfdf = rng2.Text

Loop Until Asc(rng2) < 33

rng2.MoveStart , 1

End If

rng2.End = rng2.Start + 1

rng2.Case = wdUpperCase

rng.Start = rng2.Start

endHere = rng.End

startHere = rng.Start

rng.Case = wdTitleWord

Set rng2 = rng.Duplicate

' Force lower case after hyphen?

If hyphenCap = False Then

rng2.Start = startHere

Do

' Capitalise after a colon if option set

myText = rng2.Text

hyphenPos = InStr(myText, "-")

If hyphenPos > 0 Then

rng2.MoveStart , hyphenPos

rng2.End = rng2.Start + 1

rng2.Case = wdLowerCase

rng2.End = endHere

End If

Loop Until hyphenPos = 0

End If

For wrd = 2 To rng.Words.Count

thisWord = Trim(rng.Words(wrd))

thisWord = " " & LCase(thisWord) & " "

If InStr(lclist, thisWord) Then

rng.Words(wrd).Case = wdLowerCase

End If

Next wrd

' Force upper case after colon?

If colonCap = True Then

rng.Start = startHere

Do

' Capitalise after a colon if option set

myText = rng.Text

colonPos = InStr(myText, ": ")

If colonPos > 0 Then

rng.MoveStart , colonPos + 1

rng.End = rng.Start + 1

rng.Case = wdUpperCase

rng.End = endHere

End If

Loop Until colonPos = 0

End If

Set rng = Selection.range

For Each wd In rng.Words

If wd.Font.Shadow = True Then

wd.Case = wdUpperCase

End If

Next

Selection.Font.Shadow = False

Set r = Selection.range.Duplicate

nowText = r.Text

If ActiveDocument.TrackRevisions = True And \_

trackIt = True Then

myExtra = 0

For i = 1 To Len(wasText) - 2

w = Mid(wasText, i, 1)

n = Mid(nowText, i, 1)

If n <> w Then

ActiveDocument.TrackRevisions = False

r.End = Selection.Start + i + myExtra

r.Start = r.End - 1

r.Text = w

ActiveDocument.TrackRevisions = True

r.Text = n

myExtra = myExtra + 1

End If

Next i

End If

Selection.Start = endHere + myExtra

Selection.End = endHere + myExtra

Selection.MoveRight , 1

End Sub

**Sub TitleInQuotesCapper()**

' Paul Beverley - Version 25.08.20

' Uppercases the initial letter of all major words (title case)

findSingles = True

findDoubles = True

' Do you want an initial cap after a colon?

colonCap = False

' Do you want an initial cap after a hyphen?

hyphenCap = False

' List of lowercase words, each surrounded by spaces

lclist = " a an and at by for from in into is it of "

lclist = lclist & " on or that the their they to "

lclist = lclist & " we with "

myFind = ""

If findSingles = True Then myFind = ChrW(8216)

If findDoubles = True Then myFind = myFind & ChrW(8220)

Debug.Print myFind

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & myFind & "]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do

Selection.Collapse wdCollapseEnd

Set rng = Selection.Range

DoEvents

rng.MoveEndUntil cset:=ChrW(8221) & ChrW(8217) & "'"")", Count:=wdForward

Do

Set rng2 = rng.Duplicate

rng2.Start = rng2.End + 1

rng2.End = rng2.End + 1

keepGoing = False

If UCase(rng2) <> LCase(rng2) Then

rng.End = rng.End + 1

rng.MoveEndUntil cset:=ChrW(8221) & ChrW(8217) & \_

"'"")", Count:=wdForward

keepGoing = True

End If

DoEvents

Loop Until keepGoing = False

rng.MoveStartUntil cset:=ChrW(8220) & ChrW(8216) & "'""(", Count:=wdBackward

endHere = rng.End

startHere = rng.Start

rng.Case = wdTitleWord

rng.Select

For Each wd In rng.Words

If UCase(wd) = wd And Len(Trim(wd)) > 1 Then wd.Font.Shadow = True

Next

Set rng2 = rng.Duplicate

' Force lower case after hyphen?

If hyphenCap = False Then

rng2.Start = startHere

Do

' Capitalise after a colon if option set

myText = rng2.Text

hyphenPos = InStr(myText, "-")

If hyphenPos > 0 Then

rng2.MoveStart , hyphenPos

rng2.End = rng2.Start + 1

rng2.Case = wdLowerCase

rng2.End = endHere

End If

Loop Until hyphenPos = 0

End If

For wrd = 2 To rng.Words.Count

thisWord = Trim(rng.Words(wrd))

thisWord = " " & LCase(thisWord) & " "

If InStr(lclist, thisWord) Then

rng.Words(wrd).Case = wdLowerCase

End If

Next wrd

' Force upper case after colon?

If colonCap = True Then

rng.Start = startHere

Do

' Capitalise after a colon if option set

myText = rng.Text

colonPos = InStr(myText, ": ")

If colonPos > 0 Then

rng.MoveStart , colonPos + 1

rng.End = rng.Start + 1

rng.Case = wdUpperCase

rng.End = endHere

End If

Loop Until colonPos = 0

End If

Set rng = Selection.Range

For Each wd In rng.Words

If wd.Font.Shadow = True Then

wd.Case = wdUpperCase

End If

Next

Selection.Font.Shadow = False

myResponse = MsgBox("Continue?", \_

vbQuestion + vbYesNoCancel, "TitleInQuotesCapper")

Selection.Collapse wdCollapseEnd

If myResponse = vbYes Then

Selection.Find.Execute

End If

Loop Until myResponse <> vbYes Or Selection.Find.Found = False

Beep

End Sub

**Sub TitleInQuotesCapperGlobal()**

' Paul Beverley - Version 25.08.20

' Uppercases the initial letter of all major words (title case)

findSingles = True

findDoubles = True

' Do you want an initial cap after a colon?

colonCap = False

' Do you want an initial cap after a hyphen?

hyphenCap = False

' List of lowercase words, each surrounded by spaces

lclist = " a an and at by for from in into is it of "

lclist = lclist & " on or that the their they to "

lclist = lclist & " we with "

myFind = ""

If findSingles = True Then myFind = ChrW(8216)

If findDoubles = True Then myFind = myFind & ChrW(8220)

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & myFind & "]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do

Selection.Collapse wdCollapseEnd

Set rng = Selection.Range

DoEvents

rng.MoveEndUntil cset:=ChrW(8221) & ChrW(8217) & "'"")", \_

Count:=wdForward

Do

Set rng2 = rng.Duplicate

rng2.Start = rng2.End + 1

rng2.End = rng2.End + 1

keepGoing = False

If UCase(rng2) <> LCase(rng2) Then

rng.End = rng.End + 1

rng.MoveEndUntil cset:=ChrW(8221) & ChrW(8217) & \_

"'"")", Count:=wdForward

keepGoing = True

End If

DoEvents

Loop Until keepGoing = False

rng.MoveStartUntil cset:=ChrW(8220) & ChrW(8216) & "'""(", Count:=wdBackward

endHere = rng.End

startHere = rng.Start

rng.Case = wdTitleWord

rng.Select

For Each wd In rng.Words

If UCase(wd) = wd And Len(Trim(wd)) > 1 Then wd.Font.Shadow = True

Next

Set rng2 = rng.Duplicate

' Force lower case after hyphen?

If hyphenCap = False Then

rng2.Start = startHere

Do

' Capitalise after a colon if option set

myText = rng2.Text

hyphenPos = InStr(myText, "-")

If hyphenPos > 0 Then

rng2.MoveStart , hyphenPos

rng2.End = rng2.Start + 1

rng2.Case = wdLowerCase

rng2.End = endHere

End If

Loop Until hyphenPos = 0

End If

For wrd = 2 To rng.Words.Count

thisWord = Trim(rng.Words(wrd))

thisWord = " " & LCase(thisWord) & " "

If InStr(lclist, thisWord) Then

rng.Words(wrd).Case = wdLowerCase

End If

Next wrd

' Force upper case after colon?

If colonCap = True Then

rng.Start = startHere

Do

' Capitalise after a colon if option set

myText = rng.Text

colonPos = InStr(myText, ": ")

If colonPos > 0 Then

rng.MoveStart , colonPos + 1

rng.End = rng.Start + 1

rng.Case = wdUpperCase

rng.End = endHere

End If

Loop Until colonPos = 0

End If

Set rng = Selection.Range

For Each wd In rng.Words

If wd.Font.Shadow = True Then

wd.Case = wdUpperCase

End If

Next

Selection.Font.Shadow = False

Selection.Collapse wdCollapseEnd

Selection.Find.Execute

Loop Until Selection.Find.Found = False

Beep

End Sub

**Sub TitleRemoveQuotesAndCaps()**

' Paul Beverley - Version 13.12.16

' Removes quotes from current sentence, then makes it lowercase

Selection.MoveUntil cset:=ChrW(8220) & ChrW(8216) & """'", Count:=wdBackward

Selection.TypeBackspace

Selection.MoveUntil cset:=ChrW(8221) & ChrW(8217) & """'", Count:=wdForward

Selection.Delete

Selection.MoveLeft , 2

Call TitleUnCapper

End Sub

**Sub TitleInSquaresCapperFR()**

' Paul Beverley - Version 22.01.20

' Uppercase initial letter of all major words (title case)

' Do you want an initial cap after a colon?

colonCap = True

' Do you want an initial cap after a hyphen?

hyphenCap = False

forText = "["

afterText = "]"

' List of lowercase words, each surrounded by spaces

'lclist = " a an and at by for from in into is it of "

'lclist = lclist & " on or that the their they to "

'lclist = lclist & " we with "

lcList = " de le la les des du au et dans sur "

lcList = lcList & " un une en à pour chez ou"

myBits = " d' l' d" & ChrW(8216) & " l" & ChrW(8216)

Set rng = Selection.range.Duplicate

rng.End = rng.Start + 1000

myStart = InStr(rng, forText)

myEnd = InStr(rng, afterText) - 1

rng.End = rng.Start + myEnd

rng.Start = rng.Start + myStart

If LCase(Left(rng, 1)) = UCase(Left(rng, 1)) Then

rng.Start = rng.Start + 1

End If

If LCase(Right(rng, 1)) = UCase(Right(rng, 1)) Then

rng.End = rng.End - 1

End If

isColon = False

isHyphen = False

For Each wd In rng.Words

wasColon = isColon

wasHyphen = isHyphen

tst = wd.Text

tst = LCase(Trim(tst))

isWord = LCase(tst) <> UCase(tst)

isColon = (Left(tst, 1) = ":")

isHyphen = (Left(tst, 1) = "-")

isTitle = (InStr(lcList, " " & tst & " ") = 0)

If isWord Then

Set rng2 = wd.Duplicate

If LCase(Left(rng2, 1)) = UCase(Left(rng2, 1)) Then

rng2.Start = rng2.Start + 1

End If

isApostrophe = (InStr(myBits, " " & Left(LCase(rng2.Text), 2) & " ") > 0)

Debug.Print rng2, isTitle; isApostrophe

If isTitle Then

If isApostrophe Then

If rng2.Characters(3) <> UCase(rng2.Characters(3)) Then

rng2.Characters(3) = UCase(rng2.Characters(3))

End If

If rng2.Characters(1) <> LCase(rng2.Characters(1)) Then

rng2.Characters(1) = LCase(rng2.Characters(1))

End If

Else

If rng2.Characters(1) <> UCase(rng2.Characters(1)) Then

rng2.Characters(1) = UCase(rng2.Characters(1))

End If

End If

End If

If (wasColon And colonCap) Or (wasHyphen And hyphenCap) = True Then

If rng2.Characters(1) <> UCase(rng2.Characters(1)) Then

rng2.Characters(1) = UCase(rng2.Characters(1))

End If

End If

If Not isTitle Then

If (wasColon And Not colonCap) Or (wasHyphen And Not hyphenCap) Then

If rng2.Characters(1) <> LCase(rng2.Characters(1)) Then

rng2.Characters(1) = LCase(rng2.Characters(1))

End If

End If

End If

End If

Next wd

rng2.Select

Selection.Collapse wdCollapseEnd

Selection.MoveRight wdWord, 1

End Sub

**Sub AddQuotesAndTitleCap()**

' Paul Beverley - Version 13.12.16

' Puts quotes on sentence, then makes it title case

useDoubleQuotes = True

' i.e. say False for single quotes

Selection.Expand wdSentence

Selection.Collapse wdCollapseStart

If useDoubleQuotes Then

Selection.TypeText Text:=ChrW(8220)

Else

Selection.TypeText Text:=ChrW(8216)

End If

Selection.Expand wdSentence

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

If useDoubleQuotes Then

Selection.TypeText Text:=ChrW(8221)

Else

Selection.TypeText Text:=ChrW(8217)

End If

Selection.MoveLeft , 2

Call TitleInQuotesCapper

End Sub

**Sub TitleUnCapper()**

' Paul Beverley - Version 27.10.15

' Uppercase initial letter only on first word

' Alt-F6

moveOnAfter = False

' If false, it assumes the title is in italic

wholeSentence = True

nextFind = "\<[ABC]\>"

' Do you want an initial cap after a colon?

colonCap = True

Selection.MoveRight Unit:=wdCharacter, Count:=1

If LCase(Selection) = UCase(Selection) Then \_

Selection.MoveRight Unit:=wdWord, Count:=1

' Move to start of current word

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

quoteStart = Selection.Start

Selection.MoveStart wdCharacter, -1

' check the character before the first word

myChar = Selection

openChars = ChrW(8216) & ChrW(8220) & "("

If InStr(openChars & "(", myChar) > 0 Then

allSentence = False

quoteStart = Selection.Start + 1

Else

allSentence = True

End If

If allSentence = True Then

If wholeSentence = True Then

Selection.Sentences(1).Select

myText = Selection

firstChar = Chr(Asc(myText))

If InStr(openChars, firstChar) > 0 Then Selection.MoveStart Count:=1

If firstChar = "<" Then Selection.MoveStart Count:=InStr(myText, ">")

Else

Selection.Start = quoteStart + 1

Do

Selection.MoveRight , 1

Loop Until Selection.Font.Italic = False

Selection.Start = quoteStart

End If

Else

Selection.Sentences(1).Select

myText = Selection

i = Asc(myChar)

' Find the position of the corresponding close character

lenTitle = InStr(myText, Chr(i + 1))

Selection.End = Selection.Start + lenTitle

End If

endNow = Selection.End

' Just check the initial letter is uppercase

Selection.End = Selection.Start

Do

' select more until you meet an alpha character

Selection.MoveEnd , 1

firstBit = Selection

Loop Until UCase(firstBit) <> LCase(firstBit)

Selection.range.Case = wdUpperCase

Selection.Start = Selection.End

Selection.End = endNow

' Lowercase the rest

Selection.range.Case = wdLowerCase

If colonCap = True Then

' Capitalise after a colon if option set

myText = Selection

colonPos = InStr(myText, ": ")

If colonPos > 0 Then

Selection.Start = Selection.Start + colonPos + 1

Selection.End = Selection.Start + 1

Selection.range.Case = wdUpperCase

End If

End If

Selection.Start = endNow

If moveOnAfter = True Then

nextFind = "\<[ABC]\>"

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = nextFind

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

End If

End Sub

**Sub HeadingSentenceCase()**

' Paul Beverley - Version 18.02.20

' Sentence cases this selection or paragraph (but not acronyms)

trackIt = True

allowAreaSelection = False

highlightChange = False

If Selection.Start = Selection.End Or allowAreaSelection = False \_

Then Selection.Expand wdParagraph

If Len(Selection) < 3 Then

Selection.Collapse wdCollapseEnd

Beep

Exit Sub

End If

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

myEnd = Selection.End - 1

Selection.End = Selection.Start + 1

' Skip past <> codes

If Selection = "<" Then

Do

Selection.MoveRight , 1

Loop Until Selection = ">"

Selection.MoveRight , 1

End If

While LCase(Selection) = UCase(Selection)

Selection.MoveRight Unit:=wdWord, Count:=1

Selection.MoveEnd wdCharacter, 1

Wend

myStart = Selection.Start

Set rng = ActiveDocument.Content

Set rng1 = ActiveDocument.Content

For i = myStart + 1 To myEnd

rng.Start = i

rng.End = i + 1

If Len(rng) > 0 Then

a = Asc(rng)

Else

a = 0

End If

If a > 64 And a < 91 Then

rng1.Start = i + 1

rng1.End = i + 2

a = Asc(rng1)

If a > 96 And a < 123 Then

rng1.Start = i - 2

rng1.End = i - 1

If Len(rng1) > 0 Then

a = Asc(rng1)

Else

a = 0

End If

rng1.Start = i - 1

rng1.End = i

b = Asc(rng1)

If Not (a = 46 Or a = 33 Or a = 63 Or b = 13 Or b = 9) Then

rng.Text = LCase(rng.Text)

If highlightChange = True Then rng.HighlightColorIndex = wdGray25

End If

End If

End If

Next i

ActiveDocument.TrackRevisions = myTrack

Selection.Start = myEnd + 1

End Sub

**Sub InitialCapItalicText()**

**'** Paul Beverley - Version 03.12.11

' Initial caps on significant words in italic text in notes

For Each wd In ActiveDocument.Words

If wd.Italic = True Then

myWord = wd

initChar = wd.Characters(1)

If UCase(initChar) <> initChar Then

If InStr(lclist, myWord) = 0 Then

wd.Characters(1) = UCase(initChar)

If doHighlight = True Then wd.HighlightColorIndex = myColour

End If

End If

End If

Next wd

End Sub

**Sub InitialCapItalicInNotes()**

**'** Paul Beverley - Version 03.12.11

' Initial caps on significant words in italic text in notes

' List of lowercase words, each surrounded by spaces

lclist = " a an and at by for from in into is it of "

lclist = lclist & " on or that the their they to "

lclist = lclist & " we with "

myColour = wdYellow

doHighlight = True

If ActiveDocument.Footnotes.Count > 0 Then

For Each fn In ActiveDocument.Footnotes

For Each wd In fn.Range.Words

If wd.Italic = True Then

myWord = wd

initChar = wd.Characters(1)

If UCase(initChar) <> initChar Then

If InStr(lclist, myWord) = 0 Then

wd.Characters(1) = UCase(initChar)

If doHighlight = True Then wd.HighlightColorIndex = myColour

End If

End If

End If

Next wd

Next

End If

End Sub

**Sub CapperMax()**

**'** Paul Beverley - Version 13.11.23

' Uppercases initial letter of all major words (title case)

doTrack = True

' If True then if you click in an italic area of text,

' only the italic bit will be max-capped

onlyItalic = True

' If it's mostly capitals, lowercase them all first

deCapitate = False

' List of lowercase words \*not\* to be uppercased

lclist = " a an and as at by for from if in is it into of "

lclist = lclist & " on or s that the to with "

uppercaseAfterHyphen = False

' uppercaseAfterHyphen = True

' uppercaseAfterColon = False

uppercaseAfterColon = True

If onlyItalic = True And Selection.Start = \_

Selection.End And Selection.Range.Italic = True Then

Set rng = Selection.Range.Duplicate

Do

rng.MoveStart wdWord, -1

DoEvents

Loop Until rng.Italic = 9999999

rng.MoveStart wdWord, 1

Do

rng.MoveEnd wdWord, 1

DoEvents

Loop Until rng.Italic = 9999999

rng.MoveEnd wdWord, -1

rng.Select

End If

' Create a range from the selection but whole words

' or whole paragraph, if nothing was selected

If Selection.Start = Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

Else

Set rng = Selection.Range.Duplicate

If Right(rng.Text, 1) = vbCr Then rng.MoveEnd , -1

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

rng.Start = Selection.Start

End If

rng.Select

Selection.Collapse wdCollapseEnd

hereNow = Selection.End

lclist = " " & lclist & " "

myTrack = ActiveDocument.TrackRevisions

If doTrack = False Then ActiveDocument.TrackRevisions = False

If deCapitate = True Then

numUC = 0

allLIne = rng.Text

For i = 1 To Len(allLIne)

ch = Mid(allLIne, i, 1)

If LCase(ch) <> UCase(ch) And LCase(ch) <> ch Then numUC = numUC + 1

Next i

If numUC > rng.Characters.Count / 2 Then \_

rng.Text = Left(allLIne, 1) & LCase(Mid(allLIne, 2))

Selection.Start = hereNow

End If

numWds = rng.Words.Count

wasChar = ""

wasWdr = ""

For i = 1 To numWds

Set wd = rng.Words(i)

tst = LCase(Trim(wd.Text))

init = Left(wd.Text, 1)

' Debug.Print wd.Text, wasChar, wasWd, init

myCap = UCase(init)

If InStr(lclist, " " & tst & " ") = 0 And myCap <> init \_

And init <> ChrW(8216) And init <> ChrW(8220) Then

If (wasChar = "-" And uppercaseAfterHyphen = True) \_

Or (wasChar = ":" And uppercaseAfterColon = True) \_

Or InStr(":-", wasChar) = 0 Or InStr(wasWd, ":") > 0 \_

Then wd.Characters(1) = myCap

End If

' Fudge to be sure to initial cap after a :

If (InStr(wasWd, ":") > 0 And uppercaseAfterColon = True) \_

Then wd.Characters(1) = myCap

DoEvents

wasChar = init

wasWd = wd.Text

Next i

If uppercaseAfterColon = True Then

wasChar = ""

For i = 1 To numWds

Set wd = rng.Words(i)

If wd.Text = ": " Then

nextWd = rng.Words(i + 1).Text

myInit = Left(nextWd, 1)

If LCase(myInit) = myInit Then \_

rng.Words(i + 1).Characters(1).Text = UCase(myInit)

End If

Next i

End If

ActiveDocument.TrackRevisions = myTrack

myInit = rng.Characters(1)

If UCase(myInit) <> myInit Then \_

rng.Characters(1) = UCase(myInit)

End Sub

**Sub CapperMin()**

' Paul Beverley - Version 12.11.22

' Lowercases initial letter of capitalised words (sentence case)

doTrack = True

' If it's mostly capitals lowercase them all first

deCapitate = True

uppercaseAfterColon = False

' uppercaseAfterColon = True

' List of (starts of) words \*not\* to be lowercased

notLC = " Brit United Kingd States America Engl Welsh Wales "

notLC = notLC & " Scot Irel Irish Europ France French "

' Create a range from the selection but whole words

' or whole paragraph, if nothing was selected

If Selection.Start = Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

Else

Set rng = Selection.Range.Duplicate

rng.MoveEnd , -1

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

rng.Start = Selection.Start

End If

rng.Select

Selection.Collapse wdCollapseEnd

hereNow = Selection.End

notLC = " " & notLC & " "

myTrack = ActiveDocument.TrackRevisions

If doTrack = False Then ActiveDocument.TrackRevisions = False

If deCapitate = True Then

numUC = 0

allLine = rng.Text

For i = 1 To Len(allLine)

ch = Mid(allLine, i, 1)

If LCase(ch) <> UCase(ch) And LCase(ch) <> ch Then numUC = numUC + 1

Next i

If numUC > rng.Characters.Count / 2 Then \_

rng.Text = Left(allLine, 1) & LCase(Mid(allLine, 2))

Selection.Start = hereNow

End If

numWds = rng.Words.Count

wdWas = ""

For i = 2 To numWds

Set wd = rng.Words(i)

init = Left(wd.Text, 1)

myLC = LCase(init)

doLC = True

myWd = Trim(wd.Text)

numUC = 0

If uppercaseAfterColon = True And wdWas = ": " Then

If init = LCase(init) Then wd.Characters(1) = UCase(init)

Else

If myLC <> init And UCase(myWd) <> myWd \_

And init <> ChrW(8216) Then

For j = 1 To Len(myWd)

ch = Mid(myWd, j, 1)

If ch = UCase(ch) And LCase(ch) <> ch Then numUC = numUC + 1

If InStr(notLC, " " & Left(myWd, j) & " ") > 0 Then

doLC = False

Exit For

End If

DoEvents

Next j

If doLC = True And numUC < 2 \_

And myWd <> "I" Then wd.Characters(1) = myLC

End If

End If

DoEvents

wdWas = wd.Text

Next i

If uppercaseAfterColon = False Then

wasChar = ""

For i = 1 To numWds

Set wd = rng.Words(i)

If wd.Text = ": " Then

nextWd = rng.Words(i + 1).Text

myInit = Left(nextWd, 1)

If UCase(myInit) = myInit Then \_

rng.Words(i + 1).Characters(1).Text = LCase(myInit)

End If

Next i

End If

myInit = rng.Characters(1)

If UCase(myInit) <> myInit Then \_

rng.Characters(1) = UCase(myInit)

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub AllCapsToInitialCap()**

**'** Paul Beverley - Version 14.12.11

' Initial cap any words in all caps

Set rng = ActiveDocument.Content

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

' This is the line that sets the min no of letters

.Text = "[A-Z]{3,}"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

If rng.Find.Found = True Then

' This is a highlight line - delete if you don't want it

rng.HighlightColorIndex = wdBrightGreen

rng.Start = rng.Start + 1

rng.Case = wdLowerCase

rng.Start = rng.End

stopNow = False

Else

stopNow = True

End If

Selection.Start = Selection.End

Loop Until stopNow = True

End Sub

**Sub SuperscriptNoteNumber()**

**'** Paul Beverley - Version 11.08.11

' Superscript note number

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Replacement.Text = ""

.Wrap = wdFindContinue

.Execute

End With

startHere = Selection.End

Selection.Start = Selection.End

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " "

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindContinue

.Execute

End With

Selection.End = Selection.Start

Selection.Start = startHere

Selection.Font.Superscript = True

Selection.Start = Selection.End

End Sub

**Sub Italiciser()**

' Paul Beverley - Version 03.06.22

' Toggles italic for next character or selected text

If Selection.End = Selection.Start Then

Application.Run macroName:="Italic"

Else

' If it's a long selection, just change it

If Len(Selection) > 100 Then

Set firstChar = Selection.Range.Characters(1)

Selection.Font.Italic = Not (firstChar.Font.Italic)

Else

' How many italic/roman chars?

Set rng = Selection.Range.Duplicate

charsItalic = 0

For i = 1 To Len(rng.Text)

isItalic = rng.Characters(i).Font.Italic

If isItalic Then charsItalic = charsItalic + 1

Next i

charsRoman = Len(Selection) - charsItalic

' Change to italic or roman, accordingly

If charsItalic > charsRoman Then

Selection.Font.Italic = False

Else

' Don't italicise following space or quote

If Len(Selection) > 1 Then

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

End If

rng.Font.Italic = True

rng.Select

End If

End If

End If

End Sub

**Sub Boldiser()**

' Paul Beverley - Version 03.05.22

' Toggles bold for next character or selected text

If Selection.End = Selection.Start Then

Application.Run macroName:="Bold"

Else

' If it's a long selection, just change it

If Len(Selection) > 100 Then

Set firstChar = Selection.Range.Characters(1)

Selection.Font.Bold = Not (firstChar.Font.Bold)

Else

' How many Bold/roman chars?

Set rng = Selection.Range.Duplicate

charsBold = 0

For i = 1 To Len(rng.Text)

isBold = rng.Characters(i).Font.Bold

If isBold Then charsBold = charsBold + 1

Next i

charsRoman = Len(Selection) - charsBold

' Change to Bold or roman, accordingly

If charsBold > charsRoman Then

Selection.Font.Bold = False

Else

' Don't Boldise following space or quote

If Len(Selection) > 1 Then

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

End If

rng.Font.Bold = True

rng.Select

End If

End If

End If

End Sub

**Sub BoldKill()**

**'** Paul Beverley - Version 11.08.11

' Remove bold from selected text or current line

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If Selection.End = Selection.Start Then

Selection.HomeKey Unit:=wdLine

Selection.MoveDown Unit:=wdLine, Count:=1, Extend:=wdExtend

End If

Selection.Font.Bold = False

Selection.Start = Selection.End

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub** **ItalicKill()**

**'** Paul Beverley - Version 11.08.11

' Removes italic from selected text or current line

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If Selection.End = Selection.Start Then

Selection.HomeKey Unit:=wdLine

Selection.MoveDown Unit:=wdLine, Count:=1, Extend:=wdExtend

End If

Selection.Font.Italic = False

Selection.Start = Selection.End

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub NormaliseText()**

' Paul Beverley - Version 03.04.21

' Sets Normal style and removes (some) formatting

removeHighlight = True

removeColour = True

' To reset some features, not all, set to False:

resetAllFormat = False

' which, if true, makes these redundant

resetBoldItalic = True

resetFontName = True

resetFontSize = True

resetSubSuper = True

If Selection.Start = Selection.End Then Selection.Expand wdParagraph

Set rng = Selection.Range.Duplicate

numParas = Len(rng) - Len(Replace(rng, vbCr, ""))

If Right(rng, 1) = vbCr Then rng.MoveEnd , -1

If numParas > 0 And Right(rng, 1) <> vbCr Then

rng.MoveStartUntil cset:=vbCr, Count:=wdBackward

rng.MoveEndUntil cset:=vbCr, Count:=wdForward

rng.MoveEnd , 1

End If

rng.Select

If resetAllFormat = False Then

If removeHighlight = True Then rng.HighlightColorIndex = wdNoHighlight

If removeColour = True Then rng.Font.Color = wdAuto

If resetSubSuper = True Then

rng.Font.Superscript = False

rng.Font.Subscript = False

End If

If resetBoldItalic = True Then

rng.Bold = False

rng.Italic = False

End If

If resetFontName = True Then

myFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

rng.Font.Name = myFont

End If

If resetFontSize = True Then

mySize = ActiveDocument.Styles(wdStyleNormal).Font.Size

rng.Font.Size = mySize

End If

Else

rng.Style = wdStyleNormal

On Error GoTo ReportIt

rng.Font.Reset

End If

Selection.Collapse wdCollapseEnd

Exit Sub

ReportIt:

Beep

End Sub

**Sub ItaliciseVariable()**

' Paul Beverley - Version 25.01.22

' Runs along, finds sets of alpha chars and italicises them

greekItalic = True

ucaseGreekItalic = False

avoidWords = ",to,and,or,at,sin,cos,tan"

doTrack = False

maxChars = 10

myTrack = ActiveDocument.TrackRevisions

If doTrack = False Then ActiveDocument.TrackRevisions = False

If Selection.Start = Selection.End Then

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

gotStart = False

For i = 1 To Len(rng)

myChar = rng.Characters(i)

isText = (LCase(myChar) <> UCase(myChar))

If gotStart = False And isText Then

Selection.Start = rng.Start + i - 1

gotStart = True

End If

If gotStart = True And isText = False Then

Selection.End = rng.Start + i - 1

Exit For

End If

Next i

If InStr(avoidWords, "," & Selection & ",") = 0 Then

numChars = Len(Selection)

myStart = Selection.Start

Set rng = Selection.Range

For i = 1 To numChars

gotVar = False

rng.End = myStart + i

rng.Start = rng.End - 1

ch = rng.Text

A = Asc(ch)

u = AscW(ch)

' Ordinary letter

If A = u Then gotVar = True

If greekItalic = True Then

If A = Asc("?") Then gotVar = True

If u > 915 And u < 970 Then

gotVar = True

If ucaseGreekItalic = False And u < 945 Then gotVar = False

End If

If ucaseGreekItalic = False And u < -3999 Then gotVar = False

End If

If gotVar = True Then

rng.Font.Italic = Not (isItalic)

Else

rng.Select

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

If i > maxChars Then

Beep

MsgBox "Stopped after " & Trim(Str(maxChars)) & " characters"

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

Next i

End If

Selection.Collapse wdCollapseEnd

Else

If Selection <> " " Then Selection.Font.Italic = Not (Selection.Font.Italic)

Selection.Collapse wdCollapseEnd

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub ItaliciseOneVariable()**

' Paul Beverley - Version 06.04.16

' Run along to find the next single alpha char and italicise it

' (but if a selection is made, italicise all alphas)

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If Selection.Start <> Selection.End Then

Selection.Font.Italic = False

n = Len(Selection)

Set rng = Selection.range

For i = 1 To n

myTest = rng.Characters(i)

If UCase(myTest) <> LCase(myTest) Then

rng.Characters(i).Font.Italic = True

End If

Next i

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

sdfg = Asc(Selection)

Do While LCase(Selection) = UCase(Selection) And Asc(Selection) <> 40

Selection.MoveRight , 1

Loop

Selection.MoveEnd , 1

Selection.Font.Italic = True

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub Romanise()**

' Paul Beverley - Version 24.10.16

' Removes italic from the next set of italic characters.

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.MoveEnd , 1

Do

Selection.MoveEnd , 1

Selection.MoveStart , 1

Loop Until Selection.Font.Italic = True

startHere = Selection.Start

Do

Selection.MoveEnd , 1

Selection.MoveStart , 1

Loop Until Selection.Font.Italic = False

Selection.MoveLeft , 1

Selection.Start = startHere

Selection.Font.Italic = False

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub ItalicisePhrase()**

' Paul Beverley - Version 04.10.18

' Selects text up to next punctuation mark and makes it italic

andThePunctuation = False

includeParens = True

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

OKchars = "A-Za-z '0-9\-" & ChrW(8217)

If includeParens = True Then OKchars = OKchars & "\(\)"

' The 8217 is the unicode for a curly apostrophe

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & OKchars & "]{1,}"

.MatchWildcards = True

.Execute

End With

If andThePunctuation = True Then Selection.MoveEnd , 1

Selection.Font.Italic = True

Selection.Collapse wdCollapseEnd

End Sub

**Sub** **ItalicBinomial()**

' Paul Beverley - Version 25.10.21

' Italicises binomials

jumpNext = True

noItalic = "subspecies subsp ssp var "

Selection.Expand wdWord

Selection.Font.Italic = True

Selection.MoveRight 1, wdWord

Selection.Expand wdWord

If InStr(Selection, ".") > 0 Then

Selection.MoveRight 2

Selection.Expand wdWord

End If

Selection.Expand wdWord

Selection.Font.Italic = True

Selection.MoveRight 1, wdWord

Selection.Expand wdWord

If InStr(noItalic, Replace(Trim(Selection), ChrW(160), "")) = 0 Then

Selection.Collapse wdCollapseStart

Exit Sub

End If

Selection.Font.Italic = False

Selection.MoveRight 1, wdWord

Selection.Expand wdWord

If InStr(Selection, ".") > 0 Then

Selection.MoveRight 1, wdWord

Selection.Expand wdWord

End If

Selection.Font.Italic = True

Selection.Collapse wdCollapseEnd

If jumpNext = True Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z]"

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute

End With

End If

Selection.Collapse wdCollapseStart

End Sub

**Sub PunctuationItalicOff()**

' Paul Beverley - Version 20.01.23

' Un-italicises all commas etc not followed by italic text

' stayItalicColour = wdNoHighlight

stayItalicColour = wdYellow

changeToRomanColour = wdGray25

trackIt = False

myCount = 0

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[,:;.""'\)]"

.Font.Italic = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.Start = rng.End

rng.End = rng.End + 1

If rng.Text = " " Then

rng.Start = rng.End

rng.End = rng.End + 1

End If

makeRoman = (rng.Font.Italic = False)

rng.Start = endNow - 1

rng.End = endNow

If makeRoman Then

rng.Font.Italic = False

rng.HighlightColorIndex = changeToRomanColour

myCount = myCount + 1

Else

If stayItalicColour > 0 Then rng.HighlightColorIndex = stayItalicColour

End If

rng.Start = endNow

rng.End = endNow

rng.Find.Execute

Loop

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[,:;.""'\)]"

.Font.Italic = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.Start = rng.End

rng.End = rng.End + 1

If rng.Text = " " Then

rng.Start = rng.End

rng.End = rng.End + 1

End If

makeRoman = (rng.Font.Italic = False)

rng.Start = endNow - 1

rng.End = endNow

If makeRoman Then

rng.Font.Italic = False

rng.HighlightColorIndex = changeToRomanColour

myCount = myCount + 1

Else

If stayItalicColour > 0 Then rng.HighlightColorIndex = stayItalicColour

End If

rng.Start = endNow

rng.End = endNow

rng.Find.Execute

Loop

End If

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[,:;.""'\)]"

.Font.Italic = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.Start = rng.End

rng.End = rng.End + 1

If rng.Text = " " Then

rng.Start = rng.End

rng.End = rng.End + 1

End If

makeRoman = (rng.Font.Italic = False)

rng.Start = endNow - 1

rng.End = endNow

If makeRoman Then

rng.Font.Italic = False

rng.HighlightColorIndex = changeToRomanColour

myCount = myCount + 1

Else

If stayItalicColour > 0 Then rng.HighlightColorIndex = stayItalicColour

End If

rng.Start = endNow

rng.End = endNow

rng.Find.Execute

Loop

End If

MsgBox "Changed italic to roman: " & myCount

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub PunctuationBoldOff()**

' Paul Beverley - Version 06.10.21

' Un-bolds all commas, etc. not followed by bold text

stayBoldColour = wdYellow

' stayBoldColour = wdNoHighlight

changeToRomanColour = wdYellow

trackIt = False

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

myCount = 0

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[,:;.]"

.Font.Bold = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.Start = rng.End

rng.End = rng.End + 1

If rng.Text = " " Then

rng.Start = rng.End

rng.End = rng.End + 1

End If

makeRoman = (rng.Font.Bold = False)

rng.Start = endNow - 1

rng.End = endNow

If makeRoman Then

rng.Font.Bold = False

rng.HighlightColorIndex = changeToRomanColour

myCount = myCount + 1

Else

If stayBoldColour > 0 Then rng.HighlightColorIndex = stayBoldColour

End If

rng.Start = endNow

rng.End = endNow

rng.Find.Execute

Loop

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[,:;.]"

.Font.Bold = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.Start = rng.End

rng.End = rng.End + 1

If rng.Text = " " Then

rng.Start = rng.End

rng.End = rng.End + 1

End If

makeRoman = (rng.Font.Bold = False)

rng.Start = endNow - 1

rng.End = endNow

If makeRoman Then

rng.Font.Bold = False

rng.HighlightColorIndex = changeToRomanColour

myCount = myCount + 1

Else

If stayBoldColour > 0 Then rng.HighlightColorIndex = stayBoldColour

End If

rng.Start = endNow

rng.End = endNow

rng.Find.Execute

Loop

End If

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[,:;.]"

.Font.Bold = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.Start = rng.End

rng.End = rng.End + 1

If rng.Text = " " Then

rng.Start = rng.End

rng.End = rng.End + 1

End If

makeRoman = (rng.Font.Bold = False)

rng.Start = endNow - 1

rng.End = endNow

If makeRoman Then

rng.Font.Bold = False

rng.HighlightColorIndex = changeToRomanColour

myCount = myCount + 1

Else

If stayBoldColour > 0 Then rng.HighlightColorIndex = stayBoldColour

End If

rng.Start = endNow

rng.End = endNow

rng.Find.Execute

Loop

End If

MsgBox "Changed bold to roman: " & myCount

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub UnderlineOnlyItalic()**

' Paul Beverley - Version 08.06.17

' Removes all underlining, then underlines all italic text

' If a selection is made, it just removes all underline

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = ActiveDocument.Content

rng.Font.Underline = False

If Selection.Start = Selection.End Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Underline = True

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub SuperscriptSwitch()**

' Paul Beverley - Version 17.05.22

' Toggle subscript

If Not Selection.Font.Subscript And \_

Not Selection.Font.Superscript Then

Selection.Font.Superscript = True

Exit Sub

End If

If Selection.Font.Superscript Then

Selection.Font.Subscript = True

Exit Sub

End If

If Selection.Font.Subscript = True Then \_

Selection.Font.Subscript = False

End Sub

**Sub SubscriptSwitch()**

' Paul Beverley - Version 17.05.22

' Toggle superscript and subscript

If Not Selection.Font.Subscript And \_

Not Selection.Font.Superscript Then

Selection.Font.Subscript = True

Exit Sub

End If

If Selection.Font.Subscript Then

Selection.Font.Superscript = True

Exit Sub

End If

If Selection.Font.Superscript = True Then \_

Selection.Font.Superscript = False

End Sub

**Sub ItalicQuoteToggle()**

' Paul Beverley - Version 13.07.20

' Toggles between italic and single or double quote

useSingle = True

' Make it useSingle = False for double quotes

hereNow = Selection.Start

If useSingle = True Then

openQ = 8216

closeQ = 8217

qt = "single"

Else

openQ = 8220

closeQ = 8221

qt = "double"

End If

If Selection.Font.Italic = True Then GoTo AddQuote

gotStart = False

i = 0

Do

Selection.MoveLeft Unit:=wdWord, Count:=1

Selection.MoveStart Count:=-1

If AscW(Selection) = openQ Then

Selection.Delete

gotStart = True

End If

DoEvents

i = i + 1

If i > 100 Then

Beep

MsgBox ("Sorry, I can't find an open " & qt & " quote.")

Exit Sub

End If

Loop Until gotStart = True

startItal = Selection.Start

Do

Selection.MoveRight Unit:=wdCharacter, Count:=1

DoEvents

Loop Until AscW(Selection) = closeQ

Selection.Delete Unit:=wdCharacter, Count:=1

Selection.Start = startItal

Selection.Font.Italic = True

Selection.Start = hereNow - 1

Selection.End = Selection.Start

Exit Sub

AddQuote:

myStart = 0

Do

Selection.MoveLeft 1

If Selection.Font.Italic = False Then

myStart = Selection.Start

Selection.TypeText ChrW(openQ)

End If

DoEvents

Loop Until myStart > 0

Selection.Start = hereNow + 1

myEnd = 0

Do

Selection.MoveRight 1

DoEvents

If Selection.Font.Italic = False Then

Selection.MoveLeft 1

myEnd = Selection.Start

Selection.MoveStart Count:=-1

If Asc(Selection) = 32 Then

Selection.MoveEnd Count:=-1

myEnd = myEnd + 1

Else

Selection.MoveStart Count:=1

End If

Selection.TypeText ChrW(closeQ)

Selection.End = myEnd + 1

Selection.Start = myStart

Selection.Font.Italic = False

End If

DoEvents

Loop Until myEnd > 0

Selection.Start = hereNow + 1

Selection.End = Selection.Start

End Sub

**Sub ApplyAttribute()**

' Paul Beverley - Version 19.10.13

' Apply attribute to current cell, row, column, paragraph, sentence, word or selection

' If the cursor is inside a table, select the cell (or column or row)

If Selection.Information(wdWithInTable) = True Then

myCol = Selection.Information(wdStartOfRangeColumnNumber)

myRow = Selection.Information(wdStartOfRangeRowNumber)

' Select current cell

Selection.Tables(1).Columns(myCol).Cells(myRow).Range.Select

' ... or column

' Selection.Tables(1).Columns(myCol).Select

' ... or row

' Selection.Tables(1).Rows(myRow).Select

End If

' If no text is selected, select the paragraph (or sentence or word)

If Selection.End - Selection.Start = 0 Then

Selection.Paragraphs(1).Range.Select

' or Selection.Sentences(1).Select

' or Selection.Words(1).Select

End If

' Apply attribute

Selection.Font.Underline = wdUnderlineDouble

' If in a table, replace the cursor where it was

If Selection.Information(wdWithInTable) = True Then

Selection.Tables(1).Columns(myCol).Cells(myRow).Range.Select

End If

Selection.Collapse wdCollapseStart

End Sub

**Sub SelectHighlightedText()**

**'** Paul Beverley - Version 05.09.11

' Selects all text in a given highlight

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

' Find beginning to the nearest word

nowCol = Selection.Range.HighlightColorIndex

wasEnd = Selection.End

Do

Selection.MoveStart wdWord, -1

Selection.MoveEnd wdWord, -1

Loop Until Selection.Range.HighlightColorIndex <> nowCol

Selection.MoveStart , -1

' Find beginning to the nearest character

Do

Selection.MoveStart , 1

Selection.MoveEnd , 1

Loop Until Selection.Range.HighlightColorIndex = nowCol

startHere = Selection.Start

' Find end to the nearest word

Selection.Start = wasEnd

Selection.End = wasEnd

Do

Selection.MoveEnd wdWord, 1

Selection.MoveStart wdWord, 1

Loop Until Selection.Range.HighlightColorIndex <> nowCol

Selection.MoveEnd , 1

' Find end to the nearest character

Do

Selection.MoveStart , -1

Selection.MoveEnd , -1

Loop Until Selection.Range.HighlightColorIndex = nowCol

Selection.Start = startHere

' Now do to the highlighted text whatever you want

' For example...

'Selection.Range.HighlightColorIndex = wdNoHighlight

'Selection.Font.Italic = wdToggle

'Selection.Start = Selection.End

ActiveDocument.TrackRevisions = nowTrack

End Sub

**Sub PasteUnformatted()**

**'** Paul Beverley - Version 30.01.10

' Pastes unformatted

' Selection.PasteAndFormat (wdFormatPlainText)

Selection.PasteSpecial DataType:=wdPasteText

End Sub

**Sub PasteWithEmphasis()**

**'** Paul Beverley - Version 30.12.10

' Paste with emphasis

Selection.PasteAndFormat (wdFormatSurroundingFormattingWithEmphasis)

End Sub

**Sub ClipToText()**

**'** Paul Beverley - Version 04.01.10

' Get pure text from pdfs and websites

Documents.Add

Selection.Paste

Selection.WholeStory

Selection.Cut

' Selection.PasteAndFormat (wdFormatPlainText)

Selection.PasteSpecial DataType:=wdPasteText

Selection.WholeStory

Selection.Copy

ActiveDocument.Close SaveChanges:=False

Selection.Paste

End Sub

**Sub ClipToTextWithEmphasis()**

**'** Paul Beverley - Version 23.02.11

' Get pure text from pdfs and websites

Documents.Add

Selection.Paste

Selection.WholeStory

Selection.Cut

Selection.PasteAndFormat (wdFormatSurroundingFormattingWithEmphasis)

Selection.WholeStory

Selection.Copy

ActiveDocument.Close SaveChanges:=False

Selection.Paste

End Sub

**Sub StyleCopy()**

**'** Paul Beverley - Version 04.01.11

' Copy style

Selection.CopyFormat

End Sub

**Sub StylePaste()**

**'** Paul Beverley - Version 04.01.11

' Paste style

Selection.PasteFormat

End Sub

**Sub UnifyFormatBackwards()**

' Paul Beverley - Version 27.11.20

' Makes start of para (or selection) same format as the end

doTrack = False

myTrack = ActiveDocument.TrackRevisions

startHere = Selection.Start

If doTrack = False Then ActiveDocument.TrackRevisions = False

If Selection.Start = Selection.End Then

Selection.Expand wdParagraph

Selection.End = startHere

End If

startHere = Selection.Start

Selection.Start = Selection.End - 1

Selection.Collapse wdCollapseEnd

Selection.MoveEnd , -1

If Selection = " " Or Selection = vbCr Then

Selection.MoveStart , -1

Selection.MoveEnd , -1

End If

Selection.MoveEnd , 1

Selection.CopyFormat

Selection.Start = startHere

Selection.PasteFormat

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub UnifyFormatForwards()**

' Paul Beverley - Version 27.11.20

' Make end of para (or selection) same format as the start

doTrack = False

myTrack = ActiveDocument.TrackRevisions

startHere = Selection.Start

If doTrack = False Then ActiveDocument.TrackRevisions = False

If Selection.Start = Selection.End Then

Selection.Expand wdParagraph

Selection.Start = startHere

End If

endHere = Selection.End

Selection.Collapse wdCollapseStart

Selection.MoveStart , 1

If Selection = " " Or Selection = vbCr Then

Selection.MoveStart , 1

Selection.MoveEnd , 1

End If

Selection.MoveEnd , -1

Selection.CopyFormat

Selection.Start = startHere

Selection.End = endHere

Selection.PasteFormat

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub SpikeCopy()**

**'** Paul Beverley - Version 26.02.11

' Copy to spike

NormalTemplate.AutoTextEntries.AppendToSpike Range:=Selection.Range

WordBasic.EditUnDo

End Sub

**Sub SpikeCut()**

**'** Paul Beverley - Version 26.02.11

' Cut to spike

NormalTemplate.AutoTextEntries.AppendToSpike Range:=Selection.Range

End Sub

**Sub SpikePaste()**

**'** Paul Beverley - Version 26.02.11

' Paste the whole contents of the spike

gotOne = False

For Each myEntry In NormalTemplate.AutoTextEntries

If myEntry.Name = "Spike" Then gotOne = True

Next myEntry

If gotOne = True Then

With NormalTemplate.AutoTextEntries("Spike")

.Insert Where:=Selection.Range, RichText:=True

.Delete

End With

End If

End Sub

**Sub TagA()**

' Paul Beverley - Version 29.10.11

' Adds a tag to the current paragraph

newLine = vbCr

startText = "<A>8.zxzx."

startText = "<A>"

'endText = "<ni>"

'endText = "</A>" & newLine

'endText = "</A>"

endTextOnSameLine = False

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Selection.TypeText startText

' Add close tag, if there is one

If endText > "" Then

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

' Tag on this line or next?

If endTextOnSameLine = True Then

Selection.MoveLeft Unit:=wdCharacter, Count:=1

End If

If Asc(Selection) = 13 Then

Selection.MoveRight , 1

Selection.MoveStart , -1

Selection.Delete

End If

Selection.TypeText endText

Selection.MoveEnd , -6

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub TagDQ()**

' Paul Beverley - Version 03.10.17

' Add a tag to the current paragraph

startText = "<DQ>"

'startText = "<DEF>"

endText = ""

' endText = "</C>" & vbCr

endText = "</DQ>"

'endText = "</DEF>"

endText = "</DQ>"

endTextOnSameLine = True

takeOffItalic = False

displayedQuoteStyle = "Displayed Quote"

changeFormat = False

myQuotes = """'" & ChrW(8216) & ChrW(8217) & ChrW(8220) & ChrW(8221)

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

' Go to start of the paragraph, delete quote and add tag

Selection.Expand wdParagraph

If takeOffItalic = True Then Selection.Font.Italic = False

If changeFormat = True Then

Selection.Style = ActiveDocument.Styles(displayedQuoteStyle)

End If

' Remove quote marks

Selection.Collapse wdCollapseStart

Selection.MoveEnd , 1

If InStr(myQuotes, Selection) > 0 Then Selection.Delete

Selection.InsertBefore Text:=startText

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.MoveStart , -2

Selection.MoveEnd , -1

If InStr(myQuotes, Selection) > 0 Then Selection.Delete

' Add close tag, if there is one

' Tag on this line or next?

If endText > "" Then

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

If endTextOnSameLine = True Then

Selection.MoveLeft Unit:=wdCharacter, Count:=1

End If

Selection.TypeText endText

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub TagSelectedOrItalic()**

' Paul Beverley - Version 13.04.16

' Add red tags to the selected text or italic text

tagTextSelected = "<IEQ></IEQ>"

tagTextItalic = "<i></i>"

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If Selection.Start = Selection.End Then

tagStop = InStr(tagTextItalic, "><")

startTag = Left(tagTextItalic, tagStop)

endTag = Mid(tagTextItalic, tagStop + 1)

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Else

startTag = Left(tagTextSelected, InStr(tagTextSelected, "><"))

endTag = Mid(tagTextSelected, InStr(tagTextSelected, "><") + 1)

End If

startNow = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=endTag

endNow = Selection.End

Selection.MoveStart , -(Len(endTag))

' Here’s where the colour is added

Selection.Font.Color = wdColorRed

Selection.End = startNow

Selection.TypeText Text:=startTag

Selection.MoveStart , -(Len(startTag))

' And here

Selection.Font.Color = wdColorRed

Selection.Start = endNow + Len(startTag)

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub TagList()**

' Paul Beverley - Version 19.01.13

' Add tags to a numbered or bulleted list

newLine = vbCrLf

startTextBullet = "<BL>"

' endTextBullet = "</BL>" & newLine

endTextBullet = "</BL>"

startTextNum = "<NL>"

' endTextNum = "</NL>" & newLine

endTextNum = "</NL>"

startTextLttr = "<Box>"

'startTextLttr = "<LL>"

' endTextLttr = "</LL>" & newLine

'endTextLttr = "</LL>"

endTextLttr = "</Box>"

endTextOnSameLine = True

numList = False

bulletList = False

lttrList = False

startText = startTextBullet

endText = endTextBullet

myStyle = ""

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

localFont = Selection.Font.Name

' Go to start of this line and add the tag

Selection.Paragraphs(1).Range.Select

i = Asc(Selection)

If i > 47 And i < 59 Then numList = True

If i = 149 Then bulletList = True

thisStyle = Selection.Range.Style

If InStr(thisStyle, "umber") > 0 Then numList = True: myStyle = thisStyle

If InStr(thisStyle, "ullet") > 0 Then bulletList = True: myStyle = thisStyle

myListString = Selection.FormattedText.ListFormat.ListString

If myListString > "" Then

If UCase(myListString) <> LCase(myListString) Then

lttrList = True

Else

numList = True

End If

If Selection.FormattedText.ListFormat.ListValue > 3 Then: numList = False

End If

Selection.End = Selection.Start + 1

myFont = ""

thisFont = Selection.Font.Name

If thisFont = "Symbol" Or thisFont = "Wingdings" Then

bulletList = True

myFont = thisFont

End If

If (numList = False And bulletList = False) Then

lttrList = True

startText = startTextLttr

endText = endTextLttr

End If

If (numList = True And bulletList = True) Then

myResponse = MsgBox("Bullets?", vbQuestion + vbYesNo)

If myResponse = vbNo Then

numList = True: bulletList = False

Else

numList = False: bulletList = True

End If

End If

If numList = True Then

startText = startTextNum

endText = endTextNum

Else

End If

Selection.InsertBefore Text:=startText

Selection.MoveEnd wdCharacter, -1

Selection.Font.Name = localFont

Selection.Start = Selection.End

' Find the final item

Selection.Paragraphs(1).Range.Select

myListValue = Selection.FormattedText.ListFormat.ListValue

Selection.Start = Selection.End

If myStyle > "" Then

Do

Selection.Paragraphs(1).Range.Select

thisStyle = Selection.Style

Selection.Start = Selection.End

Loop Until thisStyle <> myStyle

Else

If numList = True Then

If myListString = "" Then

Do

Selection.Paragraphs(1).Range.Select

i = Asc(Selection)

Selection.Start = Selection.End

Loop Until i < 48 Or i > 58

Else

Do

Selection.Paragraphs(1).Range.Select

thisListString = Selection.FormattedText.ListFormat.ListString

Selection.Start = Selection.End

Loop Until thisListString = ""

End If

Else

' It's either a bullet list or a letter list

If bulletList = True Then

If myFont > "" Then

' Find the end by a font change

Do

Selection.End = Selection.Start + 1

thisFont = Selection.Font.Name

Selection.Paragraphs(1).Range.Select

Selection.Start = Selection.End

Loop Until thisFont <> myFont

Else

' Find the end by bullet ASCII codes

Do

Selection.Paragraphs(1).Range.Select

i = Asc(Selection)

Selection.Start = Selection.End

Loop Until i <> 149

End If

Else

' It's a lettered list

If myListString = "" Then

Do

Selection.End = Selection.Start + 6

thisBit = Selection

Selection.Paragraphs(1).Range.Select

Selection.Start = Selection.End

Loop Until InStr(thisBit, Chr(9)) = 0

Else

Do

Selection.Paragraphs(1).Range.Select

thisListString = Selection.FormattedText.ListFormat.ListString

Selection.Start = Selection.End

Loop Until thisListString = ""

End If

End If

End If

End If

Selection.MoveUp Unit:=wdParagraph, Count:=1

Selection.Paragraphs(1).Range.Select

Selection.End = Selection.Start

' Tag on this line or next?

If endTextOnSameLine = True Then

Selection.MoveLeft Unit:=wdCharacter, Count:=1

End If

Selection.InsertBefore Text:=endText

Selection.Style = wdStyleNormal

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub FigTabBoxTagger()**

' Paul Beverley - Version 21.07.17

' Add tags to captions of all Figures, Tables and Boxes

myTag = "<Cap>"

myCaptionStartFig = "Figure"

addFullPointFig = True

myCaptionStartTab = "Table"

addFullPointTab = True

myCaptionStartBox = "Box"

addFullPointBox = True

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

checkForUnusedCaps = True

Application.ScreenUpdating = False

Dim myText(3) As String

Dim FPoint(3) As Boolean

myText(1) = myCaptionStartFig

myText(2) = myCaptionStartTab

myText(3) = myCaptionStartBox

FPoint(1) = addFullPointFig

FPoint(2) = addFullPointTab

FPoint(3) = addFullPointBox

For i = 1 To 3

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText(i)

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While Selection.Find.Found = True

isCaption = True

startFind = Selection.Start

endFind = Selection.End

Selection.HomeKey Unit:=wdLine

If Selection.Start <> startFind Then isCaption = False

If isCaption = True Then

Selection.Collapse wdCollapseStart

Selection.MoveEnd , -1

If Selection = " " Then isCaption = False

End If

Selection.Start = startFind

Selection.End = startFind

If isCaption Then

' If you've found a line beginning with Fig/Tab/Box

Selection.TypeText Text:=myTag

Selection.Expand wdParagraph

Selection.MoveEnd , -1

Selection.Start = Selection.End - 1

' Strip off any trailing space

If Selection = " " Then

Selection.Delete

Selection.MoveStart , -1

End If

' Add full point if wanted

If Selection <> "." And FPoint(i) = True Then

If Selection.Font.Superscript = False Then

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:="."

Selection.MoveStart , -1

Selection.range.HighlightColorIndex = wdGray25

End If

End If

End If

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.Find.Execute

Loop

Selection.HomeKey Unit:=wdStory

Next i

If checkForUnusedCaps = True Then

' Check if at least some of the first few captions are bold

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<Cap>Fig"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

clearUnusedCaps = False

For i = 1 To 6

If rng.Font.Bold = True Then clearUnusedCaps = True

rng.Find.Execute

rng.Collapse wdCollapseEnd

Next i

If clearUnusedCaps = True Then

' If the captions seem to be bold then remove any caption tags

' that are not bold

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<Cap>"

.Font.Bold = False

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

End If

End If

ActiveDocument.TrackRevisions = myTrack

Application.ScreenUpdating = True

Beep

End Sub

**Sub TagChecker()**

' Paul Beverley - Version 09.06.17

' Checks the continuity of a paired tag, e.g. <em>, </em>

offSymbol = "/"

Selection.Collapse wdCollapseStart

Set rng = Selection.range.Duplicate

Selection.Expand wdWord

Selection.MoveStart wdWord, -2

Selection.MoveEnd wdWord, 2

If InStr(Selection, "<") = 0 Or InStr(Selection, ">") = 0 Then

myResponse = MsgBox("Place the cursor inside the tag to be checked" \_

, , "TagChecker")

Exit Sub

End If

rng.Select

Selection.MoveEndUntil cset:=">", Count:=wdForward

Selection.MoveStartUntil cset:="<", Count:=wdBackward

Selection.MoveEnd , 1

isOff = ((InStr(Selection, offSymbol)) > 0)

If isOff Then Selection.MoveStart , 1

myFind = Selection

myStart = Selection.Start

wasStart = myStart

Set rng = Selection.range.Duplicate

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While rng.Find.Found = True

rng.Select

rng.Start = rng.Start - 1

myStart = rng.Start

If isOff = True Then

If (Asc(rng) = Asc(offSymbol)) Then

rng.Start = wasStart

rng.Select

Beep

Exit Sub

Else

isOff = False

End If

Else

If (Asc(rng) <> Asc(offSymbol)) Then

rng.Start = wasStart

rng.Select

Beep

Exit Sub

Else

isOff = True

End If

End If

rng.Collapse wdCollapseEnd

wasStart = myStart

' Go and find the next occurrence (if there is one)

rng.Find.Execute

Loop

Selection.EndKey Unit:=wdStory

Beep

Selection.Find.Text = myFind

End Sub

**Sub TagHighlighter()**

' Paul Beverley - Version 09.06.17

' Highlights all of the ranges a paired tag, e.g. <em>, </em>

myColour = wdBrightGreen

offSymbol = "/"

Selection.Collapse wdCollapseStart

Set rng = Selection.range.Duplicate

Selection.Expand wdWord

Selection.MoveStart wdWord, -2

Selection.MoveEnd wdWord, 2

If InStr(Selection, "<") = 0 Or InStr(Selection, ">") = 0 Then

myResponse = MsgBox("Place the cursor inside the tag to be checked" \_

, , "TagChecker")

Exit Sub

End If

rng.Select

Selection.MoveEndUntil cset:=">", Count:=wdForward

Selection.MoveStartUntil cset:="<", Count:=wdBackward

myTag = Selection

myTag = Replace(myTag, offSymbol, "")

Set myDoc = ActiveDocument

myDocName = myDoc.Name

Set rng = ActiveDocument.Content

rng.Copy

rng.End = 50

startBit = rng.Text

gottafile = False

For Each myWnd In Application.Windows

thisName = myWnd.Document.Name

Set rng = myWnd.Document.Content

rng.End = 50

If rng.Text = startBit And thisName <> myDocName Then

myWnd.Document.Activate

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = wdNoHighlight

gottafile = True

Exit For

End If

Next myWnd

If Not gottafile Then

Documents.Add

DoEvents

Selection.PasteSpecial DataType:=wdPasteText

Selection.HomeKey Unit:=wdStory

DoEvents

End If

myFind = "\<" & myTag & "\>\*\<" & offSymbol & myTag & "\>"

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<" & myTag & ">"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = False

.Execute

End With

Beep

End Sub

**Sub BorderParaOff()**

**'** Paul Beverley - Version 06.08.11

' Remove the borders applied to the paragraph

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.Borders(wdBorderTop).LineStyle = wdLineStyleNone

Selection.Borders(wdBorderLeft).LineStyle = wdLineStyleNone

Selection.Borders(wdBorderBottom).LineStyle = wdLineStyleNone

Selection.Borders(wdBorderRight).LineStyle = wdLineStyleNone

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub FandRapostrophe()**

**'** Paul Beverley - Version 22.04.11

' F&R text that contains curly quotes/apostrophe

Options.AutoFormatReplaceQuotes = False

Options.AutoFormatAsYouTypeReplaceQuotes = False

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^0145t Hooft"

.Replacement.Text = "^0146t Hooft"

.Wrap = wdFindContinue

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Options.AutoFormatReplaceQuotes = True

Options.AutoFormatAsYouTypeReplaceQuotes = True

End Sub

**Sub AccentPicker()**

**'** Paul Beverley - Version 10.05.11

' Call up the Insert Symbol window set to accented characters

Selection.TypeText Text:=Chr(224)

Selection.MoveStart wdCharacter, -1

Application.Run MacroName:="InsertSymbol"

End Sub

**Sub GreekPicker()**

' Paul Beverley - Version 20.06.14

' Call up the Insert Symbol window set to Greek characters

If Selection.Start = Selection.End Then

Selection.TypeText Text:=ChrW(937)

Selection.MoveStart wdCharacter, -1

Application.Run MacroName:="InsertSymbol"

Else

myLen = Len(Selection)

Selection.Collapse wdCollapseStart

Selection.TypeText Text:=ChrW(937)

Selection.MoveStart wdCharacter, -1

Selection.MoveRight , 1

Selection.MoveEnd , myLen

Selection.Delete

End If

End Sub

**Sub SciMarkPicker()**

**'** Paul Beverley - Version 10.05.11

' Call up the Insert Symbol window set to Greek characters

Selection.TypeText Text:=ChrW(8242)

Selection.MoveStart wdCharacter, -1

Application.Run MacroName:="InsertSymbol"

End Sub

**Sub ReferenceDateShift()**

' Paul Beverley - Version 14.10.16

' Move date from end of reference to after author(s)

myHLcolour = wdYellow

' myHLcolour = 0

myColour = wdColorBlue

' myColour = 0

allPrefs = " van der de den da le la vahl dos "

ignoreWords = " de den der di dos du la le ten van von and et al "

Do

Selection.Expand wdParagraph

If Len(Selection) < 10 Then

Beep

Exit Sub

End If

endPara = Selection.End

Selection.MoveEnd , -2

Selection.Start = Selection.End - 4

myDate = Val(Selection)

If myDate > 999 Then

Selection.MoveStart , -2

Selection.Delete

Selection.Expand wdParagraph

myRef = Selection

i = 0

Do

i = i + 1

myWd = Trim(Selection.range.Words(i).Text)

isLC = (LCase(myWd) = myWd) And (UCase(myWd) <> myWd)

If InStr(ignoreWords, " " & myWd & " ") Then isLC = False

Debug.Print Selection.range.Words(i).Text & "|"

Loop Until isLC = True

Selection.range.Words(i - 1).Select

Do While Asc(Selection) < 65 Or Asc(Selection) > 90

Selection.MoveLeft wdWord, 1

Loop

Selection.Collapse wdCollapseStart

Selection.TypeText Text:="(" & Trim(Str(myDate)) & ") "

Selection.Expand wdParagraph

Else

Selection.Expand wdParagraph

If myColour > 0 Then Selection.range.Font.Color = myColour

If myHLcolour > 0 Then Selection.range.HighlightColorIndex = myHLcolour

End If

Selection.Collapse wdCollapseEnd

Loop Until endPara = ActiveDocument.range.End

Selection.Start = Selection.End

End Sub

**Sub ReferenceNameFinder()**

**'** Paul Beverley - Version 08.04.11

' Get date from reference and add after author citation

myHLcolour = wdYellow

' myHLcolour = 0

' myColour = wdColorRed

myColour = 0

Do

Selection.Paragraphs(1).Range.Select

endPara = Selection.End

startPara = Selection.Start

myRef = Selection

myDatePos = InStr(myRef, "(")

myNamePos = InStr(myRef, ",") - 1

If myDatePos > 0 And myNamePos > -1 Then

Selection.End = startPara + myNamePos

myName = Selection

Selection.Start = startPara + myDatePos

Selection.End = Selection.Start + 4

myDate = Selection

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "<" & myName & ">"

.Replacement.Text = ""

.Execute

End With

If Selection.Find.Found And Selection.Start < startPara Then

startName = Selection.Start

endName = Selection.End

Selection.End = Selection.Start

Selection.MoveStart wdCharacter, -25

myBit = Selection

Selection.Start = endName

Selection.End = endName + 2

valNext = Val(Selection)

Selection.End = Selection.Start

If InStr(myBit, "(") > 0 Then

If valNext > 0 Then myDate = myDate & ":"

Else

myDate = "(" & myDate & ")"

End If

Selection.InsertAfter Text:=" " & myDate

Selection.Start = startName

If myColour > 0 Then Selection.Range.Font.Color = myColour

If myHLcolour > 0 Then Selection.Range.HighlightColorIndex = myHLcolour

End If

Else

Selection.Start = endPara

End If

Selection.Start = endPara + Len(myDate) + 1

Loop Until endPara = ActiveDocument.Range.End

Selection.Start = Selection.End

End Sub

**Sub ShortTitleLister()**

' Paul Beverley - Version 19.05.13

' Create a list of the named references in the notes

' myPattern1 = "<[A-Z][a-zA-Z]{1,} van [A-Z.]{1,}[ ,]"

' myPattern2 = "<[A-Z][a-zA-Z]{1,} [A-Z.]{1,}[ ,]"

' Smith J P[ ,] or Smith J. P.[ ,]

' myPattern = "<[A-Z][a-zA-Z]{1,}[ ,]{1,2}[A-Z. ]{1,}[ ,]"

' J P Smith[ ,] or J. P. Smith[ ,]

' myPattern = "<[A-Z. ]{1,} [A-Z][a-zA-Z]{1,}[ ,]"

' Smith, JP=[ ,] or Smith, J.P.[ ,]

' myPattern = "<[A-Z][a-zA-Z]{1,}[ ,]{1,2}[A-Z.]{1,}[ ,]"

' JP Smith[ ,] or J.P. Smith[ ,]

myPattern = "<[A-Z.]{1,} [A-Z][a-zA-Z]{1,}[ ,]"

myInitials = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

refLength = 40

myResponse = MsgBox("Test run first?", vbQuestion \_

+ vbYesNoCancel, "Short Reference Lister")

If myResponse = vbCancel Then Exit Sub

doTest = (myResponse = vbYes)

If ActiveDocument.Footnotes.Count > 0 Then

ActiveDocument.StoryRanges(wdFootnotesStory).Copy

Else

If ActiveDocument.Endnotes.Count = 0 Then

Beep

myResponse = MsgBox("Can't find any notes. Is this the correct file?", \_

vbQuestion + vbYesNoCancel, "Short Reference Lister")

Exit Sub

Else

ActiveDocument.StoryRanges(wdEndnotesStory).Copy

End If

End If

Documents.Add

Selection.Paste

myPattern1 = Replace(myPattern, "[A-Z][a-zA-Z]", "[derVDivan ]{2,8}[A-Z][a-zA-Z]")

' Highlight pattern1 for van/de/di etc

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myPattern1

.Highlight = False

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

rng.MoveStartWhile cset:=myInitials & " .", Count:=wdBackward

rng.MoveStart , 1

If Asc(rng.Text) = Asc(".") Then rng.MoveStart , 1

If Asc(rng.Text) = Asc(" ") Then rng.MoveStart , 1

rng.End = rng.End + refLength

dhfjk = rng.Font.Color

rng.Select

If rng.Font.Color = wdColorAutomatic Then

rng.HighlightColorIndex = wdGray25

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

' Main highlight pattern

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myPattern

.Highlight = False

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

rng.MoveStartWhile cset:=myInitials & " .", Count:=wdBackward

rng.MoveStart , 1

If Asc(rng.Text) = Asc(".") Then rng.MoveStart , 1

If Asc(rng.Text) = Asc(" ") Then rng.MoveStart , 1

rng.End = rng.End + refLength

crPos = InStr(rng.Text, vbCr)

If crPos > 0 Then rng.End = rng.Start + crPos - 1

If rng.Font.Color = wdColorAutomatic Then

rng.HighlightColorIndex = wdGray25

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

If doTest = True Then Exit Sub

' Split file at the beginning of each highlight

Set rng = ActiveDocument.Content

With rng.Find

.Wrap = wdFindContinue

.Text = ""

.Highlight = True

.Replacement.Text = "^p^&"

.Replacement.Highlight = False

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = Chr(2) & " "

.Replacement.Text = "zzzz"

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="zzzz!!!!" & vbCr

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = "zzzz" & "^p"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = "zzzz"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

ln = "~~~~~~~~~~~~~~~~~~~~"

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = "!!!!"

.Replacement.Text = "^p" & ln & ln & ln

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

With rng.ParagraphFormat

.LeftIndent = CentimetersToPoints(4)

.SpaceBeforeAuto = False

.SpaceAfterAuto = False

.SpaceBeforeAuto = False

.SpaceAfterAuto = False

.FirstLineIndent = CentimetersToPoints(-4)

End With

Selection.HomeKey Unit:=wdStory

Do While Selection = Chr(13)

Selection.MoveEnd , 1

Selection.Delete

Loop

Selection.HomeKey Unit:=wdStory

End Sub

**Sub AuthorForenamesInitialiser()**

**'** Paul Beverley - Version 20.12.23

' Changes author forenames to initials

' For numeric keypad

initialKey = "."

nextParaKey = "+"

stopKey = "0"

addFullPt = True

' If no numeric keypad, how about these?

' initialKey = "#"

' nextParaKey = "'"

' stopKey = "]"

Set rng = Selection.Range.Duplicate

myPosn = rng.Start

rng.Expand wdParagraph

For i = 1 To rng.Words.Count

Set wd = rng.Words(i)

If wd.Start > myPosn Then Exit For

Next i

iStart = i - 1

If iStart = 1 Then iStart = 2

Do

' For each reference

For i = iStart To rng.Words.Count

Set wd = rng.Words(i)

myInit = wd.Characters(1)

If addFullPt = True Then myInit = myInit & "."

If (LCase(wd) <> UCase(wd)) And Len(Trim(wd)) > 1 And \_

(UCase(myInit) = myInit) And (UCase(wd) <> wd) Then

wd.Select

If Right(wd, 1) = " " Then sp = " " Else sp = ""

nowWhat = InputBox("Initialise?", "AuthorForenamesInitialiser")

If nowWhat = initialKey Then

wd.Text = myInit & sp

DoEvents

End If

If nowWhat = nextParaKey Then Exit For

If nowWhat = stopKey Then Exit Sub

If InStr(initialKey & nextParaKey & stopKey, nowWhat) = 0 Then

Beep

myResponse = MsgBox("Have you read the instruction for this macro?!", \_

vbQuestion + vbYesNoCancel, "AuthorForenamesInitialiser")

If myResponse <> vbYes Then

MsgBox "See chapter 12 of my book at http://wordmacrotools.com"

Exit Sub

End If

End If

End If

Next i

posNow = wd.Start

Selection.Expand wdParagraph

Selection.Start = posNow

edPos = InStr(LCase(Selection), " ed")

If edPos = 0 Then edPos = InStr(LCase(Selection), "(ed")

If edPos > 0 Then

Set rng = Selection.Range.Duplicate

rng.Start = posNow + edPos

Beep

Else

Selection.Collapse wdCollapseEnd

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

End If

iStart = 2

Loop Until rng.End = ActiveDocument.Content.End

End Sub

**Sub AuthorsNotAllCaps()**

' Paul Beverley - Version 30.10.17

' Changes author surnames to initial cap only

' For numeric keypad

lowercaseKey = "."

nextParaKey = "+"

stopKey = "0"

' If no numeric keypad, how about these?

' lowercaseKey = "#"

' nextParaKey = "'"

' stopKey = "]"

Set rng = Selection.range.Duplicate

myPosn = rng.Start

rng.Expand wdParagraph

For i = 1 To rng.Words.Count

Set wd = rng.Words(i)

If wd.Start > myPosn Then Exit For

Next i

iStart = i - 1

Do

' for each reference

numWords = rng.Words.Count

For i = iStart To numWords

Set wd = rng.Words(i)

If (LCase(wd) <> UCase(wd)) And (UCase(wd) = wd) And \_

(LCase(wd) <> UCase(wd)) And Len(Trim(wd)) > 1 Then

wd.Select

myInit = wd.Characters(1)

nowWhat = InputBox("Lowercase?", "Authors Not All Caps")

If nowWhat = lowercaseKey Then wd.Text = myInit \_

& LCase(Mid(wd.Text, 2))

If nowWhat = nextParaKey Then Exit For

If nowWhat = stopKey Then Exit Sub

End If

Next i

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

iStart = 1

sfsdf = rng.End

dfgdfg = ActiveDocument.Content.End

xfvgdf = 0

Loop Until rng.End = ActiveDocument.Content.End

rng.Select

Beep

End Sub

**Sub AuthorCaseChange()**

' Paul Beverley - Version 16.01.21

' Lowercases author surnames (e.g. SMITH, J. to Smith, J.) in references list

' Look for e.g. "SMITH, J."

myFind = "([A-Z]{3,}), ([A-Z])."

' Look for e.g. "SMITH, A" or "SMITH, AB" or "SMITH, ABC" or

' myFind = "([A-Z]{3,}), ([A-Z]){1,3}>"

myColour = wdColorBlue

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While Selection.Find.Found = True

Selection.MoveStart , 1

Do While InStr(",", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Range.Case = wdLowerCase

Selection.Range.Font.Color = myColour

Selection.Collapse wdCollapseEnd

Selection.Find.Execute

Loop

End Sub

**Sub** **AuthorNameReinsert()**

' Paul Beverley - Version 05.09.20

' Replaces the dash-and-comma for the author's name

repeatText = "^=,"

If Selection.Start <> Selection.End Then

Set rng = Selection.Range.Duplicate

Else

Beep

myResponse = MsgBox("Work on the whole file?", \_

vbQuestion + vbYesNoCancel, "AcceptFormatting365")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = repeatText

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While rng.Find.Found = True

rng.End = rng.End - 3

rng.Expand wdParagraph

nameStart = rng.Start

commaPos = InStr(rng.Text, ",")

rng.Start = rng.Start + commaPos + 1

commaPos = InStr(rng.Text, ",")

rng.Start = rng.Start + commaPos

rng.Collapse wdCollapseStart

rng.Start = nameStart

rng.Copy

rng.Collapse wdCollapseStart

rng.Find.Execute

rng.Delete

rng.Paste

rng.Select

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

End Sub

**Sub AuthorNameReinsertParens()**

' Paul Beverley - Version 05.09.20

' Replaces the dash-and-comma for the author's name (parenthesis)

repeatText = "^=,"

If Selection.Start <> Selection.End Then

Set rng = Selection.Range.Duplicate

Else

Beep

myResponse = MsgBox("Work on the whole file?", \_

vbQuestion + vbYesNoCancel, "AcceptFormatting365")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = repeatText

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While rng.Find.Found = True

rng.End = rng.End - 3

rng.Expand wdParagraph

nameStart = rng.Start

parenPos = InStr(rng.Text, "(")

rng.Start = rng.Start + parenPos - 2

rng.Collapse wdCollapseStart

rng.Start = nameStart

rng.Copy

rng.Collapse wdCollapseStart

rng.Find.Execute

rng.Delete

rng.Paste

rng.Select

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

End Sub

**Sub AuthorDateFormatter()**

' Paul Beverley - Version 03.01.17

' Checks/corrects author/date formatting of reference list

' Is the list double-spaced?

listIsDoubleSpaced = True

listIsDoubleSpaced = False

' Is the date with the names? (i.e. for Vancouver date near end use False)

dateWithNames = True

addVancouverDate = False

' Create a separate list of all the changes?

createChangesList = True

' Highlight colour on query

myColour = wdTurquoise

' List of words likely to occur in multi-word surnames

okPrefix = "de De den Den der Der di Di dos du Du El " & \_

"la La le Le St ten Ten van Van von Von "

okPostfix = "jr Jr II III IV et al "

' Highlight references with more than so many authors

manyAuthorHighlight = True

manyAuthorColour = wdGray25

maxAuthorNumber = 8

' Choose your set of options

optionSet = 1

Select Case optionSet

Case 1 ' First set of options

' Spaced initials?

spaceInits = True

' Add a full point after each initial?

fullPtInits = True

' Comma after surname?

commaAfterSurname = True

' Use parentheses on the date?

parensOnDate = True

' Punctuation after the date?

punctuationAfterDate = ""

' punctuationAfterDate = "."

' punctuationAfterDate = ","

' Punctuation before the date?

punctuationB4Date = ""

' punctuationB4Date = "."

' punctuationB4Date = ","

' Initials before or after name on second and succeeding names?

initsBeforeName = False

' Use serial comma?

serialcomma = True

' Use 'and' with two authors? If so, which format?

' myFirstAnd = " and "

myFirstAnd = " & "

' myFirstAnd = " "

' Use 'and' with three or more authors? If so, which format?

' myFinalAnd = " and "

myFinalAnd = " & "

' myFinalAnd = " "

' Format for saying "Editors"

edText = "(ed.)"

edsText = "(eds)"

' If needed, et al.

etalText = ", et al. "

' Do you want et al in italic?

etalItalic = False

Case 2 ' Second set of options

' Spaced initials?

spaceInits = True

' Add a full point after each initial?

fullPtInits = True

' Comma after surname?

commaAfterSurname = True

' Use parentheses on the date?

parensOnDate = True

' Any punctuation after the date?

' punctuationAfterDate = ""

' punctuationAfterDate = "."

punctuationAfterDate = ","

' Any punctuation before the date?

punctuationB4Date = ""

'punctuationB4Date = "."

' punctuationB4Date = ","

' Initials before or after name on second and succeeding names?

initsBeforeName = True

' Use serial comma?

serialcomma = False

' Allow for the list to be double-spaced

listIsDoubleSpaced = False

' Use 'and' with two authors? If so, which format?

myFirstAnd = " and "

' myFirstAnd = " & "

' myFirstAnd = " "

' Use 'and' with three or more authors? If so, which format?

myFinalAnd = " and "

' myFinalAnd = " & "

' myFinalAnd = " "

' Format for saying "Editors"

edText = "(ed.)"

edsText = "(eds)"

' If needed, et al.

etalText = ", et al. "

' Do you want et al in italic?

etalItalic = True

End Select

' Start of main program

CR = vbCr: CR2 = CR & CR

For i = 1 To 30

spcs = " " & spcs

Next i

If serialcomma = True And myFinalAnd > " " Then

myFinalAnd = "," & myFinalAnd

End If

If createChangesList = True Then

gottaList = False

Set mainList = ActiveDocument

For Each myWnd In Application.Windows

Set myDoc = myWnd.Document

myDoc.Activate

Set rng = ActiveDocument.Content

rng.End = rng.Start + 15

If InStr(rng.Text, "Changes list") > 0 Then

gottaList = True

Exit For

End If

Next myWnd

If gottaList = False Then

Documents.Add

Selection.TypeText Text:="Changes list" & CR2

End If

Selection.TypeText Text:="------------------------------" & CR

Set changeList = ActiveDocument

mainList.Activate

End If

initsDelim = "": If fullPtInits = True Then initsDelim = "."

initsSpace = "": If spaceInits = True Then initsSpace = " "

maxAuNums = 30

Selection.Collapse wdCollapseStart

Selection.Expand wdParagraph

lenPara = Len(Selection)

hereNow = Selection.Start

Selection.EndKey Unit:=wdStory

Selection.MoveStart , -1

If Asc(Selection) <> 13 Then

Selection.Collapse wdCollapseEnd

Selection.TypeParagraph

End If

Selection.Start = hereNow

Selection.Collapse wdCollapseStart

Do While lenPara > 5

ReDim myWd(maxAuNums \* 2) As String

ReDim extraWord(maxAuNums \* 2) As String

ReDim auName(maxAuNums) As String

ReDim auInits(maxAuNums) As String

badData0 = False

badData1 = False

badData2 = False

badData3 = False

badData4 = False

badData5 = False

badData = False

theFinalAnd = myFinalAnd

gotEtAl = False

paraStart = Selection.Start

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[\(\)0-9]{4,}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

If Selection.Find.Found = False Then

Selection.MoveLeft 1

Selection.Expand wdParagraph

myResponse = MsgBox("No date." & CR2 & "Click 'Yes' to continue; " & \_

"'No' to highlight; 'Cancel' to exit.", \_

vbQuestion + vbYesNoCancel, "AuthorDateFormatter")

If myResponse = vbYes Then GoTo nextItem

If myResponse = vbNo Then

Selection.Expand wdParagraph

Selection.range.HighlightColorIndex = myColour

GoTo nextItem

End If

If myResponse = vbCancel Then GoTo cleanEnd

End If

'Put date into correct format

Selection.MoveEndWhile cset:=",.) abcdefg", Count:=wdForward

If Right(Selection, 1) = " " Then

Selection.MoveEnd , -1

Else

Set rng = ActiveDocument.Content

rng.End = Selection.End

rng.InsertAfter " "

End If

myOldDate = Selection

myDate = Replace(Trim(myOldDate), "(", "")

myDate = Replace(myDate, ")", "")

myDate = Replace(myDate, ",", "")

myDate = Replace(myDate, ".", "")

If parensOnDate = True Then myDate = "(" & myDate & ")"

myDate = " " & punctuationB4Date & myDate & punctuationAfterDate

' If no date

Selection.Start = paraStart

If InStr(Selection, Chr(13)) > 0 Then

Selection.Collapse wdCollapseStart

With Selection.Find

.Text = "\(\*\)"

.Execute

End With

' Put date into correct format

myDate = Selection

If InStr(Selection, Chr(13)) > 0 Then

Selection.Collapse wdCollapseStart

GoTo stopIT

End If

End If

Selection.Collapse wdCollapseEnd

Do While Asc(Selection) <> 32

Selection.MoveRight , 1

Loop

Selection.Start = paraStart

If dateWithNames = False Then

loopStop = False

Do

Selection.MoveStart wdWord, 1

myTestWd = Trim(Selection.Words(2))

If myTestWd = "-" Then myTestWd = Trim(Selection.Words(3))

If (LCase(myTestWd) = myTestWd) And (UCase(myTestWd) <> myTestWd) Then

loopStop = True

End If

If InStr(okPrefix, myTestWd) > 0 Then loopStop = False

Loop Until loopStop = True

Selection.Collapse wdCollapseStart

Selection.Start = paraStart

End If

' Move past the initial number (for Vancouver)

Do While Asc(Selection) < 65 Or Asc(Selection) = 91 Or Asc(Selection) = 93

Selection.MoveStart , 1

Loop

paraStart = Selection.Start

myText = Selection

originalText = myText

If InStr(myText, "et al") > 0 Then

theEtAlText = etalText

myText = Replace(myText, "et al. ", "")

myText = Replace(myText, "et al ", "")

theFinalAnd = ", "

Else

theEtAlText = ""

theFinalAnd = myFinalAnd

End If

' mask off of okPrefixes and okPostfixes

preWords = Split(okPrefix)

For i = 0 To UBound(preWords)

' check le/la/van/etc as the first name

If Trim(Selection.range.Words(1)) = preWords(i) Then

spPos = Len(preWords(i))

myText = Left(myText, spPos) & ChrW(124) & Mid(myText, spPos + 2)

End If

' check le/la/van/etc in succeeding names

myText = Replace(myText, " " & preWords(i) & " ", " " & preWords(i) & ChrW(124))

Next i

StatusBar = spcs & myText

DoEvents

postWords = Split(okPostfix)

For i = 0 To UBound(postWords)

myText = Replace(myText, " " & postWords(i) & " ", ChrW(124) & postWords(i) & " ")

myText = Replace(myText, " " & postWords(i) & ",", ChrW(124) & postWords(i) & ",")

Next i

' Check for "(editors)" etc

justNames = Replace(myText, myOldDate, "")

parenPosn = InStr(justNames, "(")

If parenPosn > 0 Then

parenText = Trim(Mid(justNames, parenPosn))

justNames = Replace(justNames, parenText, "")

If InStr(LCase(parenText), "ed") > 0 Then

If InStr(LCase(parenText), "s") > 0 Then

parenText = " " & edsText

Else

parenText = " " & edText

End If

End If

myDate = parenText & myDate

End If

' remove "and"

myText = Replace(justNames, " and ", ", ")

myText = Replace(myText, " & ", ", ")

' Sort out and switch the first surname and initials

i = 1

noSpace = False

Do

i = i + 1

x = Mid(myText, i, 1)

gotOne = (x = " ")

' check if you've fallen off the end

noSpace = (i > Len(myText))

Loop Until gotOne Or noSpace = True

If noSpace = True Then GoTo badDataHere

name1 = Trim(Left(myText, i - 1)) & " "

myText = Mid(myText, i + 1)

' Is this a name or initials

If UCase(name1) = name1 Then

' Initials before name

inits1 = name1

myText = Trim(myText) & " "

i = 1

Do

i = i + 1

x = Mid(myText, i, 1)

' Check for final lowercase

Loop Until UCase(x) = x

name1 = Trim(Left(myText, i)) & " "

Else

' Name first, then initials

i = 1

endInits = False

Do

i = i + 1

x = Mid(myText, i, 1)

' Check for a lowercase char in the inits

If LCase(x) = x And UCase(x) <> x Then badData5 = True

' Check for a comma or an and or

If x = "&" Or x = "," Or Mid(myText, i, 4) = " and" Then

endInits = True

Else

xy = Mid(myText, i, 3)

If UCase(xy) <> xy And LCase(xy) <> xy Then endInits = True

End If

Loop Until endInits = True Or i >= Len(myText)

inits1 = Trim(Left(myText, i)) & " "

End If

myText = inits1 & name1 & Trim(Mid(myText, i)) & " "

' count the number of spaces = number of "words"

myText = Replace(myText, ".-", "-")

myText = Replace(myText, ",", " ")

myText = Replace(myText, ".", " ")

myText = Replace(myText, " ", " ")

myText = Replace(myText, " ", " ")

myText = Replace(myText, " ", " ")

numWds = Len(myText) - Len(Replace(myText, " ", ""))

spPos = 0

isInst = True

j = 1

For i = 1 To numWds

myText = Mid(myText, spPos + 1)

spPos = InStr(myText, " ")

aBit = Left(myText, spPos - 1)

lenBit = Len(aBit)

If UCase(aBit) = aBit Then

isInst = False ' We've got at least one initial

For q = 1 To lenBit

myWd(j) = Mid(aBit, q, 1)

j = j + 1

Next q

Else

myWd(j) = aBit

j = j + 1

End If

Next i

numWds = j - 1

doCheckData = True

If isInst = True Then

myResponse = MsgBox("i can't work this one out, sorry" & CR2 \_

& "Click 'Yes' to continue; " & \_

"'No' to highlight; 'Cancel' to exit.", \_

vbQuestion + vbYesNoCancel, "AuthorDateFormatter")

Selection.Expand wdParagraph

If myResponse = vbNo Then

Selection.range.HighlightColorIndex = myColour

Selection.Collapse wdCollapseEnd

GoTo nextItem

End If

If myResponse = vbCancel Then GoTo cleanEnd

If myResponse = vbYes Then GoTo nextItem

End If

' Divide data into sets of initials or names

i = 1

j = 1

setOfInits = ""

Do While i <= numWds

If Len(myWd(i)) = 1 Then

setOfInits = setOfInits & myWd(i)

If myWd(i) <> "-" Then setOfInits = setOfInits & \_

initsDelim & initsSpace

Else

If setOfInits > "" Then

extraWord(j) = Replace(setOfInits, " -", "-")

j = j + 1

extraWord(j) = myWd(i)

j = j + 1

setOfInits = ""

Else

extraWord(j) = myWd(i)

j = j + 1

End If

End If

i = i + 1

Loop

If setOfInits > "" Then

extraWord(j) = Replace(setOfInits, " -", "-")

j = j + 1

End If

' Check for non-paired items

numWds = j - 1

If numWds / 2 > Int(numWds / 2) Then badData1 = True

' Check for "editors" without parens

If badData1 = True Then

If Left(LCase(extraWord(numWds)), 2) = "ed" Then

If InStr(extraWord(numWds), "s") Then

myDate = " " & edsText & myDate

Else

myDate = " " & edText & myDate

End If

badData1 = False

numWds = numWds - 1

End If

End If

' Put the names in one array and the initials in another

auInits(1) = Trim(extraWord(1))

auName(1) = extraWord(2)

auNum = 2

i = 3

mismatchedData = False

Do While i <= numWds

wordOne = extraWord(i)

wordTwo = extraWord(i + 1)

i = i + 2

initsOne = (UCase(wordOne) = wordOne)

initsTwo = (UCase(wordTwo) = wordTwo)

If ((initsOne = True And initsTwo = True) Or (initsOne = False \_

And initsTwo = False)) Then mismatchedData = True

If initsOne = True Then

auInits(auNum) = wordOne

auName(auNum) = wordTwo

Else

auInits(auNum) = wordTwo

auName(auNum) = wordOne

End If

auNum = auNum + 1

Loop

If mismatchedData = True Then badData3 = True

' Check for the word "ed(itors)" NOT in parentheses

totAuNum = auNum - 1

If badData3 = True Then

If Left(LCase(auName(auNum)), 2) = "ed" Then

totAuNum = auNum - 1

If InStr(auName(auNum), "s") > 0 Then

myDate = " " & edsText & myDate

Else

myDate = " " & edText & myDate

End If

badData3 = False

End If

End If

' Make up the new data as a string of names/inits

surnameSpacer = " "

If commaAfterSurname = True Then surnameSpacer = ", "

If UCase(auName(1)) = auName(1) Then

initsOne = auName(1)

auName(1) = auInits(1)

auInits(1) = initsOne

End If

newText = auName(1) & surnameSpacer & auInits(1)

auNum = 2

Do While auNum < totAuNum

If initsBeforeName = True Then

newText = newText & ", " & Trim(auInits(auNum)) & " " \_

& auName(auNum)

Else

newText = newText & ", " & auName(auNum) & surnameSpacer \_

& Trim(auInits(auNum))

End If

auNum = auNum + 1

Loop

' Add the second or final name and initials

Select Case totAuNum

Case 1 ' do nothing

Case 2:

If initsBeforeName = True Then

newText = newText & myFirstAnd & Trim(auInits(2)) & " " & auName(2)

Else

newText = newText & myFirstAnd & auName(2) & surnameSpacer & Trim(auInits(2))

End If

Case Else

' add final author at end of string

If initsBeforeName = True Then

newText = newText & theFinalAnd & Trim(auInits(totAuNum)) \_

& " " & auName(totAuNum)

Else

newText = newText & theFinalAnd & auName(totAuNum) & surnameSpacer \_

& Trim(auInits(totAuNum))

End If

End Select

If auInits(totAuNum) = "" Then badData4 = True

If dateWithNames = False And addVancouverDate = False Then myDate = " "

' Tidy up and add date

newText = Replace(newText, " ", " ") & theEtAlText & myDate

newText = Replace(newText, " ", " ")

newText = Replace(newText, " ,", ",")

newText = Replace(newText, "..", ".")

newText = Replace(newText, ChrW(124), " ")

badDataHere:

badData = noSpace Or badData0 Or badData1 Or badData2 Or \_

badData3 Or badData4 Or badData5

StatusBar = spcs & newText & "|" & spcs & originalText & "|"

DoEvents

If newText <> Trim(originalText) Then

If badData = False Then

Selection.TypeText Text:=newText

If createChangesList = True Then

changeList.Activate

start1 = Selection.Start

Selection.TypeText Text:=originalText & CR

end1 = Selection.Start

start2 = Selection.Start

Selection.TypeText Text:=newText & CR

end2 = Selection.Start

len1 = Len(originalText)

len2 = Len(newText)

maxCheck = len1

If len1 > len2 Then maxCheck = len2

For i = 1 To maxCheck

If Mid(originalText, i, 1) <> Mid(newText, i, 1) \_

Then Exit For

Next i

firstDiff = i

For i = 0 To maxCheck - 1

If Mid(originalText, len1 - i, 1) <> Mid(newText, len2 - i, 1) \_

Then Exit For

Next i

lastDiff = i

Selection.Start = start1 + firstDiff - 1

Selection.End = start2 - lastDiff - 1

Selection.range.HighlightColorIndex = myColour

Selection.Start = start2 + firstDiff - 1

Selection.End = end2 - lastDiff - 1

Selection.range.HighlightColorIndex = myColour

Selection.Start = end2

Selection.TypeParagraph

mainList.Activate

End If

End If

End If

If badData = True Then

Selection.Start = paraStart

myResponse = MsgBox("i can't work this one out, sorry." & CR2 \_

& "Click 'Yes' to continue; " & \_

"'No' to highlight; 'Cancel' to exit.", vbQuestion \_

+ vbYesNoCancel, "AuthorDateFormatter")

If myResponse = vbCancel Then GoTo cleanEnd

If myResponse = vbNo Then

Selection.Expand wdParagraph

Selection.range.HighlightColorIndex = myColour

End If

End If

Selection.Start = Selection.End - 15

etAlPos = InStr(Selection, "et al")

If etAlPos > 0 Then

Selection.MoveStart , etAlPos - 1

Selection.End = Selection.Start + 5

If etalItalic = True Then Selection.Font.Italic = True

etAlPos = 0

End If

Selection.Expand wdParagraph

If manyAuthorHighlight = True And totAuNum > maxAuthorNumber Then \_

Selection.range.HighlightColorIndex = manyAuthorColour

nextItem:

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

lenPara = Len(Selection)

If listIsDoubleSpaced = True Then

If lenPara > 5 Then GoTo stopIT

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

lenPara = Len(Selection)

End If

Loop

Beep

cleanEnd:

Selection.Collapse wdCollapseEnd

Exit Sub

stopIT:

If listIsDoubleSpaced = True Then

myResponse = MsgBox("i was expecting double-spaced references" \_

& CR2 & "Set listIsDoubleSpaced = False for single-spaced" \_

, , "AuthorDateFormatter")

Else

Beep

End If

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Selection.Collapse wdCollapseStart

End Sub

**Sub AuthorInitialCapitalReferences()**

' Paul Beverley - Version 16.01.21

' Changes author surnames in all capitals to initial capital

LCnames = "AL,EL"

stopToCheck = True

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

For Each myPara In rng.Paragraphs

For i = 1 To myPara.Range.Words.Count

Set wd = myPara.Range.Words(i).Duplicate

DoEvents

Do While InStr(" ", Right(wd.Text, 1)) > 0

wd.MoveEnd , -1

DoEvents

Loop

If Val(wd) > 1000 Then Exit For

If Len(wd) > 2 And wd <> "and" Then

wd.Text = Left(wd.Text, 1) & LCase(Mid(wd.Text, 2))

Else

If Len(wd) = 2 And InStr(LCnames, wd) > 0 Then

changeThisOne = True

If stopToCheck = True Then

wd.Select

myResponse = MsgBox("Change?", \_

vbQuestion + vbYesNoCancel, "AuthorInitialCapitalReferences")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then changeThisOne = False

End If

If changeThisOne = True Then

wd.Text = Left(wd.Text, 1) & LCase(Mid(wd.Text, 2))

End If

End If

End If

Debug.Print i & "|", wd & "|"

Next i

myPara.Range.Select

Next myPara

End Sub

**Sub EtAlElision()**

' Paul Beverley - Version 24.08.20

' Crops multi-authors to a given number before 'et al'

doJustOne = False

numAuthors = 3

delimiter = " ("

etalText = "et al."

etalItalic = True

Selection.Expand wdParagraph

Do

If Selection.Range.HighlightColorIndex > 0 Or doJustOne = True Then

endNames = Selection.Start + InStr(Selection, delimiter)

Selection.Collapse wdCollapseStart

i = 0

numNames = 0

numInitials = 0

Do

Selection.MoveEnd Unit:=wdWord, Count:=1

myWord = Selection

If LCase(myWord) <> UCase(myWord) Then

If UCase(myWord) <> myWord Then

numNames = numNames + 1

Else

numInitials = numInitials + 1

If numInitials > numNames + 1 Then numInitials = numNames + 1

End If

End If

DoEvents

Debug.Print myWord & " | " & numNames & " | " & numInitials

Selection.Collapse wdCollapseEnd

Loop Until numNames >= numAuthors And numInitials >= numAuthors

Selection.End = endNames

If Left(Selection, 1) = "." Then Selection.MoveStart , 1

If Right(Selection, 1) = " " Then Selection.MoveEnd , -1

Selection.TypeText " " & etalText

If etalItalic = True Then

Selection.MoveStart , -(Len(etalText))

Selection.Font.Italic = True

End If

Selection.Expand wdParagraph

End If

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

Loop Until Len(Selection) < 3 Or doJustOne = True

Selection.MoveRight , 1

Beep

End Sub

**Sub EtAlCitationElision()**

**'** Paul Beverley - Version 04.10.23

' Crops multi-author citation to single name + 'et al'

isItalic = False

etAlText = " et al., "

Selection.Expand wdWord

Selection.Collapse wdCollapseEnd

Selection.MoveEnd , 1

If Selection = "," Then Selection.Delete

Selection.MoveEnd Unit:=wdWord, Count:=10

myStart = Selection.Start

wdsMax = 10

For i = 1 To wdsMax

Debug.Print Selection.Words(i)

If InStr("12", Selection.Words(i).Characters(1)) > 0 Then Exit For

Next i

If i = wdsMax + 1 Then

WordBasic.EditUndo

Beep

MsgBox ("Can't find a date (year)")

Exit Sub

End If

Selection.Words(i).Select

Selection.Collapse wdCollapseStart

Selection.Start = myStart

Selection.TypeText Text:=etAlText

If isItalic Then

Selection.Start = Selection.Start - 8

Selection.End = Selection.End - 2

Selection.Font.Italic = True

Selection.Collapse wdCollapseEnd

End If

End Sub

**Sub HighlightMultiAuthorCitations()**

' Paul Beverley - Version 09.06.17

' Finds and highlights all the multi-author citations in the text

myColour = wdYellow

minDate = 1900

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\(\*\)"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

endNow = rng.End

Set rng2 = rng.Duplicate

numYrs = 0

For Each wd In rng2.Words

If Val(wd.Text) > minDate Then

numYrs = numYrs + 1

End If

Next wd

If numYrs > 0 Then

txt = rng.Text

numCommas = Len(txt) - Len(Replace(txt, ",", ""))

If numCommas > numYrs Then rng.HighlightColorIndex = myColour

End If

rng.Start = endNow

rng.End = endNow

rng.Find.Execute

Loop

End Sub

**Sub AuthorNameSwap()**

' Paul Beverley - Version 01.10.14

' Changes author surname and initials/given name

avoidWords = "the "

' create list of avoidWords

avoidWords = LCase(Trim(avoidWords)) & " "

numNoWds = Len(avoidWords) - Len(Replace(avoidWords, " ", ""))

ReDim notWords(numNoWds) As String

For i = 1 To numNoWds

spPos = InStr(avoidWords, " ")

notWords(i) = Left(avoidWords, spPos - 1) & " "

avoidWords = Mid(avoidWords, spPos + 1)

Next i

maxAuNums = 4

Selection.Collapse wdCollapseStart

Selection.Expand wdParagraph

lenPara = Len(Selection)

Do Until 0

If lenPara < 2 Then

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

lenPara = Len(Selection)

If lenPara < 2 Then

Selection.Collapse wdCollapseEnd

Beep

Exit Sub

End If

End If

ReDim myWd(maxAuNums \* 2) As String

ReDim extraWord(maxAuNums \* 2) As String

ReDim auName(maxAuNums) As String

ReDim auInits(maxAuNums) As String

paraStart = Selection.Start

With Selection.Find

.Text = "[.,\(]"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Selection.Collapse wdCollapseStart

Selection.Start = paraStart

tryThisOne = True

allNames = Selection.Text

If Selection.Font.Italic = True Then tryThisOne = False

If InStr(allNames, Chr(34)) > 0 Then tryThisOne = False

If InStr(allNames, "HYPERLINK") > 0 Then tryThisOne = False

If tryThisOne = True Then

maxWds = Selection.Words.Count

longWds = 0

For i = 1 To maxWds

myWd(i) = Selection.Words(i)

If Len(myWd(i)) > 2 Then longWds = longWds + 1

Next i

If maxWds < 5 And InStr(allNames, ",") > 0 Then

forename = myWd(maxWds)

surname = myWd(maxWds - 2)

Selection.Words(maxWds) = surname

Selection.Words(maxWds - 2) = forename

Selection.Words(maxWds - 1) = " "

End If

End If

nextItem:

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

lenPara = Len(Selection)

Loop

Beep

cleanEnd:

Selection.Collapse wdCollapseEnd

Exit Sub

stopIT:

If listIsDoubleSpaced = True Then

myResponse = MsgBox("I was expecting double-spaced references" \_

& CR2 & "Set listIsDoubleSpaced = False for single-spaced" \_

, , "AuthorDateFormatter")

Else

Beep

End If

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Selection.Collapse wdCollapseStart

End Sub

**Sub SwapNames()**

' Paul Beverley - Version 06.10.14

' Swap adjacent names

addComma = True

fpOnInitials = True

minLen = 50

startAgain:

startPoint = Selection.Start

Selection.Expand wdParagraph

If Len(Selection) < minLen Then GoTo cantDoIt

myText = Replace(Selection, ".", "")

myText = Replace(myText, ",", "")

wd = Split(myText, " ")

startHere = Selection.Start

If addComma = True Then w2 = w2 & ","

' This is for Fred Bloggs

If Len(wd(0)) > 1 And Len(wd(1)) > 1 Then

secondNameEnd = InStr(Selection, wd(1)) + Len(wd(1))

If addComma = True Then wd(1) = wd(1) & ","

Selection.End = startHere + secondNameEnd - 1

Selection.TypeText Text:=wd(1) & " " & wd(0)

GoTo getNext

End If

' This is for F. Bloggs

If Len(wd(0)) = 1 And Len(wd(1)) > 1 Then

secondNameEnd = InStr(Selection, wd(1)) + Len(wd(1))

If addComma = True Then wd(1) = wd(1) & ","

Selection.End = startHere + secondNameEnd - 1

If fpOnInitials = True Then

wd(0) = wd(0) & "."

End If

Selection.TypeText Text:=wd(1) & " " & wd(0)

GoTo getNext

End If

' If none of the above!

cantDoIt:

Beep

Exit Sub

getNext:

Selection.Expand wdParagraph

lenPara = Len(Selection)

Selection.Collapse wdCollapseEnd

endPoint = Selection.Start

If lenPara > minLen And endPoint <> startPoint Then GoTo startAgain

Beep

End Sub

**Sub YearMoveToEnd()**

' Paul Beverley - Version 16.02.22

' Moves the year to end of reference

doNumber = 5

patternCut = ", 1948."

patternPaste = ", 1948."

' patternCut = " (1948)."

' patternPaste = ", (1948)."

doFootnotes = Selection.Information(wdInFootnote)

doEndnotes = Selection.Information(wdInEndnote)

If doFootnotes Then Set rng = \_

ActiveDocument.StoryRanges(wdFootnotesStory)

If doEndnotes Then Set rng = \_

ActiveDocument.StoryRanges(wdEndnotesStory)

If Not (doFootnotes) And Not (doEndnotes) Then \_

Set rng = ActiveDocument.Content

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

myFind = Replace(patternCut, "1948", "^#^#^#^#")

For Each pa In rng.Paragraphs

Set rng2 = pa.Range

gotYear = False

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^#^#^#^#"

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

If rng2.Find.Found = True Then

doNumber = doNumber - 1

gotYear = True

myYear = rng2.Text

Set rng2 = pa.Range

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^#^#^#^#"

.Text = myFind

.Replacement.Text = ""

.Execute Replace:=wdReplaceOne

End With

Set rng2 = pa.Range

rng2.End = rng2.End - 1

myTest = Right(rng2, 1)

myReplace = Replace(patternPaste, "1948", myYear)

If Right(rng2, 1) = "." And Right(myReplace, 1) = "." Then

myReplace = Replace(myReplace, ".", "")

rng2.End = rng2.End - 1

End If

nowEnd = rng.End

rng2.InsertAfter Text:=myReplace

Set rng3 = rng2.Duplicate

rng3.End = rng2.End

rng3.Start = rng3.End - Len(myReplace)

rng3.Font.Italic = False

End If

DoEvents

If doNumber = 0 Then

rng3.Select

myText = InputBox("Do more?", "YearMoveToEnd")

doNumber = Val(myText)

If doNumber = 0 Then Beep: Exit Sub

End If

Next pa

End Sub

**Sub InitialSwapper()**

' Paul Beverley - Version 06.06.22

' Swaps initials and surname (P.E. Beverley -> Beverley, P.E.)

' Expand selection to single or whole words

myStart = Selection.Start - 1

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = myStart

Selection.MoveStart wdWord, -1

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z.^32\-]{1,}"

.Forward = False

.MatchWildcards = True

.Execute

DoEvents

End With

If Left(rng.Text, 1) = " " Then rng.MoveStart , 1

rng.Cut

Selection.Collapse wdCollapseEnd

Selection.MoveEnd , 1

If Selection = "," Then

Selection.Collapse wdCollapseEnd

Else

Selection.Collapse wdCollapseStart

Selection.TypeText Text:=","

End If

Selection.TypeText Text:=" "

Selection.Paste

Set rng = Selection.Range.Duplicate

rng.MoveStart , -1

rng.MoveEnd , 1

If rng.Text = " " Then rng.Text = " "

' Dummy find to reset the Find parameters

With rng.Find

.Text = " "

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute

DoEvents

End With

End Sub

**Sub InitialSwapperReverse()**

**'** Paul Beverley - Version 22.07.22

' Swaps initials and surname (Beverley, P.E. -> P.E. Beverley)

' Expand selection to single or whole words

myStart = Selection.Start - 1

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = myStart

Selection.MoveStart wdWord, -1

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z.,^32\-]{1,}"

.Forward = True

.MatchWildcards = True

.Execute

DoEvents

End With

rng.Select

If Right(rng.Text, 1) = " " Then rng.MoveEnd , -1

If Right(rng.Text, 1) = "," Then rng.MoveEnd , -1

rng.Cut

Selection.Collapse wdCollapseStart

myStart = Selection.Start

Selection.Paste

Selection.TypeText Text:=" "

Selection.Start = myStart

Selection.End = Selection.Start + 2

If Selection = ", " Then Selection.Delete

' Dummy find to reset the Find parameters

With rng.Find

.Text = " "

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute

DoEvents

End With

End Sub

**Sub ReferenceNameDateMover()**

**'** Paul Beverley - Version 10.04.23

' Moves forename/initials to before surname or move year to end

includePublisher = True

needAFullStop = True

needAFullStop = False

myExtraText = "City, State: Publisher, "

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseStart

Else

GoTo doYear

End If

If Val(rng) > 0 Then GoTo doYear

' Find the comma

Dim wd(10) As String

Selection.MoveEnd wdWord, 4

commaPos = InStr(Selection, ", ")

Selection.MoveStart , commaPos + 1

Selection.Collapse wdCollapseStart

Selection.MoveEnd wdWord, 6

For i = 1 To 6

wd(i) = Selection.Range.Words(i)

DoEvents

Next i

deleteFstop = False

For i = 1 To 6

Debug.Print wd(i - 1), wd(i)

If LCase(wd(i)) = UCase(wd(i)) And wd(i) <> ". " Then Exit For

If Len(wd(i - 1)) > 1 And wd(i) = ". " Then

deleteFstop = True

Exit For

End If

If wd(i) = ". " And Len(wd(i - 1)) > 1 And Right(wd(i - 1), 1) <> " " Then

i = i - 1

Exit For

End If

DoEvents

Next i

If i = 7 Then

Beep

MsgBox ("Can't find where the name ends, sorry")

Exit Sub

End If

Selection.Collapse wdCollapseStart

Selection.MoveEnd wdWord, i - 1

needSpace = (Right(Selection, 1) <> " ")

Selection.Cut

If deleteFstop = True Then

Selection.MoveEnd , 2

Selection.Delete

End If

rng.Select

Selection.Collapse wdCollapseStart

Selection.Paste

If needSpace Then Selection.TypeText Text:=" "

rng.Collapse wdCollapseEnd

rng.MoveEnd , 3

If Right(rng, 3) = ", ," Then rng.Text = ","

Exit Sub

doYear:

If Selection.Start = Selection.End Then

yrText = rng.Text

' Delete the year text (and full stop)

rng.Start = rng.Start - 2

rng.End = rng.End + 2

If Right(rng.Text, 1) <> "." Then rng.MoveEnd , -1

rng.Delete

rng.MoveStart , -1

rng.MoveEnd , 1

If rng.Text = ",," Then rng.Text = ","

' Find the end of the para

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

rng.MoveStart , -2

rng.MoveEnd , -1

' Add a full stop if necessary

If needAFullStop Then

If rng.Text <> "." Then

rng.Collapse wdCollapseEnd

rng.InsertAfter Text:="."

End If

Else

If rng.Text = "." Then rng.Delete

End If

rng.Collapse wdCollapseStart

If includePublisher = True Then

Do

rng.MoveStart , -1

DoEvents

Loop Until InStr(",.", Left(rng.Text, 1)) > 0

rng.Collapse wdCollapseStart

rng.MoveEnd , 1

If rng.Text = "." Then

rng.Delete

rng.MoveStart , 1

Else

rng.MoveStart , 2

End If

rng.InsertBefore Text:="("

rng.Expand wdParagraph

rng.MoveEnd , -1

rng.Start = rng.End - 1

rng.Select

If rng.Text = "." Then

rng.InsertBefore Text:=", " & yrText & ")"

Else

rng.InsertAfter Text:=", " & yrText & ")"

End If

Else

rng.InsertBefore Text:=" (" & yrText & ")"

End If

rng.Collapse wdCollapseEnd

rng.Select

Else

Set rng = Selection.Range.Duplicate

rng.MoveEnd , -1

If InStr(rng.Text, " ") = 0 Then GoTo addCSP

' Is there a ")" at the end?

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

Do

rng.MoveStart , -1

DoEvents

Loop Until Asc(Left(rng.Text, 1)) > 47

rng.Collapse wdCollapseStart

rng.Expand wdWord

rng.Collapse wdCollapseEnd

rng.MoveEnd , 1

If rng.Text <> ")" Then rng.InsertAfter Text:=")"

' Add "(" at the beginning

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Expand wdWord

rng.Collapse wdCollapseStart

rng.InsertBefore Text:="("

' Remove "(" from middle

Set rng = Selection.Range.Duplicate

parenPos = InStr(rng, "(")

If parenPos > 0 Then

rng.MoveStart , parenPos - 1

rng.End = rng.Start + 1

rng.Delete

rng.MoveStart , -1

rng.InsertBefore Text:=","

End If

End If

Selection.Collapse wdCollapseEnd

Exit Sub

addCSP:

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

Selection.TypeText Text:=myExtraText

End Sub

**Sub AlphabeticOrderChecker()**

' Paul Beverley - Version 22.10.16

' Find any suspicious non-alphabetism

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

firstWordOne = Selection.Words(1)

Do

Selection.MoveEnd Unit:=wdParagraph, Count:=1

' Stop if the second line is blank

If Len(Selection.range.Paragraphs(2).range.Text) < 2 Then

Beep

Selection.Collapse wdCollapseEnd

Selection.MoveRight , 1

Exit Sub

End If

firstWordTwo = Selection.range.Paragraphs(2).range.Words(1)

myLine1 = LCase(Selection.range.Paragraphs(1).range.Text)

myLine2 = LCase(Selection.range.Paragraphs(2).range.Text)

myLine1 = Replace(myLine1, "-", " ")

myLine1 = Replace(myLine1, ChrW(8216), "")

myLine1 = Replace(myLine1, "'", "")

myLine1 = Replace(myLine1, ",", "")

myLine2 = Replace(myLine2, "-", " ")

myLine2 = Replace(myLine2, ChrW(8216), "")

myLine2 = Replace(myLine2, "'", "")

myLine2 = Replace(myLine2, ",", "")

' Check the alphabetism

If myLine1 > myLine2 And firstWordTwo <> firstWordOne Then

Selection.Collapse wdCollapseEnd

Selection.MoveDown Unit:=wdParagraph, Count:=1

Selection.MoveUp Unit:=wdParagraph, Count:=1

Selection.MoveStart Unit:=wdParagraph, Count:=-2

Exit Sub

End If

Selection.MoveStart Unit:=wdParagraph, Count:=1

firstWordOne = Selection.Words(1)

Loop Until Selection.End = ActiveDocument.Content.End

Selection.MoveLeft , 1

Beep

End Sub

**Sub AlphaOrderChecker()**

' Paul Beverley - Version 27.10.17

' Creates an alpha-sorted Paul Beverley - Version of selected text showing changes

myList = Selection

If Len(myList) < 10 Then

myResponse = MsgBox("Check whole file?", vbQuestion \_

+ vbYesNoCancel, "AlphaOrderChecker")

If myResponse <> vbYes Then Exit Sub

myList = ActiveDocument.Content.Text

End If

If Right(myList, 1) = vbCr Then \_

myList = Left(myList, Len(myList) - 1)

Documents.Add

Set firstDoc = ActiveDocument

Selection.TypeText Text:=Trim(myList)

Documents.Add

Set secondDoc = ActiveDocument

Selection.TypeText Text:=Trim(myList)

Set rng = ActiveDocument.Content

rng.Sort ExcludeHeader:=False, FieldNumber:="Paragraphs", \_

SortOrder:=wdSortOrderAscending, SortFieldType:=wdSortFieldAlphanumeric

' Now compare the sorted and unsorted copies

Application.CompareDocuments \_

OriginalDocument:=Documents(firstDoc.Name), \_

RevisedDocument:=Documents(secondDoc.Name), \_

Destination:=wdCompareDestinationRevised

firstDoc.Close SaveChanges:=False

secondDoc.Activate

End Sub

**Sub AlphabeticOrderByLine()**

' Paul Beverley - Version 14.08.18

' Finds any suspicious non-alphabetism

' checkSecondWord = False

checkSecondWord = True

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

firstWordOne = Selection.Words(1)

Do

Selection.MoveEnd Unit:=wdParagraph, Count:=1

' Stop if the second line is blank

If Len(Selection.range.Paragraphs(2).range.Text) < 2 Then

Beep

Selection.Collapse wdCollapseEnd

Selection.MoveUp , 1

Exit Sub

End If

firstWordTwo = Selection.range.Paragraphs(2).range.Words(1)

myLine1 = LCase(Selection.range.Paragraphs(1).range.Text)

myLine2 = LCase(Selection.range.Paragraphs(2).range.Text)

tabPos = InStr(myLine1, vbTab)

If tabPos > 0 Then

myLine1 = Mid(myLine1, tabPos + 1)

End If

tabPos = InStr(myLine2, vbTab)

If tabPos > 0 Then

myLine2 = Mid(myLine2, tabPos + 1)

End If

myLine1 = Replace(myLine1, "-", " ")

myLine1 = Replace(myLine1, ChrW(8216), "")

myLine1 = Replace(myLine1, "'", "")

myLine1 = Replace(myLine1, ",", "")

myLine2 = Replace(myLine2, "-", " ")

myLine2 = Replace(myLine2, ChrW(8216), "")

myLine2 = Replace(myLine2, "'", "")

myLine2 = Replace(myLine2, ",", "")

' Check the alphabetism

If checkSecondWord = True Then

If myLine1 > myLine2 Then

Selection.Collapse wdCollapseEnd

Selection.MoveDown Unit:=wdParagraph, Count:=1

Selection.MoveUp Unit:=wdParagraph, Count:=1

Selection.MoveStart Unit:=wdParagraph, Count:=-2

Exit Sub

End If

Else

If myLine1 > myLine2 And firstWordTwo <> firstWordOne Then

Selection.Collapse wdCollapseEnd

Selection.MoveDown Unit:=wdParagraph, Count:=1

Selection.MoveUp Unit:=wdParagraph, Count:=1

Selection.MoveStart Unit:=wdParagraph, Count:=-2

Exit Sub

End If

End If

Selection.MoveStart Unit:=wdParagraph, Count:=1

firstWordOne = Selection.Words(1)

Loop Until Selection.End = ActiveDocument.Content.End

Selection.MoveUp , 1

Beep

End Sub

**Sub VancouverCitationChecker()**

' Paul Beverley - Version 11.04.20

' Finds all citations and creates a list

' Citations in parenthese

' myType = 1

' Citations in square brackets

' myType = 2

' Citations as superscript

myType = 3

Set rng = ActiveDocument.Content

allCites = ""

Select Case myType

Case 1

mySearch = "\([0-9, \-" & ChrW(8211) & ChrW(8722) & "]@\)"

Case 2

mySearch = "\[[0-9, \-" & ChrW(8211) & ChrW(8722) & "]@\]"

Case 3

mySearch = "[0-9,\-" & ChrW(8211) & ChrW(8722) & "]{1,}"

End Select

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySearch

.Wrap = wdFindStop

If myType = 3 Then .Font.Superscript = True

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

thisOne = rng.Text

Set rng2 = ActiveDocument.Range(rng.End, rng.End + 1)

If rng.Text <> " " And InStr(" " & vbCr, rng2) > 0 Then

rng.Select

If thisOne > "" Then

If thisOne = "[1]" And allCites > "" Then

allCites = allCites & vbCr & vbCr & "References" & vbCr

End If

allCites = allCites & thisOne & vbCr

Else

rng.End = rng.Start + 3

End If

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Set thisDoc = ActiveDocument

Documents.Add

Selection.TypeText allCites

Selection.HomeKey Unit:=wdStory

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = wdBrightGreen

Set rng = ActiveDocument.Content

With rng.Find

.Text = " "

.Replacement.Text = " "

.Replacement.Highlight = True

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Go back to main document (for use with FRedit)

' thisdoc.Activate

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub VancouverAllCited()**

' Paul Beverley - Version 26.05.18

' Creates a list of all cited Vancouver references

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[[0-9, -" & ChrW(8211) & "]@\]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

allCites = ""

Do While Selection.Find.Found = True

Selection.MoveEnd wdCharacter, -1

Selection.MoveStart wdCharacter, 1

allCites = allCites & "," & Selection

Selection.Collapse wdCollapseEnd

' Go and find the next occurence (if there is one)

Selection.Find.Execute

Loop

Documents.Add DocumentType:=wdNewBlankDocument

Selection.TypeText Text:=allCites

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " "

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.Text = ","

.Replacement.Text = "^p"

.Forward = True

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

' Expand numbers to three digitss

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]{1,}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While Selection.Find.Found = True

If Len(Selection) = 2 Then Selection.InsertBefore "0"

If Len(Selection) = 1 Then Selection.InsertBefore "00"

Selection.Collapse wdCollapseEnd

' Go and find the next occurence (if there is one)

Selection.Find.Execute

Loop

Selection.WholeStory

Selection.Sort ExcludeHeader:=False, FieldNumber:="Paragraphs", \_

SortOrder:=wdSortOrderAscending, SortFieldType:=wdSortFieldAlphanumeric

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub ReferencesCollator()**

' Paul Beverley - Version 11.07.18

' Finds reference lists, and colours, collates and sorts them

myTitle = "References"

stopWords = "Table ,Figure ,Figures ,[[[[,<CH>"

Dim myCol(10)

myCol(0) = wdBlack

myCol(1) = wdBlue

myCol(2) = wdPink

myCol(3) = wdRed

myCol(4) = wdBrightGreen

myCol(5) = wdDarkBlue

myCol(6) = wdTurquoise

myColTotal = 7

wd = Split(stopWords, ",")

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^m"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Selection.End = 2

Selection.Copy

Selection.Collapse wdCollapseStart

myIndex = 0

i = 0

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myTitle & "^p"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

chapNum = 0

Do While Selection.Find.Found = True

chapNum = chapNum + 1

Selection.Start = myIndex

Selection.Expand wdParagraph

Selection.TypeText Text:=vbCr & "Chapter " & chapNum & vbCr

gogo = True

Do

Selection.Expand wdParagraph

myLine = Selection.range.Text

Selection.Collapse wdCollapseEnd

For j = 0 To UBound(wd)

If Left(myLine, Len(wd(j))) = wd(j) Then gogo = False

Next j

Loop Until gogo = False

Selection.MoveLeft , 1

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Selection.Start = myIndex

Selection.range.Font.ColorIndex = myCol(i)

i = (i + 1) Mod 7

Selection.Collapse wdCollapseEnd

myIndex = Selection.End

Selection.Find.Execute

Loop

Selection.End = ActiveDocument.Content.End

Selection.Delete

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending

Selection.HomeKey Unit:=wdStory

Selection.MoveEnd , 1

Do While InStr(vbCr & Chr(12), Selection.Text) > 0

Selection.Delete

Selection.MoveEnd , 1

Loop

Selection.Collapse wdCollapseStart

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13Chapter [0-9]{1,}>"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Beep

End Sub

**Sub Yesterdate()**

**'** Paul Beverley - Version 09.01.12

' Type yesterday's date into the text

Selection.TypeText Text:=Format(Date - 1, "d mmmm yyyy")

End Sub

**Sub CompareTexts()**

' Paul Beverley - Version 14.11.19

' Compares copied text (i.e. clipboard contents) with selected text

myColour = wdYellow

minLength = 10

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

myText = Selection

Selection.range.HighlightColorIndex = myColour

myStart = Selection.Start

myEnd = Selection.End

Selection.Collapse wdCollapseEnd

Selection.Paste

Selection.Start = myEnd

myOtherText = Selection

WordBasic.editunDo

If myText = myOtherText Then

Selection.Start = myStart

Selection.range.HighlightColorIndex = wdNoHighlight

Selection.Collapse wdCollapseEnd

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

Selection.Start = myStart

wds = Selection.range.Words.Count

Do

Do

Selection.MoveEnd wdWord, -1

txtPos = InStr(myOtherText, Selection.Text)

If txtPos > 0 Then

Selection.range.HighlightColorIndex = wdNoHighlight

Selection.Collapse wdCollapseEnd

Selection.End = myEnd

Selection.MoveStart wdWord, 1

DoEvents

End If

Loop Until Len(Selection.Text) < minLength

Selection.End = myEnd

Selection.MoveStart wdWord, 1

Loop Until Len(Selection.Text) < minLength

Beep

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub CompareTextsOLD()**

' Paul Beverley - Version 07.12.13

' Select non-identical text (was called IdenticalTextCheck)

myColour = wdGray25

myText = Selection

Selection.Range.HighlightColorIndex = wdNoHighlight

myStart = Selection.Start

myEnd = Selection.End

Selection.End = Selection.Start

Selection.Paste

Selection.Start = myStart

myOtherText = Selection

WordBasic.editunDo

If myText = myOtherText Then

Beep

Else

For myCount = 1 To Len(myText)

If Left(myText, myCount) <> Left(myOtherText, myCount) Then Exit For

Next myCount

myChangedStart = myCount - 1

For myCount = 1 To Len(myText)

If Right(myText, myCount) <> Right(myOtherText, myCount) Then Exit For

Next myCount

myChangedEnd = myCount - 1

Selection.Start = myEnd - myChangedEnd

Selection.End = myEnd

Selection.Range.HighlightColorIndex = myColour

Selection.Start = myStart

Selection.End = myStart + myChangedStart

Selection.Range.HighlightColorIndex = myColour

Selection.Collapse wdCollapseEnd

End If

End Sub

**Sub SpaceBeforeSelector()**

' Paul Beverley - Version 18.02.13

' Increase and decrease space before paragraph

myValues = "0,3,6,9,12,0"

' myValues = "0,12,9,6,3,0"

' myValues = "0,6,12,0"

' myValues = "6,9,6"

spNow = Trim(Str(Selection.ParagraphFormat.SpaceBefore))

myValues = "," & myValues & ","

myStart = InStr(myValues, "," & spNow & ",")

If myStart = 0 Then myStart = 0: spNow = ""

spNext = Mid(myValues, myStart + Len(spNow) + 2)

spNext = Left(spNext, InStr(spNext, ",") - 1)

Selection.ParagraphFormat.SpaceBefore = Val(spNext)

End Sub

**Sub SpaceAfterSelector()**

' Paul Beverley - Version 18.02.13

' Increase and decrease space after paragraph

myValues = "0,3,6,9,12,0"

' myValues = "0,12,9,6,3,0"

' myValues = "0,6,12,0"

' myValues = "6,9,6"

spNow = Trim(Str(Selection.ParagraphFormat.SpaceAfter))

myValues = "," & myValues & ","

myStart = InStr(myValues, "," & spNow & ",")

If myStart = 0 Then myStart = 0: spNow = ""

spNext = Mid(myValues, myStart + Len(spNow) + 2)

spNext = Left(spNext, InStr(spNext, ",") - 1)

Selection.ParagraphFormat.SpaceAfter = Val(spNext)

End Sub

**Sub IndentsMeasure()**

' Paul Beverley - Version 01.02.13

' Measure and display current indent settings

lft = PointsToCentimeters(Selection.ParagraphFormat.LeftIndent)

fst = PointsToCentimeters(Selection.ParagraphFormat.FirstLineIndent)

rt = PointsToCentimeters(Selection.ParagraphFormat.RightIndent)

lft = Int(lft \* 100 + 0.5) / 100

fst = Int(fst \* 100 + 0.5) / 100

rt = Int(rt \* 100 + 0.5) / 100

MsgBox ("Left = " & lft & " First = " & fst & " Right = " & rt)

End Sub

**Sub IndentsSet()**

' Paul Beverley - Version 01.02.13

' Set indent settings

Selection.ParagraphFormat.LeftIndent = CentimetersToPoints(1.27)

Selection.ParagraphFormat.FirstLineIndent = CentimetersToPoints(-1.27)

Selection.ParagraphFormat.RightIndent = CentimetersToPoints(0)

End Sub

**Sub IndentLeftSelector()**

' Paul Beverley - Version 18.02.13

' Set left indent settings to specific values

myValues = "0,0.5,1,-1,-0.5,0"

' myValues = "0,-1,1,0"

' myValues = "0,1.27,-1.27"

myValues = "1"

Dim myValue(10)

myValues = myValues & ","

myValues = Replace(myValues, ",,", ",")

numItems = Len(myValues) - Len(Replace(myValues, ",", ""))

myValues = myValues & Left(myValues, InStr(myValues, ",") - 1) & ","

For i = 0 To numItems

leftText = Left(myValues, InStr(myValues, ",") - 1)

myValue(i) = Val(leftText)

myValues = Mid(myValues, Len(leftText) + 2)

Next i

myValue(i + 1) = myValue(0)

indentNow = PointsToCentimeters(Selection.ParagraphFormat.LeftIndent)

indentNow = Int(indentNow \* 100 + 0.5) / 100

For i = 0 To numItems

If myValue(i) = indentNow Then Exit For

Next i

Selection.ParagraphFormat.LeftIndent = CentimetersToPoints(myValue(i + 1))

End Sub

**Sub IndentFirstSelector()**

' Paul Beverley - Version 18.02.13

' Set left indent settings to specific values

myValues = "0,0.5,1,-1,-0.5,0"

' myValues = "0,-1,1,0"

' myValues = "0,1.27,-1.27,0"

' myValues = "1"

Dim myValue(10)

myValues = myValues & ","

myValues = Replace(myValues, ",,", ",")

numItems = Len(myValues) - Len(Replace(myValues, ",", ""))

myValues = myValues & Left(myValues, InStr(myValues, ",") - 1) & ","

For i = 0 To numItems

leftText = Left(myValues, InStr(myValues, ",") - 1)

myValue(i) = Val(leftText)

myValues = Mid(myValues, Len(leftText) + 2)

Next i

myValue(i + 1) = myValue(0)

indentNow = PointsToCentimeters(Selection.ParagraphFormat.FirstLineIndent)

indentNow = Int(indentNow \* 100 + 0.5) / 100

For i = 0 To numItems

If myValue(i) = indentNow Then Exit For

Next i

Selection.ParagraphFormat.FirstLineIndent = CentimetersToPoints(myValue(i + 1))

End Sub

**Sub IndentRightSelector()**

' Paul Beverley - Version 18.02.13

' Set left indent settings to specific values

myValues = "0,2,4,6"

' myValues = "0,1,2,0"

' myValues = "0,0.5,1,0"

Dim myValue(10)

myValues = myValues & ","

myValues = Replace(myValues, ",,", ",")

numItems = Len(myValues) - Len(Replace(myValues, ",", ""))

myValues = myValues & Left(myValues, InStr(myValues, ",") - 1) & ","

For i = 0 To numItems

leftText = Left(myValues, InStr(myValues, ",") - 1)

myValue(i) = Val(leftText)

myValues = Mid(myValues, Len(leftText) + 2)

Next i

myValue(i + 1) = myValue(0)

indentNow = PointsToCentimeters(Selection.ParagraphFormat.RightIndent)

indentNow = Int(indentNow \* 100 + 0.5) / 100

For i = 0 To numItems

If myValue(i) = indentNow Then Exit For

Next i

Selection.ParagraphFormat.RightIndent = CentimetersToPoints(myValue(i + 1))

End Sub

**Sub DisplayedTextFormat()**

' Paul Beverley - Version 28.03.17

' Removes quotes and romanises (and trims trailing spaces)

removeItalic = True

myDeletables = " '""" & ChrW(8216) & ChrW(8217) & \_

ChrW(8220) & ChrW(8221) & Chr(9)

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.Expand wdParagraph

If removeItalic = True Then Selection.range.Font.Italic = False

Selection.End = Selection.Start + 1

If InStr(myDeletables, Selection) > 0 Then Selection.Delete

Selection.Collapse wdCollapseStart

Selection.End = Selection.Start + 1

If InStr(myDeletables, Selection) > 0 Then Selection.Delete

Selection.Collapse wdCollapseStart

Selection.TypeText Text:="<DQ>"

Selection.Expand wdParagraph

theEnd = Selection.End

Selection.Start = theEnd - 2

Selection.End = theEnd - 1

If InStr(myDeletables, Selection) > 0 Then Selection.Delete

For i = 1 To 5

Selection.MoveStart , -1

Selection.End = Selection.Start + 1

If InStr(myDeletables, Selection) > 0 Then Selection.Delete

Next i

Selection.Expand wdParagraph

With Selection.ParagraphFormat

.SpaceBeforeAuto = False

.SpaceAfterAuto = False

.FirstLineIndent = CentimetersToPoints(-0)

End With

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.TypeText Text:="</DQ>"

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub UnderlineStyle()**

' Paul Beverley - Version 01.10.18

' Changes the underline style of underlined text

myTypes = ",1,2,3,4,6,7,9,10,11,20,23,25,26,27,39,43,55"

myNum = Split(myTypes, ",")

totNums = UBound(myNum)

myStyle = Val(InputBox("1 - Single" & vbCr & "2 - Words" \_

& vbCr & "3 - Double" & vbCr & "4 - Dotted" & \_

vbCr & "5 - Thick" & vbCr & "6 - Dash" & vbCr \_

& "7 - DotDash" & vbCr & "8 - DotDotDash" & \_

vbCr & "9 - Wavy" & vbCr & "10 - DottedHeavy" \_

& vbCr & "11 - DashHeavy" & vbCr & \_

"12 - DotDashHeavy" & vbCr & "13 - DotDotDashHeavy" \_

& vbCr & "14 - WavyHeavy" & vbCr & "15 - DashLong" \_

& vbCr & "16 - WavyDouble" & vbCr & \_

"17 - DashLongHeavy", "UnderlineStyle"))

If myStyle < 1 Or myStyle > 17 Then Beep: Exit Sub

myUnderline = Val(myNum(myStyle))

For i = 1 To totNums

If i <> myStyle Then

If Selection.Start = Selection.End Then

Set rng = ActiveDocument.Content

Else

Set rng = Selection.range.Duplicate

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = Val(myNum(i))

.Replacement.Font.Underline = myUnderline

.Replacement.Text = ""

.Wrap = False

.MatchCase = False

.Execute Replace:=wdReplaceAll

End With

End If

Next i

End Sub

**Sub FormatNumbers()**

**'** Paul Beverley - Version 02.03.24

' Formats number at cursor or numbers within a selection

decimalFormat = "###,###,###,0.00"

nonDecimalFormat = "###,###,###,0"

fourDigitComma = False

commaReplacement = ""

' commaReplacement = ChrW(8201) ' thin space

' commaReplacement = ChrW(160) ' non-breaking space

If Selection.Start = Selection.End Then Selection.Expand wdCharacter

Set allSelection = Selection.Range.Duplicate

myMarkup = ActiveWindow.View.RevisionsFilter.Markup

ActiveWindow.View.RevisionsFilter.Markup = wdRevisionsMarkupSimple

Dim rng As Range

Do

myCount = myCount + 1

Set rng = Selection.Range.Duplicate

rng.End = rng.Start + 1

ch = rng.Text

If InStr("0123456789", ch) > 0 Then

Do

rng.MoveStart , -1

DoEvents

ch = rng.Characters.First

Loop Until InStr("0123456789.,", ch) = 0

rng.MoveStart , 1

Do

rng.MoveEnd , 1

DoEvents

ch = rng.Characters.Last

Loop Until InStr("0123456789.,", ch) = 0

rng.MoveEnd , -1

If rng.Font.Superscript = 0 Then

dotPos = InStr(1, rng.Text, ".")

If dotPos > 0 Then

rng = Format(rng.Text, decimalFormat)

If dotPos = 5 And fourDigitComma = False Then

txt = rng.Text

newText = Left(txt, 1) & Mid(txt, 3)

rng.Text = newText

End If

Else

If Len(rng.Text) <> 4 Or fourDigitComma = True Then \_

rng = Format(rng.Text, nonDecimalFormat)

End If

If commaReplacement > "" Then

newText = Replace(rng.Text, ",", commaReplacement)

rng.Text = newText

End If

Else

rng.HighlightColorIndex = wdBrightGreen

End If

rng.Collapse wdCollapseEnd

rng.Select

End If

Selection.MoveRight , 1

DoEvents

Loop Until Selection.End > (allSelection.End - 1) Or myCount > 1000

ActiveWindow.View.RevisionsFilter.Markup = myMarkup

End Sub

**Sub OxfordCommaSelectiveDelete()**

' Paul Beverley - Version 07.02.20

' Moves to next Oxford comma and/or deletes current comma first

myDelete = ""

myJumpNext = "."

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ", and>"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While Selection.Find.Found = True

myResponse = InputBox("Delete?", "Oxford Comma Selective Kill")

Select Case myResponse

Case myDelete: Selection.Collapse wdCollapseStart

Selection.MoveEnd , 1

Selection.Delete

Case myJumpNext

Case Else: Exit Sub

End Select

Selection.Collapse wdCollapseEnd

Selection.Find.Execute

Loop

End Sub

**Sub RulersShow()**

' Paul Beverley - Version 10.02.20

' Switches Word’s ruler on and off

ActiveWindow.ActivePane.DisplayRulers = Not \_

ActiveWindow.ActivePane.DisplayRulers

End Sub

**Sub However()**

' Paul Beverley - Version 15.06.21

' Starts a sentence with 'However', and removes the later "however"

Selection.MoveLeft wdWord, 1

Selection.TypeText Text:="However, "

Selection.MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdExtend

Selection.Text = LCase(Selection.Text)

Selection.MoveRight Unit:=wdWord, Count:=10, Extend:=wdExtend

hPosn = InStr(Selection, "however")

If hPosn > 0 Then

Selection.MoveStart , hPosn - 1

Selection.End = Selection.Start + 8

Selection.Delete

End If

End Sub

**Sub MoveToStart()**

' Paul Beverley - Version 06.02.23

' Moves the selected text to the beginning of the sentence

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Else

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Expand wdWord

Selection.Start = rng.Start

rng.End = Selection.End

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.End = rng.End

End If

Set rng = Selection.Range.Duplicate

Selection.Range.Characters(1) = \_

UCase(Selection.Range.Characters(1))

rng.Select

rng.Collapse wdCollapseEnd

rng.MoveEnd , 1

If rng.Text = " " Then

Selection.Cut

Selection.MoveStart , -1

Selection.Delete

Selection.Expand wdSentence

Selection.Collapse wdCollapseStart

Selection.TypeText Text:=" "

Selection.MoveEnd , 1

Selection.Text = LCase(Selection)

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.Paste

Else

' With a punctuation mark

Selection.Cut

Selection.MoveStart , -1

Selection.Delete

Selection.Expand wdSentence

Selection.Collapse wdCollapseStart

Selection.MoveEnd , 1

Selection.Text = LCase(Selection)

Selection.Collapse wdCollapseStart

Selection.Paste

Selection.TypeText Text:=" "

End If

End Sub

**Sub SmallCapWord()**

' Paul Beverley - Version 15.06.21

' Makes the current word smallcaps

If Selection.Start = Selection.End Then \_

Selection.Words(1).Select

Selection.Text = LCase(Selection.Text)

Selection.Font.SmallCaps = True

End Sub

**Sub ReplicateThisEdit()**

' Paul Beverley - Version 07.01.17

' Replicates the change just made, through the rest of the file

killItalic = True

myColour = wdYellow

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

myReplace = Selection

myFind = ""

' thin spaces for spaces

If InStr(myReplace, ChrW(8201)) > 0 Then myFind = Replace(myReplace, ChrW(8201), " ")

' en dashes for hyphen

If InStr(myReplace, ChrW(8211)) > 0 Then myFind = Replace(myReplace, ChrW(8211), "-")

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

If myFind = "" Then

Selection.Range.Revisions.RejectAll

myFind = Selection

WordBasic.EditUndo

If myFind <> myReplace Then

ActiveDocument.TrackRevisions = True

killItalic = False

End If

End If

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = False

.Replacement.Text = myReplace

If ActiveDocument.TrackRevisions = False Then .Replacement.Highlight = True

If killItalic Then .Replacement.Font.Italic = False

.Forward = True

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myReplace

.Wrap = False

.Forward = True

.MatchCase = True

.MatchWildcards = False

.Execute

End With

If Selection.Find.Found = False Then

Beep

Else

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myReplace

.Wrap = False

.Forward = False

.MatchCase = True

.MatchWildcards = False

.Execute

End With

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub WhatChar()**

**'** Paul Beverley - Version 07.06.23

' Show ASCII and unicode and character names

' Decide where to show results

showOnStatusbar = False

showInMessageBox = True

useVoice = False

' To use voice, remove the apostrophe from the following line

If useVoice = True Then Set speech = New SpVoice

' If the macro gives an error on locked files, make this True

lockedFile = False

lockedFile = True

' If the character is a space, don't show a message box.

justBeepForSpace = False

' If you also want to know what the hex of the ANSI code is:

showHexANSI = True

' If you also want to use this code in FRedit:

prepareFReditCode = True

If showOnStatusbar + showInMessageBox + useVoice = 0 Then \_

Beep: MsgBox "Alter options at the top of the macro code to show results.": \_

Exit Sub

charSelected = (Selection.Start <> Selection.End)

If AscW(Selection) = 32 And justBeepForSpace Then Beep: Exit Sub

CR2 = vbCr & vbCr

If Selection.Start = Selection.End Then Selection.MoveEnd wdCharacter, 1

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.MoveEnd , 1

If lockedFile = False Then

Selection.MoveEnd , 1

Selection.Range.Revisions.AcceptAll

End If

ansicode = AscW(rng)

nextCharCode = AscW(Right(rng.Text, 1))

uCode = Val(Dialogs(wdDialogInsertSymbol).CharNum)

If Asc(ansicode) = 63 And uCode <> 63 Then

ansiBit = "ANSI: ???"

Else

ansiBit = ">>>>>> ANSI: " & Str(ansicode)

If showHexANSI = True Then ansiBit = ansiBit & \_

" (hex " & Hex(ansicode) & ")"

End If

hexCode = Replace(Hex(uCode), "FFFF", "")

ucodeBit = vbCr & "Unicode: " & Str(uCode) & " (hex " & \_

hexCode & ")"

If Selection.Font.Name <> \_

ActiveDocument.Styles(wdStyleNormal).Font.Name Then

fontBit = vbCr & "Font: " & Selection.Font.Name

Else

fontBit = ""

End If

fntSize = Selection.Font.Size

normalSize = ActiveDocument.Styles(wdStyleNormal).Font.Size

fontBit = fontBit & " Font Size: " & Str(fntSize)

HexBit = ""

If uCode < 0 Then

HexBit = vbCr & vbCr & "For FRedit, use Find: <&H" & hexCode & ">"

If prepareFReditCode = True Then

Selection.Collapse wdCollapseStart

startCode = Selection.Start

Selection.TypeText "<&H" & hexCode & ">|"

Selection.Start = startCode

Selection.Cut

End If

End If

' To correct for Mac codes

Select Case ansicode

Case 2: ExtraBit = "Foot note end note marker"

Case 32: ExtraBit = "Ordinary space"

Case 34: ExtraBit = "Straight double quote"

Case 39: ExtraBit = "Straight apostrophe"

Case 40: If Selection.Font.Name = "Symbol" Then ExtraBit = "Funny Symbol font character!"

Case 45: ExtraBit = "Ordinary hyphen"

Case 46: ExtraBit = "Full point"

Case 48: ExtraBit = "Number 0"

Case 49: ExtraBit = "Number 1"

Case 50: ExtraBit = "Number 2"

Case 51: ExtraBit = "Number 3"

Case 58: ExtraBit = "Colon"

Case 59: ExtraBit = "Semicolon"

Case 63: ExtraBit = "Question mark"

Case 73: ExtraBit = "Uppercase I(eye)"

Case 76: ExtraBit = "Uppercase L"

Case 79: ExtraBit = "Uppercase O"

Case 88: ExtraBit = "Uppercase X"

Case 96: ExtraBit = "Back tick"

Case 105: ExtraBit = "Lowercase i (eye)"

Case 108: ExtraBit = "Lowercase l (el)"

Case 111: ExtraBit = "Lowercase o"

Case 120: ExtraBit = "Lowercase x"

Case 150: ExtraBit = "En dash"

Case 151: ExtraBit = "Em dash"

Case 174: ExtraBit = "Registered trademark"

Case 176: ExtraBit = "Proper degree symbol"

Case Else: ExtraBit = ""

End Select

' The unicode information will overwrite the ANSI

Select Case uCode

Case 9: ExtraBit = "Tab character"

Case 11: ExtraBit = "Manual line break"

Case 13: ExtraBit = "Just an ordinary paragraph end"

Case 30: ExtraBit = "Non-breaking hyphen"

Case 31: ExtraBit = "Optional hyphen"

Case 124: ExtraBit = "Vertical bar"

Case 160: ExtraBit = "Non-breaking space"

Case 176: ExtraBit = "Proper degree symbol"

Case 178: ExtraBit = "Dodgy squared symbol!"

Case 179: ExtraBit = "Dodgy cubed symbol!"

Case 180: ExtraBit = "Dodgy apostrophe!"

Case 186: ExtraBit = "Masculine ordinal"

Case 215: ExtraBit = "Proper multiply symbol"

Case 937: ExtraBit = "Omega"

Case 956: ExtraBit = "Mu = micro"

Case 8194: ExtraBit = "En space"

Case 8195: ExtraBit = "Em space"

Case 8201: ExtraBit = "Thin space"

Case 8216: ExtraBit = "Single open curly quote"

Case 8217: ExtraBit = "Single close curly quote = apostrophe"

Case 8211: ExtraBit = "En dash"

Case 8212: ExtraBit = "Em dash"

Case 8220: ExtraBit = "Double open curly quote"

Case 8221: ExtraBit = "Double close curly quote"

Case 8222: ExtraBit = "German open curly quote"

Case 8226: ExtraBit = "Ordinary bullet"

Case 8230: ExtraBit = "Ellipsis"

Case 8242: ExtraBit = "Unicode: single prime"

Case 8243: ExtraBit = "Unicode: double prime"

Case 8249: ExtraBit = "French open quote"

Case 8250: ExtraBit = "French close quote"

Case 8722: ExtraBit = "Proper minus sign"

End Select

If ExtraBit > "" Then ExtraBit = " >>>> " & ExtraBit

If Selection.Font.Superscript = True Then

ExtraBit = ExtraBit & " >>>> superscripted"

End If

If Selection.Font.Subscript = True Then

ExtraBit = ExtraBit & " >>>> subscripted"

End If

If Selection.Font.Italic = True Then

ExtraBit = ExtraBit & " >>>> italic"

End If

If nextCharCode > 688 And nextCharCode < 861 Then

If useVoice = False Then Beep

ExtraBit = ExtraBit & " >>>> followed by a no-space diacritic" & Str$(nextCharCode)

End If

' This isn't a 'funny' Symbol font, undo the Revisions.AcceptAll

If lockedFile = False And uCode > 0 Then WordBasic.EditUndo

' Show on the status bar

s = " "

If showOnStatusbar = True Then \_

StatusBar = s & s & s & s & ansiBit & " " & ExtraBit \_

& " " & ucodeBit & " " & fontBit & HexBit

If useVoice And ExtraBit > "" And charSelected = False Then \_

speech.Speak ExtraBit, SVSFPurgeBeforeSpeak

' Display in a message box

If (useVoice And charSelected) Or showInMessageBox \_

Or ExtraBit = "" Or uCode < 0 Then \_

MsgBox ansiBit & ExtraBit & vbCr & ucodeBit & vbCr & fontBit \_

& HexBit, , "WhatChar"

Selection.Collapse wdCollapseStart

End Sub

**Sub Chirimbolos()**

' Paul Beverley - Version 04.08.20

' Chirimbolos - WhatChar in SPANISH by Marcela Ronaina

' Muestra el nombre y código ASCII/unicode de un caracter

' y el formato si es diferente a la plantilla Normal

' Macro adaptada de Paul Beverley

' To use voice, "uncomment" this line:

Set speech = New SpVoice

' And set this as True

useVoice = True

' Decidir dónde mostrar el resultado

showOnStatusbar = False

showInMessageBox = True

' No mostrar CD si el caracter es un espacio

justBeepForSpace = True

' Mostrar también valor hex del código ANSI

showHexANSI = True

' If you also want to use this code in FRedit:

prepareFReditCode = False

If showOnStatusbar + showInMessageBox + useVoice = 0 Then \_

Beep: MsgBox "Alter options in macro code to show results.": \_

Exit Sub

charSelected = (Selection.Start <> Selection.End)

If AscW(Selection) = 32 Then Beep: Exit Sub

CR2 = vbCr & vbCr

' Si no hay nada seleccionado, usar caracter a la derecha

If Selection.Start = Selection.End Then Selection.MoveEnd wdCharacter, 1

' aceptar todos los cambios de la selección

Selection.Range.Revisions.AcceptAll

ansicode = Asc(Selection)

uCode = Val(Dialogs(wdDialogInsertSymbol).CharNum)

If Asc(ansicode) = 63 And uCode <> 63 Then

ansiBit = "ANSI: ???"

Else

ansiBit = ">>>>>> ANSI: " & Str(ansicode) & \_

" (hex " & Hex(ansicode) & ")"

End If

' expresar uCode en hexadecimal

hexCode = Replace(Hex(uCode), "FFFF", "")

ucodeBit = vbCr & "Unicode: " & Str(uCode) & " (hex " & \_

hexCode & ")"

'Mostrar fuente/tamaño/color si difiere del estilo Normal

fntName = Selection.Font.Name

normalName = ActiveDocument.Styles(wdStyleNormal).Font.Name

If normalName <> fntName Then

fontBit = vbCr & "Fuente: " & fntName

Else

fontBit = ""

End If

fntSize = Selection.Font.Size

normalSize = ActiveDocument.Styles(wdStyleNormal).Font.Size

If normalSize <> fntSize Then \_

fontBit = fontBit & " Tamaño: " & Str(fntSize)

fntColor = Selection.Font.ColorIndex

normalColor = ActiveDocument.Styles(wdStyleNormal).Font.ColorIndex

If normalColor <> fntColor Then \_

fontBit = fontBit & " Color: " & Str(fntColor)

fntNormal = "Normal:" & vbCr & "Fuente: " & normalName & \_

" Tamaño: " & Str(normalSize) & \_

" Color: " & Str(normalColor)

' Cut the FREditBit to the clipboard, which you can then paste to a script

HexBit = ""

If uCode < 0 Then

HexBit = CR2 & "For FRedit, use Find: <&H" & hexCode & ">"

If prepareFReditCode = True Then

Selection.Collapse wdCollapseStart

startCode = Selection.Start

Selection.TypeText Text:="<&H" & hexCode & ">|"

Selection.Start = startCode

Selection.Cut

End If

End If

' To correct for Mac codes

Select Case ansicode

Case 2: extraBit = "Remisión a una nota"

Case 32: extraBit = "Espacio"

Case 34: extraBit = "Comillas rectas (doble)"

Case 39: extraBit = "Comilla recta (sencilla) ¡Ojo! No se usa."

Case 40: If Selection.Font.Name = "Symbol" Then extraBit = "¡Ni idea qué es esto!"

Case 44: extraBit = "Coma"

Case 45: extraBit = "Guion"

Case 46: extraBit = "Punto"

Case 48: extraBit = "0 (número)"

Case 49: extraBit = "1 (número)"

Case 50: extraBit = "2 (número)"

Case 51: extraBit = "3 (número)"

Case 58: extraBit = "Dos puntos"

Case 59: extraBit = "Punto y coma"

Case 60: extraBit = "Paréntesis angulado (apertura)/Menor que (Matem.)"

Case 62: extraBit = "Paréntesis angulado (cierre)/Mayor que (Matem.)"

Case 63: extraBit = "Horizontal Bar"

Case 73: extraBit = "I mayúscula"

Case 76: extraBit = "L mayúscula"

Case 79: extraBit = "O mayúscula"

Case 88: extraBit = "X mayúscula"

Case 95: extraBit = "Subraya"

Case 96: extraBit = "Acento grave ¡suelto!"

Case 105: extraBit = "i minúscula"

Case 108: extraBit = "l (ele) minúscula"

Case 111: extraBit = "o minúscula"

Case 120: extraBit = "x minúscula"

Case 139: extraBit = "Comilla angular simple (apertura)"

Case 155: extraBit = "Comilla angular simple (cierre)"

Case 150: extraBit = "Semirraya "

Case 151: extraBit = "Raya"

Case 161: extraBit = "Exclamación (apertura)"

Case 174: extraBit = "Marca registrada"

Case 171: extraBit = "Comillas latinas (apertura)"

Case 176: extraBit = "Grado"

Case 187: extraBit = "Comillas latinas (cierre)"

Case Else: extraBit = ""

End Select

' The unicode information will overwrite the ANSI

Select Case uCode

Case 9: extraBit = "Marca de tabulación"

Case 11: extraBit = "Salto de línea manual"

Case 13: extraBit = "Marca de párrafo"

Case 30: extraBit = "Guion de no separación"

Case 31: extraBit = "Guion opcional"

Case 47: extraBit = "Barra"

Case 124: extraBit = "Pleca"

Case 160: extraBit = "Espacio de no separación"

Case 167: extraBit = "Párrafo/Sección"

Case 176: extraBit = "Grado"

Case 178: extraBit = "Dodgy squared symbol!"

Case 179: extraBit = "Dodgy cubed symbol!"

Case 180: extraBit = "Acento agudo ¡Tilde suelto!"

Case 182: extraBit = "Calderón"

Case 183: extraBit = "Punto alto o centrado"

Case 186: extraBit = "Ordinal masculino"

Case 215: extraBit = "Multiplicación"

Case 937: extraBit = "Omega"

Case 956: extraBit = "Mu = micro"

Case 8194: extraBit = "en-space"

Case 8195: extraBit = "em-space"

Case 8201: extraBit = "Espacio fino"

Case 8216: extraBit = "Comilla simple (apertura)"

Case 8217: extraBit = "Comilla simple (cierre) = apóstrofo"

Case 8211: extraBit = "Semirraya"

Case 8212: extraBit = "Raya"

Case 8220: extraBit = "Comillas inglesas (apertura)"

Case 8221: extraBit = "Comillas inglesas (cierre)"

Case 8222: extraBit = "Comilla alemana (apertura)"

Case 8226: extraBit = "Bolo"

Case 8230: extraBit = "Puntos suspensivos"

Case 8242: extraBit = "Minuto de ángulo >> Unicode: Prima"

Case 8243: extraBit = "Segundo de ángulo >> Unicode: Doble prima"

Case 8249: extraBit = "Comilla angular simple (apertura)"

Case 8250: extraBit = "Comilla angular simple (cierre)"

Case 8722: extraBit = "Menos (Mat.)"

End Select

If extraBit > "" Then extraBit = " >>>> " & extraBit

' Revisar superíndice/subíndice,cursiva,negrita y versalitas

If Selection.Font.Superscript = True Then

extraBit = extraBit & " >>> superíndice"

Else

If Selection.Font.Subscript = True Then

extraBit = extraBit & " >>> subíndice"

End If

End If

If Selection.Font.Italic = True Then

extraBit = extraBit & " >>> cursiva"

End If

If Selection.Font.Bold = True Then

extraBit = extraBit & " >>> negrita"

End If

If Selection.Font.SmallCaps = True Then

extraBit = extraBit & " >>> versalitas"

End If

' This isn't a 'funny' Symbol font, undo the Revisions.AcceptAll

If uCode > 0 Then WordBasic.EditUndo

' Mostrar en la Barra de estado

s = " "

If showOnStatusbar = True Then \_

StatusBar = s & s & s & s & ansiBit & " " & \_

extraBit & " " & \_

ucodeBit & " " & fontBit & HexBit

If useVoice And extraBit > "" And charSelected = False Then \_

speech.Speak extraBit, SVSFPurgeBeforeSpeak

' Mostrar en un Cuadro de diálogo

If (useVoice And charSelected) Or showInMessageBox \_

Or extraBit = "" Or uCode < 0 Then \_

MsgBox ansiBit & extraBit & vbCr & ucodeBit & vbCr & fontBit \_

& HexBit & vbCr & fntNormal, , "¿Qué chirimbolo es este?"

Selection.Collapse wdCollapseStart

End Sub

**Sub TextProbe()**

' Paul Beverley - Version 05.05.17

' Find funny character codes

showEverything = False

showUnicode = True

showReturn = False ' code = 13

showDashes = False

showCurlyQuotes = False

showThinSpaces = False

showTab = False ' code = 9

showTableCellMarker = True ' code = 7

showPictureMarker = True ' code = 1

showNewPage = False ' code = 12

showNoteMarker = True ' code = 2

showSoftReturn = True ' code = 11

showFieldBraces = True ' code = 21

If showEverything = True Then

showUnicode = True

showReturn = True ' code = 13

showDashes = True

showThinSpaces = True

showCurlyQuotes = True

showTab = True ' code = 9

showTableCellMarker = True ' code = 7

showPictureMarker = True ' code = 1

showNewPage = True ' code = 12

showNoteMarker = True ' code = 2

showSoftReturn = True ' code = 11

showFieldBraces = True ' code = 21

End If

Set rng = ActiveDocument.range

theEnd = rng.End

Selection.MoveEnd wdWord, -1

rng.Start = Selection.Start

rng.End = Selection.Start

Do

rng.MoveEnd wdWord, 1

' Check if any character < 31 or any unicode

myWord = rng

showIt = False

For myChar = 1 To Len(myWord)

i = Asc(Mid(myWord, myChar, 1))

If showCurlyQuotes = False Then

If i = 145 Or i = 146 Or i = 148 Or i = 149 Then showCode = False

End If

If i < 32 Then

showCode = True

If showReturn = False And i = 13 Then showCode = False

If showTab = False And i = 9 Then showCode = False

If showNewPage = False And i = 12 Then showCode = False

If showNoteMarker = False And i = 2 Then showCode = False

If showSoftReturn = False And i = 11 Then showCode = False

If showTableCellMarker = False And i = 7 Then showCode = False

If showPictureMarker = False And i = 1 Then showCode = False

If showFieldBraces = False And i = 21 Then showCode = False

If showCode = True Then showIt = True

End If

If showUnicode = True Then

i = AscW(Mid(myWord, myChar, 1))

Select Case i

Case 8216: If showCurlyQuotes = True Then showIt = True

Case 8217: If showCurlyQuotes = True Then showIt = True

Case 8220: If showCurlyQuotes = True Then showIt = True

Case 8221: If showCurlyQuotes = True Then showIt = True

Case 8211: If showDashes = True Then showIt = True

Case 8212: If showDashes = True Then showIt = True

Case 8201: If showThinSpaces = True Then showIt = True

Case Else: If i > 255 Then showIt = True

End Select

End If

Next myChar

If showIt = True Then

' Now make up string of text + codes

rng.Select

foundThis = ""

For i = 1 To Len(myWord)

myChar = Mid(myWord, i, 1)

If Asc(myChar) > 31 Then

Uni = AscW(myChar)

If showUnicode = True And Uni > 255 Then

foundThis = foundThis & "{" & myChar & " = " & Trim(Str(Uni)) & "}"

Else

foundThis = foundThis & myChar

End If

Else

Select Case Asc(myChar)

Case 1: nameCode = " (Picture marker) "

Case 2: nameCode = " (Note marker) "

Case 7: nameCode = " (Table cell marker) "

Case 9: nameCode = " (Tab) "

Case 11: nameCode = " (Soft return) "

Case 12: nameCode = " (New page) "

Case 13: nameCode = " (CR) "

Case 21: nameCode = " (Field braces) "

Case 30: nameCode = " (Non-breaking hyphen) "

Case 31: nameCode = " (Optional hyphen) "

Case Else: nameCode = ""

End Select

foundThis = foundThis & "[" & Trim(Str(Asc(myChar))) & "]" & nameCode

End If

Next i

myResponse = MsgBox(foundThis & vbCr & vbCr \_

& "Continue?", vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

End If

rng.MoveStart wdWord, 1

Loop Until rng.End = theEnd

rng.Select

Selection.Start = Selection.End

Beep

End Sub

**Sub SpellcheckWordUK()**

**'** Paul Beverley - Version 25.08.11

' Spellcheck single word UK

' Alt-`

Selection.Words(1).Select

If Application.CheckSpelling(Selection, \_

MainDictionary:=Languages(wdEnglishUK).NameLocal) = False Then

ActiveDocument.CheckSpelling

Else

Selection.Start = Selection.End

beep

End If

End Sub

**Sub SpellcheckWordUS()**

**'** Paul Beverley - Version 25.08.11

' Spellcheck single word US

Selection.Words(1).Select

If Application.CheckSpelling(Selection, \_

MainDictionary:=Languages(wdEnglishUS).NameLocal) = False Then

ActiveDocument.CheckSpelling

Else

Selection.Start = Selection.End

beep

End If

End Sub

**Sub SpellcheckWordCurrent()**

**'** Paul Beverley - Version 10.12.11

' Spellcheck single word in current language

' Alt-`

Selection.Words(1).Select

If Application.CheckSpelling(Selection, \_

MainDictionary:=Languages(Selection.LanguageID).NameLocal) = False Then

ActiveDocument.CheckSpelling

Else

Selection.End = Selection.Start

beep

End If

End Sub

**Sub SpellcheckWordUSUK()**

**'** Paul Beverley - Version 19.12.11

' Spellcheck single word in NOT the current language

Selection.Words(1).Select

If Selection.LanguageID = wdEnglishUK Then

checkLang = wdEnglishUS

Else

checkLang = wdEnglishUK

End If

If Application.CheckSpelling(Selection, \_

MainDictionary:=Languages(checkLang).NameLocal) = False Then

ActiveDocument.CheckSpelling

Else

Selection.Start = Selection.End

Beep

End If

End Sub

**Sub Spellcheck()**

**'** Paul Beverley - Version 06.12.10

' Run a spellcheck

' F7

If Selection.LanguageID = wdEnglishUS Then beep

ActiveDocument.CheckSpelling

End Sub

**Sub SpellcheckWarn()**

**'** Paul Beverley - Version 30.04.11

' Run a spellcheck, but beep if not UK or langauges are mixed

' F7

langHere = Selection.LanguageID

If langHere <> wdEnglishUK Then Beep

Set rng = ActiveDocument.Content

langMix = rng.LanguageID

If langMix > 9999 Then MsgBox \_

("Beware: multiple languages within document")

ActiveDocument.CheckSpelling

End Sub

**Sub SpellWordChange()**

' Paul Beverley - Version 14.01.21

' Accepts Word's spelling suggestion for current word

' Select the word, and nothing but the word

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

langText = Languages(Selection.LanguageID).NameLocal

' Make erWord and actual Word 'word'

Set erWord = Selection.Words.First

' If it's a spelling error...

If Application.CheckSpelling(erWord, MainDictionary:=langText) = False Then

Set suggList = Application.GetSpellingSuggestions(erWord, MainDictionary:=langText)

If suggList.Count > 0 Then

' and if an alternative is available, type it

Selection.TypeText suggList.Item(1).Name

Else

' otherwise just colour it

Selection.Range.HighlightColorIndex = wdGray25

End If

' Beep twice

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Else

' But if correctly spelt, beep once and move on

Beep

End If

Selection.Start = Selection.End + 1

End Sub

**Sub SpellingShowToggle()**

' Paul Beverley - Version 25.05.20

' Switches visible spelling error indication on and off

Set rng = Selection.Range.Duplicate

Application.ScreenUpdating = False

Selection.HomeKey Unit:=wdStory

If Options.CheckSpellingAsYouType = True Then

Options.CheckSpellingAsYouType = False

ActiveDocument.ShowSpellingErrors = False

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Else

Options.CheckSpellingAsYouType = True

ActiveDocument.ShowSpellingErrors = True

Beep

End If

Selection.TypeText Text:="!"

Selection.MoveStart , -1

Selection.Delete

rng.Select

Application.ScreenUpdating = True

End Sub

**Sub CountPhrase()**

' Paul Beverley - Version 18.07.22

' Counts this word or phrase

doFormatCount = True

doCountWhole = True

' If nothing is selected, select the current word

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

If InStr(Selection, " ") = 0 Then justOneWord = True

oldStart = Selection.Start

oldEnd = Selection.End

myPhrase = Trim(Selection)

thisBit = Replace(myPhrase, "^", "^^")

thisBit = Replace(thisBit, Chr(13), "^p")

If Right(thisBit, 1) = ChrW(8217) Then thisBit \_

= Left(thisBit, Len(thisBit) - 1)

CR = vbCr: CR2 = CR & CR

' Find whether we're in a footnote

InANote = Selection.Information(wdInFootnote)

If InANote = True Then

lineJump = 0

Do

Selection.MoveUp Unit:=wdLine, count:=1

lineJump = lineJump + 1

Loop Until Selection.Information(wdInFootnote) = False

oldStart = Selection.Start

oldEnd = Selection.Start

End If

Set rng = ActiveDocument.Content

at = rng.Text

myTot = ActiveDocument.Range.End

ntsText = ""

' Are there any footnotes?

If ActiveDocument.Footnotes.Count > 0 Then

ntsText = ntsText & ActiveDocument.StoryRanges(wdFootnotesStory).Text

at = at & ntsText

End If

If ActiveDocument.Endnotes.Count > 0 Then

ntsText = ntsText & ActiveDocument.StoryRanges(wdEndnotesStory).Text

at = at & ntsText

End If

at = Replace(at, Chr(2), "")

' Count all occurences

aTlcase = LCase(at)

myTot = Len(at)

allCount = Len(Replace(aTlcase, LCase(myPhrase), myPhrase & "!")) - myTot

myText = "Any case: " & Str(allCount) & CR

' Count case sensitively

caseCount = Len(Replace(at, myPhrase, myPhrase & "!")) - myTot

myText = myText & "Exact same case: " & Str(caseCount) & CR2

If doFormatCount = True Then

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

myTotNow = ActiveDocument.Range.End

' Count bold italic

With rng.Find

.ClearFormatting

.MatchCase = False

.Text = myPhrase

.Font.Italic = True

.Font.Bold = True

.Replacement.Text = "^&!"

.Execute Replace:=wdReplaceAll

End With

biCount = ActiveDocument.Range.End - myTotNow

If biCount > 0 Then

WordBasic.EditUndo

myText = myText & "Bold italic (main text) : " \_

& Str(biCount) & CR

End If

' Count italic

With rng.Find

.ClearFormatting

.MatchCase = False

.Font.Italic = True

.Execute Replace:=wdReplaceAll

End With

iCount = ActiveDocument.Range.End - myTotNow

If iCount > 0 Then

WordBasic.EditUndo

myText = myText & "Italic: " \_

& Str(iCount) & CR

End If

' Count bold

With rng.Find

.ClearFormatting

.Font.Bold = True

.Execute Replace:=wdReplaceAll

End With

bCount = ActiveDocument.Range.End - myTotNow

If bCount > 0 Then

WordBasic.EditUndo

myText = myText & "Bold: " & \_

Str(bCount) & CR2

End If

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

ActiveDocument.TrackRevisions = myTrack

End If

If doCountWhole = True Then

chs = " , . ! : [ ] { } ( ) / \ + "

chs = chs & ChrW(8220) & " "

chs = chs & ChrW(8221) & " "

chs = chs & ChrW(8201) & " "

chs = chs & ChrW(8222) & " "

chs = chs & ChrW(8217) & " "

chs = chs & ChrW(8216) & " "

chs = chs & ChrW(8212) & " "

chs = chs & ChrW(8722) & " "

chs = chs & vbCr & " "

chs = chs & vbTab & " "

chs = " " & chs & " "

chs = Replace(chs, " ", " ")

chs = Left(chs, Len(chs) - 1)

chars = Split(chs, " ")

For i = 1 To UBound(chars)

at = Replace(at, chars(i), " ")

Next i

' Count as whole words (case sensitive)

If justOneWord Then

p = " " & myPhrase & " "

at = Replace(at, " ", " ")

aTlcase = LCase(at)

myTot = Len(at)

wholeWdCaseCount = Len(Replace(at, p, \_

p & "!")) - myTot

wholeWdNoCaseCount = Len(Replace(aTlcase, LCase(p), \_

p & "!")) - myTot

myText = myText & "Whole words (Any case):" & \_

Str(wholeWdNoCaseCount) & CR

myText = myText & "Whole words (Exact same case):" & \_

Str(wholeWdCaseCount) & CR

titleText = "Characters searched: """

Else

titleText = "Phrase searched: """

End If

End If

printResult:

Selection.End = oldStart

Selection.MoveRight Unit:=wdCharacter, count:=1

Selection.Start = oldStart

Selection.End = oldEnd

myText = titleText & myPhrase & """" & CR2 & myText

MsgBox myText, 0, "CountPhrase"

End Sub

**Sub HyphenSpaceWordCount()**

**'** Paul Beverley - Version 13.05.24

' Counts hyphenated word forms

numChars = Len(Selection.Text)

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

tx = Selection.Text

For i = 2 To Len(tx) - 2

wd1 = Left(tx, i)

wd2 = Mid(tx, i + 1)

myPair = tx

If Application.CheckSpelling(wd1) And \_

Application.CheckSpelling(wd2) Then

myPair = wd1 & " " & wd2

Exit For

End If

Next i

myPair = InputBox("OK? (Add a space if necessary)", "HyphenSpaceWordCount", myPair)

If InStr(myPair, " ") = 0 Then Beep: Exit Sub

Else

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

rng.Select

myPair = Selection

End If

Debug.Print myPair

brkPos = InStr(myPair, " ")

If brkPos = 0 Then brkPos = InStr(myPair, "-")

If brkPos = 0 Then brkPos = InStr(myPair, ChrW(8211))

If brkPos = 0 Then brkPos = InStr(myPair, "/")

If brkPos = 0 Then

Beep

myResponse = MsgBox("Can't work out where the spilt is, sorry.", vbOKOnly, \_

"HyphenSpaceWordCount")

Exit Sub

End If

wd1 = LCase(Left(myPair, brkPos - 1))

wd2 = LCase(Mid(myPair, brkPos + 1))

Set rng = ActiveDocument.Content

allText = LCase(rng.Text)

myTot = Len(allText)

p = wd1 & " " & wd2

spaceCount = Len(Replace(allText, p, p & "!")) - myTot

p = wd1 & "-" & wd2

hyphenCount = Len(Replace(allText, p, p & "!")) - myTot

p = wd1 & ChrW(8211) & wd2

dashCount = Len(Replace(allText, p, p & "!")) - myTot

p = wd1 & "/" & wd2

slashCount = Len(Replace(allText, p, p & "!")) - myTot

p = wd1 & wd2

oneWordCount = Len(Replace(allText, p, p & "!")) - myTot

myResult = wd1 & wd2 & ": " & Str(oneWordCount) & vbCr

myResult = myResult & wd1 & " " & wd2 & ": " & \_

Str(spaceCount) & vbCr

myResult = myResult & wd1 & "-" & wd2 & ": " & Str(hyphenCount) \_

& vbCr & vbCr

myResult = myResult & wd1 & "<dash>" & wd2 & ": " & \_

Str(dashCount) & vbCr

myResult = myResult & wd1 & "/" & wd2 & ": " & Str(slashCount)

' Load WC find box

mySch = Left(wd1, Len(wd1) - 1) & "[" & Right(wd1, 1) & \_

"^32/\-" & ChrW(8211) & Left(wd2, 1) & "]{2,3}" & Mid(wd2, 2)

myInit = Left(mySch, 1)

mySch = "[" & LCase(myInit) & UCase(myInit) & "]" & Mid(mySch, 2)

Selection.Collapse wdCollapseStart

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySch

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

End With

MsgBox myResult, 0, "HyphenSpaceWordCount"

End Sub

**Sub CountRemainderSimple()**

' Paul Beverley - Version 25.04.16

' Count words below the cursor

wordsTotal = ActiveDocument.Content.Words.Count

Selection.End = ActiveDocument.Content.End

wordsLeft = Selection.range.Words.Count

Selection.Collapse wdCollapseStart

perCent = Int(1000 \* (wordsLeft / wordsTotal)) / 10

MsgBox (perCent & "% left. (Very roughly " & \_

Int(0.0007 \* wordsLeft) & " thousand words)")

End Sub

**Sub CountRemainder()**

**'** Paul Beverley - Version 27.06.23

' Counts words below the cursor

altDisplay = False

' altDisplay = True

Dim wordsTotal As Long, wordsLeft As Long, wordsDone As Long

On Error GoTo ReportIt

wordsTotal = ActiveDocument.Content.ComputeStatistics(wdStatisticWords)

On Error GoTo 0

' Turn to a string with single decimal place

If altDisplay = True Then

totWords = Trim(Str(Int(wordsTotal / 1000))) & "," & Right(Trim(wordsTotal), 3)

Else

totWords = Trim(Str(Int(wordsTotal / 100) / 10))

If InStr(totWords, ".") = 0 Then totWords = totWords & ".0"

If wordsTotal < 1000 Then totWords = "0" & totWords

End If

' Count remainder

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

wordsLeft = rng.ComputeStatistics(wdStatisticWords)

wordsDone = wordsTotal - wordsLeft

If altDisplay = True Then

readWords = Trim(Str(Int(wordsDone / 1000))) & "," & Right(Trim(wordsDone), 3)

Else

readWords = Trim(Str(Int(wordsDone / 100) / 10))

If InStr(readWords, ".") = 0 Then readWords = readWords & ".0"

If wordsDone < 1000 Then readWords = "0" & readWords

End If

' Calculate words to go as a proportion of actual total

If altDisplay = True Then

wordsToGo = Trim(Str(Int(wordsLeft / 1000))) & "," & Right(Trim(wordsLeft), 3)

Else

wordsToGo = Trim(Str(Int(wordsLeft / 100) / 10))

If InStr(wordsToGo, ".") = 0 Then wordsToGo = wordsToGo & ".0"

If wordsLeft < 1000 Then wordsToGo = "0" & wordsToGo

End If

pCent = Trim(Str(Int(wordsLeft / wordsTotal \* 100)))

myResponse = MsgBox(wordsToGo & " left, out of " & Trim(totWords) & vbCr \_

& vbCr & readWords & " done." & vbCr & vbCr & "To go: " \_

& pCent & " %", vbOKOnly, "CountRemainder")

Exit Sub

ReportIt:

If Err.Number = 4658 Then

lang = "Unknown language"

If Selection.LanguageID = wdEnglishUK Then lang = "UK English"

If Selection.LanguageID = wdEnglishUS Then lang = "US English"

myResponse = MsgBox("Mixed language setting used." & vbCr & vbCr & \_

"Set language to " & lang & "?", vbQuestion + vbYesNoCancel, \_

"CountRemainder")

If myResponse <> vbYes Then

Exit Sub

Else

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

ActiveDocument.Content.LanguageID = Selection.LanguageID

ActiveDocument.TrackRevisions = myTrack

Resume

End If

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub** **ItalicCount()**

**'** Paul Beverley - Version 22.07.10

' Counts the number of words that are in italic

Selection.EndKey Unit:=wdStory

theEnd = Selection.Start

' copy all the footnotes to the end of the text

If ActiveDocument.Footnotes.Count > 0 Then

For Each fn In ActiveDocument.Footnotes

fn.Range.Copy

Selection.Paste

Next

End If

' copy all the endnotes to the end of the text

If ActiveDocument.Endnotes.Count > 0 Then

For Each fn In ActiveDocument.Endnotes

fn.Range.Copy

Selection.Paste

Next

End If

' copy all the textboxes to the end of the text

Set rng = ActiveDocument.Range

rng.Start = rng.End

If ActiveDocument.Shapes.Count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.TextFrame.HasText Then

Set rng2 = shp.TextFrame.TextRange

rng2.Copy

rng.Paste

rng.Start = rng.End

End If

Next

End If

Selection.HomeKey Unit:=wdStory

totItalic = 0

totRoman = 0

totChars = ActiveDocument.Characters.Count

For Each myChar In ActiveDocument.Characters

If myChar.Font.Italic = True Then

totItalic = totItalic + 1

Else

totRoman = totRoman + 1

End If

If totItalic Mod 100 = 0 Then StatusBar = \_

" Press Ctrl-Break to stop. " \_

& "Remaining: " & Int((totChars - totItalic - totRoman) / 100)

Next

' Delete all the added text

Selection.EndKey Unit:=wdStory

Selection.Start = theEnd

Selection.Delete

Selection.HomeKey Unit:=wdStory

StatusBar = ""

MsgBox ("Italic: " & totItalic & vbCrLf & vbCrLf \_

& "Roman: " & totRoman)

End Sub

**Sub ItalicWordList()**

' Paul Beverley - Version 19.09.18

' Creates a list of all words in italic

Call CopyTextSimple

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = False

.Replacement.Text = "^p"

.Forward = True

.Wrap = wdFindContinue

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13{1,}"

.Replacement.Text = "^p"

.Forward = True

.Wrap = wdFindContinue

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub MultiFileCount()**

' Paul Beverley - Version 13.12.17

' Count words in a group of files

useStats = True

useBoth = True

countEquations = True

Dim FilesTotal As Integer, myCountMain As Long, myCountAll As Long

Dim StatWordsMain As Long, StatWordsAll As Long, myErr As Long

Dim myCountMainTot As Long, myCountAllTot As Long

Dim StatWordsMainTot As Long, StatWordsAllTot As Long

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Count ALL the files in this folder:" & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "Multifile Wordcount")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Count just the files listed here?", \_

vbQuestion + vbYesNoCancel, "Multifile Wordcount")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

Documents.Add

Set myResults = ActiveDocument

totMathType = 0

numEqnEditor = 0

For i = 1 To numFiles

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set myDoc = Application.Documents.Open(FileName:=thisFile)

StatusBar = allMyFiles(i)

numMathType = ActiveDocument.InlineShapes.Count

numEqnEditor = ActiveDocument.OMaths.Count

DoEvents

' Clean up the document

ActiveDocument.ConvertNumbersToText

Set rng = ActiveDocument.Range

rng.Fields.Unlink

If ActiveDocument.Comments.Count > 0 Then ActiveDocument.DeleteAllComments

ActiveDocument.TrackRevisions = False

Selection.Range.Revisions.AcceptAll

' Mark the end of the main text

Selection.EndKey Unit:=wdStory

Selection.TypeText "zczczczc" & vbCr & vbCr

Selection.EndKey Unit:=wdStory

' copy all the footnotes to the end of the text

If ActiveDocument.Footnotes.Count > 0 Then

For Each fn In ActiveDocument.Footnotes

fn.Range.Copy

Selection.Paste

Selection.TypeText " "

Next

End If

' copy all the endnotes to the end of the text

If ActiveDocument.Endnotes.Count > 0 Then

For Each en In ActiveDocument.Endnotes

en.Range.Copy

Selection.Paste

Selection.TypeText " "

Next

End If

' copy all the textboxes to the end of the text

Set rng = ActiveDocument.Range

rng.Start = rng.End

If ActiveDocument.Shapes.Count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

Set rng2 = shp.TextFrame.TextRange

rng2.Cut

rng.Paste

rng.Start = rng.End

End If

End If

Next

End If

' Delete all notes

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Text = "^2"

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "[0-9].[0-9]"

.Replacement.Text = "111"

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "[0-9].[0-9]"

.Replacement.Text = "111"

.Execute Replace:=wdReplaceAll

End With

If useStats = True Or useBoth = True Then

' Ensure only a single language

Set rng = ActiveDocument.Content

rng.LanguageID = wdEnglishUK

' Count the whole of the text

Selection.EndKey Unit:=wdStory

StatWordsAll = ActiveDocument.Content.ComputeStatistics(wdStatisticWords) - 1

StatWordsAllTot = StatWordsAllTot + StatWordsAll

' Now delete the extra text ready for a recount

Set rng = ActiveDocument.Range

With rng.Find

.MatchWildcards = False

.Text = "zczczczc"

.Replacement.Text = " "

.Execute

End With

rng.End = ActiveDocument.Range.End

rng.Cut

' Count just the main text

Selection.HomeKey Unit:=wdStory

Selection.EndKey Unit:=wdStory

StatWordsMain = ActiveDocument.Content.ComputeStatistics(wdStatisticWords)

If StatWordsAll - StatWordsMain = -1 Then StatWordsMain = StatWordsAll

StatWordsMainTot = StatWordsMainTot + StatWordsMain

' Put the extra text back in

Selection.Paste

End If

If useStats = False Or useBoth = True Then

' Deal with hyphens and apostrophes (e.g. half-hearted and can't)

' treating them as single words

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "([a-zA-Z])[-'\&" & ChrW(8217) & "]([a-zA-Z])"

.Replacement.Text = "\1x\2"

.Execute Replace:=wdReplaceAll

End With

' Changes '/' into a space for "this/that" type occurences;

' also punctuation, brackets etc.

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "[\_/\(\)\<\>\&\@\[\]""'\!\?\,\.+\-;:…^l^m^t^13" & ChrW(8220) \_

& ChrW(8221) & ChrW(8216) & ChrW(8217) & ChrW(8226) & "]"

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = " [!a-zA-Z] "

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

' Count all the text -1 for zczc, and -1 because .Count adds one!

tot = ActiveDocument.Content.Words.Count - 2

myCountAll = tot

myCountAllTot = myCountAllTot + myCountAll

' Now delete the extra text ready for a recount

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "zczczczc"

.Replacement.Text = " "

.Execute

End With

rng.End = ActiveDocument.Range.End

rng.Cut

myCountMain = ActiveDocument.Content.Words.Count - 1

If myCountAll - myCountMain = -1 Then myCountMain = myCountAll

myCountMainTot = myCountMainTot + myCountMain

End If

myDoc.Close SaveChanges:=wdDoNotSaveChanges

FilesTotal = FilesTotal + 1

myFileName = Left(allMyFiles(i), InStr(allMyFiles(i), ".") - 1)

Selection.TypeText myFileName

If useStats = True Or useBoth = True Then \_

Selection.TypeText vbTab & Str(StatWordsMain) & \_

vbTab & Str(StatWordsAll - StatWordsMain) & vbTab & Str(StatWordsAll)

If useStats = False Or useBoth = True Then \_

Selection.TypeText vbTab & Str(myCountMain) & vbTab & \_

Str(myCountAll - myCountMain) & vbTab & Str(myCountAll)

If useBoth = True Then

myErr = myCountAll - StatWordsAll

myErrPC = Int(2000 \* myErr / (myCountAll + StatWordsAll)) / 10

If myErr = 0 Then

myPC = ChrW(8212)

Else

myPC = Trim(Str(Abs(myErrPC)))

If InStr(myPC, ".") = 0 Then myPC = myPC & ".0"

If Abs(myErrPC) < 1 Then myPC = "0" & myPC

myPC = myPC & "%"

If myErr > 0 Then

myPC = "+" & myPC

Else

myPC = ChrW(8722) & myPC

End If

End If

Selection.TypeText vbTab & myPC

End If

If countEquations = True And (numMathType + numEqnEditor) > 0 Then

Selection.TypeText vbTab & Str(numMathType) & vbTab & \_

Str(numEqnEditor)

totMathType = totMathType + numMathType

totEqnEditor = totEqnEditor + numEqnEditor

End If

Selection.TypeText vbCr

Next i

Selection.TypeText vbCr

Selection.TypeText "Totals (" & Trim(Str(FilesTotal)) & ChrW(160) & "files)"

If useStats = True Or useBoth = True Then \_

Selection.TypeText vbTab & Str(StatWordsMainTot) & vbTab & \_

Str(StatWordsAllTot - StatWordsMainTot) & vbTab & Str(StatWordsAllTot)

If useStats = False Or useBoth = True Then \_

Selection.TypeText vbTab & Str(myCountMainTot) & vbTab & \_

Str(myCountAllTot - myCountMainTot) & vbTab & Str(myCountAllTot)

If countEquations = True And (totMathType + totEqnEditor) > 0 Then \_

Selection.TypeText vbTab & vbTab & Str(totMathType) & vbTab & \_

Str(totEqnEditor)

' Add title line and make it bold

Selection.HomeKey Unit:=wdStory

Selection.TypeText "File"

If useStats = True Or useBoth = True Then \_

Selection.TypeText vbTab & "Main (stats)" & vbTab & \_

"Extras" & vbTab & "All"

If useStats = False Or useBoth = True Then \_

Selection.TypeText vbTab & "Main (count)" & vbTab & \_

"Extras" & vbTab & "All"

If useBoth = True Then Selection.TypeText vbTab & "Diff."

If countEquations = True And (totMathType + totEqnEditor) > 0 Then \_

Selection.TypeText vbTab & "Math Type" & vbTab & "Eqn Editor"

Selection.TypeText vbCr & vbCr

Selection.MoveUp Unit:=wdParagraph, Count:=1, Extend:=wdExtend

Selection.Start = 0

Selection.Font.Bold = True

' Make the totals line bold

Selection.EndKey Unit:=wdStory

Selection.MoveUp Unit:=wdParagraph, Count:=1, Extend:=wdExtend

Selection.Font.Bold = True

Selection.WholeStory

Set tbl = Selection.ConvertToTable(Separator:=wdSeparateByTabs)

tbl.Columns.AutoFit

Selection.HomeKey Unit:=wdStory

End Sub

**Sub** **WordTotaller()**

**'** Paul Beverley - Version 21.01.11

' Adds up word numbers in selected texts

addPageNum = True

wordsQuote = 3

myTotal = Selection.Words.Count

ss = Selection.Start

se = Selection.End

Selection.End = ss

Selection.MoveEnd wdWord, wordsQuote

myQuote = Selection

If addPageNum = True Then

Selection.Start = se

Selection.Fields.Add Range:=Selection.Range, Type:=wdFieldEmpty, Text:= \_

"PAGE \\* Arabic ", PreserveFormatting:=True

Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend

pageNum = Selection

Selection.Delete

Selection.Start = ss

Selection.End = se

End If

Set thisDoc = ActiveDocument

' Find the doc with the totals

For Each myDoc In Documents

myDoc.Activate

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = "====="

.Replacement.Text = ""

.Execute

End With

If rng.Find.Found Then

gotOne = True

Exit For

Else

gotOne = False

End If

Next myDoc

' If no totals file, create one

If gotOne = False Then

Documents.Add

Selection.TypeText Text:="=====" & vbCrLf & "0" & vbCrLf & vbCrLf

Selection.HomeKey Unit:=wdStory

Selection.MoveDown Unit:=wdLine, Count:=1, Extend:=wdExtend

Else

rng.End = rng.End + 1

rng.Select

End If

' Add current word number to total

Selection.InsertBefore Text:=Str(myTotal) & Chr(9) & "p." & \_

pageNum & " – " & myQuote & vbCrLf

Selection.Start = Selection.End

Selection.MoveDown Unit:=wdLine, Count:=1, Extend:=wdExtend

myTotal = myTotal + Val(Selection)

Selection.TypeText Text:=Str(myTotal)

Selection.MoveRight Unit:=wdCharacter, Count:=1

thisDoc.Activate

End Sub

**Sub ColumnTotal()**

' Paul Beverley - Version 12.05.15

' Check the column total

allowErrorPercent = 0.01

okChars = "0123456789.,-" & ChrW(8211) & ChrW(8212)

myTotal = 0

On Error GoTo ReportIt

Do

gogo = True

Selection.Expand wdCell

myText = Selection

' Changes sign if it starts with a hyphen or en dash

signBit = 1

aBit = Left(myText, 1)

' A hyphen or en or em dash means either minus

' or nothing (zero); this will cope with either

If aBit = "-" Or aBit = ChrW(8211) Or aBit = ChrW(8212) Then

myText = Right(myText, Len(myText) - 1)

signBit = -1

End If

' Remove any commas and find the value

previousNumber = thisNumber

thisNumber = signBit \* Val(Replace(myText, ",", ""))

If thisNumber = 0 And InStr(okChars, aBit) = 0 Then gogo = False

myTotal = myTotal + thisNumber

Selection.MoveDown Unit:=wdLine, Count:=1

' Go down a line and check for a characterless cell

Do

myNext = Asc(Selection)

If myNext <> 13 Then

Selection.MoveRight Unit:=wdCharacter, Count:=1

myNext = Asc(Selection)

End If

Loop Until myNext = 13

' Keep going until you drop out of the table

Loop Until Asc(Right(Selection, 1)) <> 7 Or gogo = False

If thisNumber = 0 Then thisNumber = previousNumber

' At this point, the total of the column should be twice

' the final figure, i.e. the total plus the total

myDiff = myTotal - 2 \* thisNumber

' Make the difference a positive number

If myDiff < 0 Then myDiff = -myDiff

' Check how small the difference is compared to the total

If myDiff / myTotal < allowErrorPercent / 100 Then

Beep

Else

MsgBox ("I make the total: " & myTotal - thisNumber)

End If

Exit Sub

ReportIt:

MsgBox ("Please ensure that the cursor is the a cell containing a number")

End Sub

**Sub NumberTotaller()**

' Paul Beverley - Version 30.11.22

' Sums the numbers within the selected text (or checks the sum)

myError = 0.0001

If Selection.Start = Selection.End Then

Beep

myResponse = MsgBox("Please select an area of text", \_

vbExclamation, "NumberTotaller")

Exit Sub

End If

Set rng = Selection.Range.Duplicate

myEnd = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9,.]{1,}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

myCount = 0

allNumText = ""

Do While rng.Find.Found = True

If InStr(",.", rng.Characters.Last) > 0 Then rng.MoveEnd , -1

If rng.End <> rng.Start Then

allNumText = allNumText & "!" & rng.Text

Else

rng.MoveStart , 1

End If

rng.Collapse wdCollapseEnd

rng.End = myEnd

rng.Find.Execute

DoEvents

Loop

allNumText = Replace(allNumText, ",", "")

myNum = Split(allNumText, "!")

numCount = UBound(myNum)

fstNum = Val(myNum(1))

lastNum = Val(myNum(numCount))

For i = 1 To UBound(myNum)

myTot = myTot + Val(myNum(i))

Next i

Beep

If Abs(myTot - 2 \* fstNum) < myError Or \_

Abs(myTot - 2 \* lastNum) < myError Then

Else

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

myResponse = MsgBox("Total = " & Str(myTot), vbOKOnly, "NumberTotaller")

End If

End Sub

**Sub FullFileNameCopy()**

' Paul Beverley - Version 09.05.17

' Copy the full file name of the current file to the clipboard

myFullName = ActiveDocument.FullName

Selection.Collapse wdCollapseEnd

hereNow = Selection.Start

Selection.TypeText Text:=myFullName

Selection.Start = hereNow

Selection.Copy

WordBasic.EditUndo

End Sub

**Sub GoogleFetchCA()**

' Paul Beverley - Version 18.11.21

' Launches selected text on Google Canada

mySite = "http://www.google.ca/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleFetchQuotesCA()**

' Paul Beverley - Version 18.11.21

' Launches selected text - with quotes - on Google Canada

mySite = "http://www.google.ca/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, vbCr, "")

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = "%22" & mySubject & "%22"

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleFetchUS()**

' Paul Beverley - Version 18.11.21

' Launches selected text on Google.Com

mySite = "http://www.google.ca/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleFetchQuotesUS()**

' Paul Beverley - Version 18.11.21

' Launches selected text - with quotes - on Google.Com

mySite = "http://www.google.com/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, vbCr, "")

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = "%22" & mySubject & "%22"

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleFetch()**

**'** Paul Beverley - Version 29.05.24

' Launches selected text on Google

alsoCopySubject = False

' mySite = "http://www.worldcat.org/search?q="

mySite = "http://www.google.co.uk/search?q="

mySite = "http://www.google.co.uk/search?q="

' Or for Google Scholar, use

' mySite = "https://scholar.google.com/scholar?q="

' USA

' mySite = "http://www.google.com/search?q="

' Canada

' mySite = "http://www.google.ca/search?q="

' Australia

' mySite = "http://www.google.com.au/search?q="

' India

' mySite = "http://www.google.co.in/search?q="

' Netherlands

' mySite = "http://www.google.nl/search?q="

' Germany

' mySite = "http://www.google.de/search?q="

' South Africa

' mySite = "http://www.google.co.za/search?q="

' Chile

' mySite = "https://www.google.com.cl/search?q="

'Colombia

' mySite = "http://www.google.com.co/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

If Len(Selection) < 3 Then

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

If alsoCopySubject = True Then Selection.Copy

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

Debug.Print mySite & mySubject

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleBooksFetch()**

' Paul Beverley - Version 20.07.22

' Launches selected text on Google Books

mySite = "https://www.google.com/search?tbm=bks&q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleFetchQuotes()**

' Paul Beverley - Version 18.11.21

' Launches selected text - with quotes - on Google

mySite = "http://www.google.co.uk/search?q="

' Or for Google Scholar

' mySite = "https://scholar.google.com/scholar?q="

' USA

' mySite = "http://www.google.com/search?q="

' Canada

' mySite = "http://www.google.ca/search?q="

' India

' http://www.google.co.in/search?q=

' Netherlands

' mySite = "http://www.google.nl/search?q="

' Germany

' mySite = "http://www.google.de/search?q="

' India

' mySite = "http://www.google.co.in/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, vbCr, "")

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = "%22" & mySubject & "%22"

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleScholarFetch()**

' Paul Beverley - Version 06.12.21

' Launches selected text on Google Scholar

mySite = "https://scholar.google.com/scholar?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub** **DTMEfetch()**

' Paul Beverley - Version 10.05.22

' Launches selected text on

mySite = "http://dtme.ranm.es/buscador.aspx?NIVEL\_BUS=3&LEMA\_BUS="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub BibleHubFetch()**

' Paul Beverley - Version 20.07.22

' Launches selected text on BibleHub

mySite = "https://biblehub.com/bsb/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "'", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.MoveStart , -2

If InStr("123", Left(Selection.Text, 1)) = 0 Then \_

Selection.MoveStart , 2

myChar = Right(Selection.Text, 1)

Do While InStr("0123456789.: ", myChar) > 0

Selection.MoveEnd , 1

myChar = Right(Selection.Text, 1)

If LCase(myChar) <> UCase(myChar) Then Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

Selection.Copy

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub BibleGatewayFetch()**

**'** Paul Beverley - Version 12.04.23

' Launches Bible references on BibleGateway

mySite = "https://www.biblegateway.com/passage/?search="

myVersion = "NIV"

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "'", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.MoveStart , -2

If InStr("123", Left(Selection.Text, 1)) = 0 Then \_

Selection.MoveStart , 2

Selection.MoveEnd , 1

myChar = Right(Selection.Text, 1)

Do While InStr("0123456789,.:- ", myChar) > 0

Selection.MoveEnd , 1

myChar = Right(Selection.Text, 1)

Debug.Print myChar

If LCase(myChar) <> UCase(myChar) Then Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

mySubject = mySubject & "&version=" & myVersion

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleFetchDE()**

' Paul Beverley - Version 18.10.21

' Launches selected text on Google

mySite = "http://www.google.de/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleFetchQuotesDE()**

' Paul Beverley - Version 18.10.21

' Launches selected text - with quotes - on Google

mySite = "http://www.google.de/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, vbCr, "")

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = "%22" & mySubject & "%22"

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub TeAkaFetch()**

' Paul Beverley - Version 29.05.22

' Launches selected text on the Maori dictionary Te Aka

mySite = "https://maoridictionary.co.nz/search?keywords="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub NgramFetch()**

' Paul Beverley - Version 18.10.21

' Launches selected text on Google Ngram

mySite = "https://books.google.com/ngrams/graph?content="

yearStart = "1800"

' yearStart = ""

yearEnd = "2000"

' yearEnd = ""

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, " ,", ",")

mySubject = Replace(mySubject, ", ", ",")

mySubject = Replace(mySubject, ",", "%2C")

mySubject = Replace(mySubject, ChrW(8217), "'")

Debug.Print mySite & mySubject

If yearStart > "" Then mySubject = mySubject & \_

"&year\_start=" & yearStart

If yearEnd > "" Then mySubject = mySubject & \_

"&year\_end=" & yearEnd

Debug.Print mySite & mySubject

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

End Sub

**Sub GoogleMapFetch()**

**'** Paul Beverley - Version 06.09.23

' Launches selected text to Google Maps

mySite = "http://www.google.com.eg/maps/dir/"

myHome = "NR86TR"

myWork = "NR13JQ"

myChars = ".,!? '""" & vbCr & ChrW(8217) & ChrW(8221)

If Selection.Start = Selection.End Then

Set rng = Selection.Range.Duplicate

Do

rng.MoveEnd , 1

lastChar = rng.Characters.Last

DoEvents

Loop Until InStr(myChars, lastChar) > 0

rng.MoveEnd , -1

Do

rng.MoveStart , -1

firstChar = rng.Characters.First

DoEvents

Loop Until InStr(myChars, firstChar) > 0 Or rng.Start = 0

If rng.Start > 0 Then rng.MoveStart , 1

Else

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

End If

mySubject = Trim(rng)

mySubject = Replace(mySubject, vbCr, "")

mySubject = Replace(mySubject, " to ", "/")

myPlace = Split(mySubject, "/")

mySubject = ""

For i = 0 To UBound(myPlace)

If myPlace(i) = "h" Then myPlace(i) = myHome

If myPlace(i) = "w" Then myPlace(i) = myWork

mySubject = mySubject & myPlace(i) & "/"

Next i

mySubject = mySubject & "/"

mySubject = Replace(mySubject, "//", "")

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

Debug.Print mySubject

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

End Sub

**Sub** **BLcatalogueFetch()**

' Paul Beverley - Version 21.11.20

' Launches selected text on Google

mySite = "http://explore.bl.uk/primo\_library/libweb/action/search.do;jsessionid=ECBBDE1655942F9F323B807D6BFFDC36?fn=search&ct=search&initialSearch=true&mode=Basic&tab=local\_tab&indx=1&dum=true&srt=rank&vid=BLVU1&frbg=&tb=t&vl%28freeText0%29=<>&scp.scps=scope%3A%28BLCONTENT%29&vl%282084770704UI0%29=any&vl%282084770704UI0%29=title&vl%282084770704UI0%29=any"

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

mySite = Replace(mySite, "<>", mySubject)

ActiveDocument.FollowHyperlink Address:=mySite

End Sub

**Sub OneLookFetch()**

' Paul Beverley - Version 18.10.21

' Launches selected text to OneLook.com

mySite = "http://www.onelook.com"

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub MerriamFetch()**

**'** Paul Beverley - Version 03.04.24

' Launches selected text to Merriam

mySite = "https://app.macquariedictionary.com.au/?search\_word\_type=dictionary&word="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub** **MerriamLegalFetch()**

' Paul Beverley - Version 13.09.20

' Launches selected text to Merriam-Webster legal website

mySite = "https://unabridged.merriam-webster.com/legal/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub MerriamCollegiateFetch()**

' Paul Beverley - Version 02.08.22

' Launches selected text to Merriam-Webster collegiate website

mySite = "https://unabridged.merriam-webster.com/collegiate/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub MerriamUnabridgedFetch()**

' Paul Beverley - Version 13.09.20

' Launches selected text to Merriam-Webster unabridged website

mySite = "https://unabridged.merriam-webster.com/unabridged/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub MerriamThesaurusFetch()**

' Paul Beverley - Version 13.09.20

' Launches selected text to Merriam-Webster thesaurus website

mySite = "https://unabridged.merriam-webster.com/thesaurus/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub MerriamMedicalFetch()**

' Paul Beverley - Version 13.09.20

' Launches selected text to Merriam-Webster medical website

mySite = "https://www.merriam-webster.com/medical/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub LawDictionaryFetch()**

' Paul Beverley - Version 13.09.20

' Launches selected text to Dictionary.Law website

mySite = "https://dictionary.law.com/Default.aspx?typed="

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject & "&type=1"

Selection.Collapse wdCollapseEnd

End Sub

**Sub WikiFetch()**

' Paul Beverley - Version 18.10.21

' Launches selected text on Wikipedia

mySite = "http://en.wikipedia.org/wiki/Special:Search?search="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub WikiFetchDE()**

' Paul Beverley - Version 18.10.21

' Launches selected text on Wikipedia

mySite = "http://de.wikipedia.org/wiki/Special:Search?search="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub OUPFetch()**

' Paul Beverley - Version 08.08.20

' Launches selected text to OUP

mySite = "http://en.oxforddictionaries.com/definition/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub OUPFetchPremium()**

**'** Paul Beverley - Version 10.06.24

' Launches selected text to OUP Premium website.

mySite = "https://premium.oxforddictionaries.com/definition/english/"

myWait = 0.75

' Depending on your browser + internet connection

' you might need a longer wait, or could get away with less.

allowRerun = True

' allowRerun = False

testMode = False

' testMode = True

If Selection.Start = Selection.End Then

Selection.Expand wdWord

If LCase(Selection) = UCase(Selection) Then

Selection.Collapse wdCollapseStart

Selection.MoveStart wdWord, -1

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

Selection.Copy

ActiveDocument.FollowHyperlink Address:=mySite

If testMode = True Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

myTime = Timer

Do

Loop Until Timer > myTime + myWait

SendKeys "^v"

SendKeys "{ENTER}"

myTime = Timer

If allowRerun = True Then

myResponse = MsgBox("Rerun the search?! (Or press<Esc>)", \_

vbQuestion + vbYesNoCancel, "OUPFetchPremium")

If myResponse <> vbYes Then

SendKeys "{NUMLOCK}"

Exit Sub

End If

Else

SendKeys "{NUMLOCK}"

Exit Sub

End If

ActiveDocument.FollowHyperlink Address:=mySite

myTime = Timer

Do

Loop Until Timer > myTime + myWait

SendKeys "^v"

SendKeys "{ENTER}"

SendKeys "{NUMLOCK}"

End Sub

**Sub** **RAEfetch()**

' Paul Beverley - Version 10.05.22

' Launches selected text on Real Academia Española

mySite = "https://dle.rae.es/"

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub PubMedFetch()**

' Paul Beverley - Version 08.08.20

' Launches selected text to PubMed

mySite = "http://www.ncbi.nlm.nih.gov/pubmed/?term="

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub ThesaurusFetch()**

' Paul Beverley - Version 08.08.20

' Launch selected text to thesaurus.com

mySite = "http://thesaurus.com/browse/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub ThesaurusFetchDE()**

' Paul Beverley - Version 08.08.20

' Launch selected text to openthesaurus.de

mySite = "https://www.openthesaurus.de/synonyme/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub DictionaryFetch()**

' Paul Beverley - Version 08.08.20

' Launches selected text to dictionary.com

mySite = "http://dictionary.com/browse/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub** **DictionaryMultipleFetch()**

' Paul Beverley - Version 17.06.21

' Launches selected text to dictionary.com and other dictionaries

numSites = 3

ReDim mySite(numSites) As String

mySite(1) = "http://dictionary.com/browse/"

mySite(2) = "https://en.oxforddictionaries.com/definition/"

mySite(3) = "https://www.collinsdictionary.com/dictionary/english/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

For i = 1 To numSites

ActiveDocument.FollowHyperlink Address:=mySite(i) & mySubject

Selection.Collapse wdCollapseEnd

Next i

End Sub

**Sub DictionaryFetchDE()**

' Paul Beverley - Version 08.08.20

' Launches selected text to duden.de

mySite = "https://www.duden.de/suchen/dudenonline/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub MacquarieFetch()**

' Paul Beverley - Version 08.08.20

' Launches selected text to the Macquarie dictionary

mySite = "https://www.macquariedictionary.com.au/features/word/search?word="

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub DictionaryFetchByLanguage()**

' Paul Beverley - Version 21.03.22

' Checks the current word in a language-related dictionary

' Based on an idea by Rob Worth

Set rng = Selection.Range.Duplicate

myLanguage = rng.LanguageID

If Len(rng.Text) < 3 Then rng.Expand wdWord

Select Case myLanguage

Case wdEnglishUS

mySite = "https://www.merriam-webster.com/dictionary/"

Case wdEnglishAUS

mySite = "https://www.macquariedictionary.com.au/features/word/search/"

Case wdGerman

mySite = "https://www.duden.de/suchen/dudenonline/"

Case wdFrench

mySite = "https://www.collinsdictionary.com/dictionary/french-english/"

Case Else

mySite = "https://www.lexico.com/definition/"

End Select

mySubject = Trim(rng.Text)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

Set newLink = ActiveDocument.Hyperlinks.Add(Anchor:=Selection.Range, \_

Address:=mySite & mySubject)

newLink.Follow

ActiveDocument.Undo

End Sub

**Sub GoogleTranslate()**

' Paul Beverley - Version 11.07.21

' Launches selected text to GoogleTranslate

myLanguage = "de"

myLanguage = "es"

mySite = "https://translate.google.co.uk/?oe=utf-8&hl=<LANG>&sl=auto&tl=<LANG>&text=<TEXT>&op=translate"

If Len(Selection) = 1 Then

Selection.Expand wdParagraph

Selection.MoveEnd , -1

End If

wd = Selection.Words(1)

If Application.CheckSpelling(wd, \_

MainDictionary:=Languages(wdEnglishUK).NameLocal) \_

= False And Application.CheckSpelling(wd, \_

MainDictionary:=Languages(wdEnglishUS).NameLocal) \_

= False Then myLanguage = "en"

If Selection.Words.Count > 1 Then

wd = Selection.Words(2)

If Application.CheckSpelling(wd, \_

MainDictionary:=Languages(wdEnglishUK).NameLocal) \_

= False And Application.CheckSpelling(wd, \_

MainDictionary:=Languages(wdEnglishUS).NameLocal) \_

= False Then myLanguage = "en"

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

myURL = Replace(mySite, "<LANG>", myLanguage)

myURL = Replace(myURL, "<TEXT>", mySubject)

Debug.Print myURL

ActiveDocument.FollowHyperlink Address:=myURL

Selection.Collapse wdCollapseEnd

End Sub

**Sub WikiFetchNL()**

' Paul Beverley - Version 08.08.20

' Launches selected text on WikipediaNL

mySite = "http://nl.wikipedia.org/wiki/Special:Search?search="

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleFetchNL()**

' Paul Beverley - Version 08.08.20

' Launches selected text on GoogleNL

mySite = "http://www.google.nl/search?q="

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleFetchQuotesNL()**

' Paul Beverley - Version 08.08.20

' Launches selected text - with quotes - on GoogleNL

mySite = "http://www.google.nl/search?q="

mySubject = Trim(Selection)

mySubject = Replace(mySubject, vbCr, "")

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = "%22" & mySubject & "%22"

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub ThesaurusFetchNL()**

' Paul Beverley - Version 08.08.20

' Launch selected text to https://synoniemen.net

mySite = "https://synoniemen.net/index.php?zoekterm="

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub DictionaryFetchNL()**

' Paul Beverley - Version 08.08.20

' Launches selected text to https://www.vandale.nl

mySite = "https://www.vandale.nl/gratis-woordenboek/nederlands/betekenis/"

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GroeneBoekjeFetch()**

' Paul Beverley - Version 08.08.20

' Launches selected text to https://woordenlijst.org

mySite = "https://woordenlijst.org/#/?q="

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleSSFetch()**

' Paul Beverley - Version 12.01.21

' Uses Google to search a single specific site

'mySite = "wordmacrotools.com"

'mySite = "archivepub.co.uk"

mySite = "uea.ac.uk"

myGoogle = "http://www.google.co.uk/search?q="

If Len(Selection) = 1 Then Selection.Expand wdWord

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

mySubject = mySubject & " site:" & mySite

ActiveDocument.FollowHyperlink Address:=myGoogle & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub CMOSFetch()**

' Paul Beverley - Version 28.08.22

' Launches selected text to CMS website

mySite = "https://www.chicagomanualofstyle.org/search.html?clause="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

End Sub

**Sub CMOSFetchAlt()**

' Paul Beverley - Version 28.08.22

' Launches selected text to CMS website.

mySite = "https://www.chicagomanualofstyle.org/home.html": numTabs = 12: myWait = 0.9

testMode = False

' testMode = True

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

Selection.Copy

ActiveDocument.FollowHyperlink Address:=mySite

If testMode = True Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

myTime = Timer

Do

Loop Until Timer > myTime + myWait

For i = 1 To numTabs

SendKeys "{TAB}"

Next i

SendKeys "^v"

SendKeys "{ENTER}"

End Sub

**Sub SomethingFetch()**

' Paul Beverley - Version 28.08.22

' Launches selected text to Xyz website.

mySite = ""

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

End Sub

**Sub SomethingFetchAlt()**

' Paul Beverley - Version 27.08.22

' Launches selected text to Xyz website.

' mySite = "https://www.merriam-webster.com/": numTabs = 0: myWait = 0.9

' mySite = "http://google.com/": numTabs = 0: myWait = 0.5

' mySite = "https://dictionary.com/": numTabs = 11: myWait = 2

' mySite = "http://thesaurus.com/browse/": numTabs = 11: myWait = 1.5

mySite = "https://www.macquariedictionary.com.au": numTabs = 0: myWait = 2.5

' mySite = "https://premium.oxforddictionaries.com/definition/english": numTabs = 99: myWait = 5

testMode = False

' testMode = True

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

Selection.Copy

ActiveDocument.FollowHyperlink Address:=mySite

If testMode = True Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

myTime = Timer

Do

Loop Until Timer > myTime + myWait

For i = 1 To numTabs

SendKeys "{TAB}"

Next i

SendKeys "^v"

SendKeys "{ENTER}"

End Sub

**Sub URLlauncher()**

' Paul Beverley - Version 14.01.21

' Launches successive URLs from the text

numberOfURLs = 10

highlightURL = True

myHighlight = wdGray25

acceptableChars = "/a-zA-Z0-9.,;:=&#\?\(\)\[\]\_+\-%" \_

& ChrW(8211) & ChrW(8212)

If Selection.Start = Selection.End Then

numberOfURLs = 1

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

Else

myInput = InputBox("How many?", "URL launcher", \_

Trim(Str(numberOfURLs)))

numberOfURLs = Val(myInput)

If numberOfURLs = 0 Then Exit Sub

Selection.Collapse wdCollapseEnd

End If

hereNow = Selection.End

linksTotal = ActiveDocument.Hyperlinks.Count

If linksTotal > 0 Then

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Selection.End = hereNow

Selection.Start = 0

Selection.Delete

linksTotal = ActiveDocument.Hyperlinks.Count

If linksTotal > 0 Then

For i = 1 To linksTotal

Set hy = ActiveDocument.Hyperlinks(i)

myURL = ActiveDocument.Hyperlinks(i).Address

hy.Range.Select

Selection.Font.Superscript = False

URLtext = Selection.Text

Selection.Collapse wdCollapseEnd

If myURL <> URLtext Then

Selection.TypeText " " & myURL & " "

End If

Next i

End If

ActiveDocument.Fields.Unlink

Selection.HomeKey Unit:=wdStory

Beep

End If

nowColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myHighlight

On Error GoTo ReportIt

myFind = "[hpstw]{3,5}:[" & acceptableChars & "]{1,}"

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = False

.Forward = True

.MatchWildcards = True

.Execute

End With

Selection.Collapse wdCollapseStart

i = 0

Do While i < numberOfURLs And Selection.Find.Found

i = i + 1

Selection.Find.Execute

Do While InStr(".,", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

myURL = Selection

If Left(myURL, 4) = "www." Then

myURL = "http://" & myURL

End If

ActiveDocument.FollowHyperlink Address:=myURL

Selection.Collapse wdCollapseEnd

If highlightURL Then

Selection.Range.HighlightColorIndex = myHighlight

End If

Selection.Collapse wdCollapseEnd

Loop

Options.DefaultHighlightColorIndex = nowColour

If numberOfURLs > 2 Then Beep

Exit Sub

ReportIt:

If Err.Number = 4198 Then

MsgBox ("URL not found: " & vbCr & vbCr & myURL)

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub URLlink()**

' Paul Beverley - Version 23.01.21

' Makes the URL/email at the cursor a clickable link

' Extra characters at the ends of a URL, NOT to be included

extraneousChars = ".,)(;[]:< " & ChrW(8211) & ChrW(8212) \_

& ChrW(8220) & ChrW(8221)

oldFind = Selection.Find.Text

Selection.Collapse wdCollapseStart

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13 " & ChrW(8212) & ChrW(8220) & "\[]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = False

.MatchWildcards = True

.Execute

End With

URLstart = Selection.Start + 1

With Selection.Find

.Text = "[^13 \]]"

.Wrap = wdFindContinue

.Forward = True

.MatchWildcards = True

.Execute

End With

Selection.MoveLeft , 1

Selection.Start = URLstart

Do While InStr(extraneousChars, Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Do While InStr(extraneousChars, Left(Selection.Text, 1)) > 0

Selection.MoveStart , 1

DoEvents

Loop

myAddress = Selection

If InStr(myAddress, "@") > 0 Then myAddress = "mailto:" & myAddress

ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, Address:=myAddress

Selection.Find.Text = oldFind

End Sub

**Sub HiLightON()**

**'** Paul Beverley - Version 01.01.10

**'** Adds highlight in currently selected colour

Selection.Range.HighlightColorIndex = Options.DefaultHighlightColorIndex

Selection.Start = Selection.End

End Sub

**Sub HiLightTurquoise()**

**'** Paul Beverley - Version 01.01.10

**'** Adds highlight in turquoise

Selection.Range.HighlightColorIndex = wdTurquoise

Selection.Start = Selection.End

End Sub

**Sub HiLightOFF()**

**'** Paul Beverley - Version 01.01.10

**'** Removes highlight (text colour) from selected text

Selection.Range.HighlightColorIndex = wdNoHighlight

' Selection.Range.Font.Color = wdColorAutomatic

Selection.Start = Selection.End

End Sub

**Sub HiLightOffALL()**

**'** Paul Beverley - Version 01.01.10

**'** Removes ALL highlights (text colour) from whole text

Selection.WholeStory

Selection.Range.HighlightColorIndex = wdNoHighlight

' Selection.Range.Font.Color = wdColorAutomatic

Selection.Start = Selection.End

End Sub

**Sub HiLightOffCurrentLine()**

**'** Paul Beverley - Version 22.01.10

**'** Removes highlight (text colour) from selected text or current line

' Switch off track changes

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

' If nothing selected, select the whole line

If Selection.Start = Selection.End Then

Selection.HomeKey Unit:=wdLine

Selection.MoveDown Unit:=wdLine, Count:=1, Extend:=wdExtend

End If

' Remove highlighting

Selection.Range.HighlightColorIndex = wdNoHighlight

' Remove text colouring

' Selection.Range.Font.Color = wdColorAutomatic

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub ClearHighlightAndColour()**

' Paul Beverley - Version 28.01.22

' Removes all highlighting, coloration and (optionally) underlining

doUnderlinesToo = True

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = wdNoHighlight

rng.Font.Color = wdColorAutomatic

If doUnderlinesToo = True Then rng.Font.Underline = False

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

rng.HighlightColorIndex = wdNoHighlight

rng.Font.Color = wdColorAutomatic

If doUnderlinesToo = True Then rng.Font.Underline = False

End If

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

rng.HighlightColorIndex = wdNoHighlight

rng.Font.Color = wdColorAutomatic

If doUnderlinesToo = True Then rng.Font.Underline = False

End If

End Sub

**Sub ColourToggle()**

**'** Paul Beverley - Version 05.05.10

**'** Red text on/off

' F11

nowColour = Selection.Font.Color

' If it’s red or a mix of red and black, switch to black

If nowColour = wdColorRed Or nowColour > 100 Then

Selection.Font.Color = wdColorAutomatic

Else

Selection.Font.Color = wdColorRed

End If

End Sub

**Sub HighlightPlus()**

' Paul Beverley - Version 22.02.24

' Adds highlight in a choice of colours

' Make one or other True:

doParagraph = False

doWord = True

Dim myCol(20)

myCol(0) = wdNoHighlight

myCol(1) = wdBrightGreen

myCol(2) = wdYellow

myCol(3) = wdTurquoise

myCol(4) = wdPink

myCol(5) = wdRed

myCol(6) = wdGray50

myCol(7) = wdGray25

myColTotal = 7

' Remember track changes status, then switch it off

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Dim v As Variable: Dim nowColour As Long

Dim wasStart As Long: Dim wasEnd As Long

numVars = 0

For Each v In ActiveDocument.Variables

If v.Name = "colNum" Then numVars = numVars + 1

If v.Name = "selEnd" Then numVars = numVars + 1

If v.Name = "selStart" Then numVars = numVars + 1

Next v

If numVars = 1 Or numVars = 2 Then

numVars = 0

For i = ActiveDocument.Variables.Count To 1 Step -1

ActiveDocument.Variables(i).Delete

Next i

End If

If numVars = 0 Then

ActiveDocument.Variables.Add "selStart", 0

ActiveDocument.Variables.Add "selEnd", 0

ActiveDocument.Variables.Add "colNum", 0

Else

wasStart = ActiveDocument.Variables("selStart")

wasEnd = ActiveDocument.Variables("selEnd")

wasCol = ActiveDocument.Variables("colNum")

End If

' If no text is selected ...

Set rng = Selection.Range.Duplicate

If rng.Start = rng.End Then

rng.MoveEnd , 1

If rng.Text = vbCr Then

rng.MoveStart , -1

rng.Collapse wdCollapseStart

Else

rng.MoveEnd , -1

End If

' Cursor outside the previous range? So start from scratch

If rng.Start < wasStart Or rng.End > wasEnd Then

' If the cursor is outside the previous range

If doParagraph = True Then

rng.Expand wdParagraph

Else

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

End If

ActiveDocument.Variables("selStart") = rng.Start

ActiveDocument.Variables("selEnd") = rng.End

' Debug.Print rng.Start, rng.End

nowCol = 1

Else

' Otherwise check the highlight colour

rng.Start = wasStart

rng.End = wasStart + 1

nowColour = rng.HighlightColorIndex

If nowColour <> myCol(wasCol) Then

' colour has changed, so go back to colour 1

nowCol = 1

Else

' go to next colour

nowCol = wasCol + 1

If nowCol > myColTotal Then nowCol = 0

End If

rng.End = wasEnd

End If

Else: ' if some text is selected ...

' Record current selection range

ActiveDocument.Variables("selStart") = rng.Start

ActiveDocument.Variables("selEnd") = rng.End

nowCol = 1

End If

ActiveDocument.Variables("colNum") = nowCol

rng.HighlightColorIndex = myCol(nowCol)

rng.Collapse wdCollapseStart

rng.Select

theEnd:

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub HighlightMinus()**

' Paul Beverley - Version 22.02.24

' Adds highlight in a choice of colours

' Make one or other True:

doParagraph = False

doWord = True

Dim myCol(20)

myCol(0) = wdNoHighlight

myCol(1) = wdBrightGreen

myCol(2) = wdYellow

myCol(3) = wdTurquoise

myCol(4) = wdPink

myCol(5) = wdRed

myCol(6) = wdGray50

myCol(7) = wdGray25

myColTotal = 7

' Remember track changes status, then switch it off

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Dim v As Variable: Dim nowColour As Long

Dim wasStart As Long: Dim wasEnd As Long

numVars = 0

For Each v In ActiveDocument.Variables

If v.Name = "colNum" Then numVars = numVars + 1

If v.Name = "selEnd" Then numVars = numVars + 1

If v.Name = "selStart" Then numVars = numVars + 1

Next v

If numVars = 1 Or numVars = 2 Then

numVars = 0

For i = ActiveDocument.Variables.Count To 1 Step -1

ActiveDocument.Variables(i).Delete

Next i

End If

If numVars = 0 Then

ActiveDocument.Variables.Add "selStart", 0

ActiveDocument.Variables.Add "selEnd", 0

ActiveDocument.Variables.Add "colNum", 0

Else

wasStart = ActiveDocument.Variables("selStart")

wasEnd = ActiveDocument.Variables("selEnd")

wasCol = ActiveDocument.Variables("colNum")

End If

' If no text is selected ...

Set rng = Selection.Range.Duplicate

If rng.Start = rng.End Then

rng.MoveEnd , 1

If rng.Text = vbCr Then

rng.MoveStart , -1

rng.Collapse wdCollapseStart

Else

rng.MoveEnd , -1

End If

' Cursor outside the previous range? So start from scratch

If rng.Start < wasStart Or rng.End > wasEnd Then

' If the cursor is outside the previous range

If doParagraph = True Then

rng.Expand wdParagraph

Else

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

End If

ActiveDocument.Variables("selStart") = rng.Start

ActiveDocument.Variables("selEnd") = rng.End

' Debug.Print rng.Start, rng.End

nowCol = 0

Else

' Otherwise check the highlight colour

rng.Start = wasStart

rng.End = wasStart + 1

nowColour = rng.HighlightColorIndex

If nowColour <> myCol(wasCol) Then

' colour has changed, so go back to colour 7

nowCol = 7

Else

' go to next colour

nowCol = wasCol - 1

If nowCol < 0 Then nowCol = myColTotal

End If

rng.End = wasEnd

End If

Else: ' if some text is selected ...

' Record current selection range

ActiveDocument.Variables("selStart") = rng.Start

ActiveDocument.Variables("selEnd") = rng.End

nowCol = 0

End If

ActiveDocument.Variables("colNum") = nowCol

rng.HighlightColorIndex = myCol(nowCol)

rng.Collapse wdCollapseStart

rng.Select

theEnd:

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub ColourPlus()**

**'** Paul Beverley - Version 15.01.24

' Adds font colour in a choice of colours

Dim myCol(10)

myCol(0) = wdColorBlack

myCol(1) = wdColorBlue

myCol(2) = wdColorRed

myCol(3) = wdColorPink

myCol(4) = wdColorBrightGreen

myCol(5) = wdColorGray50

myCol(6) = wdColorGray25

myColTotal = 6

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Dim v As Variable, nowColour As Long

varsExist = False

For Each v In ActiveDocument.Variables

If v.Name = "selStart" Then varsExist = True: Exit For

Next v

If varsExist = False Then

ActiveDocument.Variables.Add "selStart", 0

ActiveDocument.Variables.Add "selEnd", 0

ActiveDocument.Variables.Add "colNum", 0

End If

wasStart = ActiveDocument.Variables("selStart")

wasEnd = ActiveDocument.Variables("selEnd")

wasCol = ActiveDocument.Variables("colNum")

' If no text is selected ...

startAgain:

If Selection.Start = Selection.End Then

If Selection.Text = vbCr Then Selection.MoveLeft , 1

' If the cursor is outside the area, select the word and start again

If Selection.Start < wasStart Or Selection.End > wasEnd Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

DoEvents

GoTo startAgain

End If

' Otherwise check the font colour

Selection.Start = wasStart

Selection.End = wasStart + 1

nowColour = Selection.Font.Color

If nowColour <> myCol(wasCol) Then

' colour has changed, so go back to first colour

nowCol = 1

Else

' go to next colour

nowCol = wasCol + 1

If nowCol > myColTotal Then nowCol = 0

End If

Selection.End = wasEnd

Else: ' if some text is selected ...

' Record current selection range

ActiveDocument.Variables("selStart") = Selection.Start

ActiveDocument.Variables("selEnd") = Selection.End

nowCol = 1

End If

ActiveDocument.Variables("colNum") = nowCol

Selection.Font.Color = myCol(nowCol)

Selection.End = Selection.Start

theEnd:

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub ColourMinus()**

' Paul Beverley - Version 16.03.21

' Removes or adds font colour in a choice of colours

removeUnderline = True

Dim myCol(10)

myCol(0) = wdColorBlack

myCol(1) = wdColorBlue

myCol(2) = wdColorRed

myCol(3) = wdColorPink

myCol(4) = wdColorSkyBlue

myCol(5) = wdColorBrightGreen

myCol(6) = wdColorGray50

myCol(7) = wdColorGray25

myColTotal = 7

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = ActiveDocument.Content

If removeUnderline And Selection = rng Then Selection.Font.Underline = False

Dim v As Variable, nowColour As Long

varsExist = False

For Each v In ActiveDocument.Variables

If v.Name = "selStart" Then varsExist = True: Exit For

Next v

If varsExist = False Then

ActiveDocument.Variables.Add "selStart", 0

ActiveDocument.Variables.Add "selEnd", 0

ActiveDocument.Variables.Add "colNum", 0

End If

wasStart = ActiveDocument.Variables("selStart")

wasEnd = ActiveDocument.Variables("selEnd")

wasCol = ActiveDocument.Variables("colNum")

' If no text is selected ...

If Selection.Start = Selection.End Then

If Selection.Text = vbCr Then Selection.MoveLeft , 1

' If the cursor is outside the area, give up

If Selection.Start < wasStart Or Selection.End > wasEnd Then GoTo LineClear

' Otherwise check the font colour

Selection.Start = wasStart

Selection.End = wasStart + 1

nowColour = Selection.Font.Color

If nowColour <> myCol(wasCol) Then

' colour changed, so go to last colour on list

nowCol = 0

Else

' go to next colour

nowCol = wasCol - 1

If nowCol < 0 Then nowCol = myColTotal

End If

Selection.End = wasEnd

Else: ' if some text is selected ...

' Record current selection range

ActiveDocument.Variables("selStart") = Selection.Start

ActiveDocument.Variables("selEnd") = Selection.End

If Selection.Start = 0 Then ActiveDocument.Variables("selStart") \_

= Selection.End

' go to last colour on list

nowCol = 0

End If

ActiveDocument.Variables("colNum") = nowCol

Selection.Font.Color = myCol(nowCol)

Selection.End = Selection.Start

ActiveDocument.TrackRevisions = myTrack

Exit Sub

LineClear:

Selection.HomeKey Unit:=wdLine

clearStart = Selection.Start

Selection.MoveDown Unit:=wdLine, count:=1

Set rng = Selection.Range

rng.Start = clearStart

rng.Font.Color = wdColorAutomatic

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub UnHighlightAndColour()**

' Paul Beverley - Version 04.11.17

' Removes all highlight/coloration of the current colour

Set wasSelected = Selection.range.Duplicate

Selection.Collapse wdCollapseStart

Selection.MoveEnd , 1

myColour = Selection.range.Font.ColorIndex

If myColour = 1 Then myColour = 0

myHighlight = Selection.range.HighlightColorIndex

styleColour = Selection.range.Style.Font.ColorIndex

If myColour + myHighlight = 0 Then

MsgBox "Please place the cursor in the colour/highlight to be removed."

Exit Sub

End If

If myColour > 0 And myColour = styleColour Then

MsgBox "Sorry, this macro doesn't work with colours in styles."

Exit Sub

End If

Set rng = ActiveDocument.Content

mixedColour = 9999999

If myHighlight > 0 Then

For Each myPara In rng.Paragraphs

col = myPara.range.HighlightColorIndex

If col <> mixedColour Then

If col = myHighlight Then myPara.range.HighlightColorIndex = wdNoHighlight

Else

For Each wd In myPara.range.Words

col = wd.HighlightColorIndex

If col <> mixedColour Then

If col = myHighlight Then wd.HighlightColorIndex = wdNoHighlight

Else

For Each ch In wd.Characters

col = ch.HighlightColorIndex

If col <> mixedColour Then

If col = myHighlight Then \_

ch.HighlightColorIndex = wdNoHighlight

End If

DoEvents

Next ch

End If

DoEvents

Next wd

End If

DoEvents

j = j + 1

If j Mod 5 = 0 Then

myPara.range.Select

Selection.Collapse wdCollapseEnd

End If

Next myPara

Else

For Each myPara In rng.Paragraphs

col = myPara.range.Font.ColorIndex

If col <> mixedColour Then

If col = myColour Then myPara.range.Font.ColorIndex = wdNoHighlight

Else

For Each wd In myPara.range.Words

col = wd.Font.ColorIndex

If col <> mixedColour Then

If col = myColour Then wd.Font.ColorIndex = wdNoHighlight

Else

For Each ch In wd.Characters

col = ch.Font.ColorIndex

If col <> mixedColour Then

If col = myColour Then ch.Font.ColorIndex = wdNoHighlight

End If

DoEvents

Next ch

End If

DoEvents

Next wd

End If

DoEvents

Next myPara

End If

Application.ScreenUpdating = False

Selection.EndKey Unit:=wdStory

wasSelected.Select

Application.ScreenUpdating = True

ActiveDocument.ActiveWindow.SmallScroll down:=1

Beep

End Sub

**Sub HighlightAllOff()**

' Paul Beverley - Version 02.04.13 (minor mod 07.12.15)

' Remove all highlighting, including in boxes

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = 0

If ActiveDocument.Shapes.Count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

shp.TextFrame.TextRange.HighlightColorIndex = 0

End If

End If

Next

Beep

End If

End Sub

**Sub UnHighlightExcept()**

**'** Paul Beverley - Version 11.01.11

' Remove all highlights except one/two chosen colours

keepColour1 = wdYellow

' And another colour as well?

' keepColour2 = wdBrightGreen

keepColour2 = wdTurquoise

' keepColour2 = wdRed

' keepColour2 = 0

Set rng = ActiveDocument.Content

theEnd = rng.End

gotOne = False

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

rng.End = 0

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

gotOne = rng.Find.Found

If gotOne = True Then

foundColour = rng.HighlightColorIndex

If foundColour > 99 Then

' Mixed colours of highlighting

colEnd = rng.End

Do

rng.End = rng.Start + 1

foundColour = rng.HighlightColorIndex

If (foundColour <> keepColour1) And (foundColour <> keepColour2) Then

rng.HighlightColorIndex = 0

End If

If foundColour > 99 Then rng.HighlightColorIndex = 0

rng.Start = rng.End

Loop Until rng.End = colEnd

Else

If (foundColour <> keepColour1) And (foundColour <> keepColour2) Then

rng.HighlightColorIndex = 0

End If

End If

StatusBar = "On large files, this may take some time. " \_

& Str(theEnd - rng.End)

rng.Start = rng.End

End If

Loop Until gotOne = False

' Now remove all the 'Not formatted' track revisions

theEnd = ActiveDocument.Content.End

started = False

For Each rev In ActiveDocument.Range.Revisions

Set rng = rev.Range

myType = rev.FormatDescription

If myType = "Formatted: Not Highlight" Then rng.Revisions.AcceptAll

StatusBar = "Getting rid of 'Formatted: Not Highlight'..." \_

& Str(theEnd - rng.End)

Next rev

ActiveDocument.TrackRevisions = nowTrack

StatusBar = " Finished!!!!!!!!!!!!!!!"

Selection.HomeKey Unit:=wdStory

End Sub

**Sub SelectiveUnColourUnHighlight()**

' Paul Beverley - Version 04.11.17

' Removes highlighting + removes coloration, but only on non-program text

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = wdNoHighlight

For Each myPara In ActiveDocument.Paragraphs

If myPara.range.Characters(1).Font.Name <> "Courier New" Then \_

myPara.range.Font.Color = wdColorAutomatic

Next myPara

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub ShowFixedSpaces()**

**'** Paul Beverley - Version 09.02.10

' Fixed spaces visible

myColour = wdGray25

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Text = "^s"

.Highlight = True

.Wrap = wdFindContinue

.Execute

End With

If rng.Find.Found = False Then

OldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^s"

.Replacement.Text = "^s"

.Replacement.Highlight = True

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = OldColour

Else

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^s"

.Replacement.Text = "^s"

.Replacement.Highlight = False

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

End If

End Sub

**Sub HighlightSame()**

' Paul Beverley - Version 16.01.21

' Highlight all occurrences of this text in this colour

nonTextColour = wdGray25

textColour = wdBrightGreen

' colour for thin spaces

otherColour = wdYellow

' Preserve TC status and existing highlight colour

oldColour = Options.DefaultHighlightColorIndex

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Dim v As Variable, nowColour As Long

varsExist = False

For Each v In ActiveDocument.Variables

If v.Name = "selStart" Then varsExist = True: Exit For

Next v

If varsExist Then

wasStart = ActiveDocument.Variables("selStart")

wasEnd = ActiveDocument.Variables("selEnd")

If Selection.Start > wasStart - 1 And Selection.End < wasEnd + 1 Then

Selection.Start = wasStart

Selection.End = wasEnd

End If

End If

Set rng = ActiveDocument.Content

rng.Start = Selection.Start - 1

rng.End = Selection.Start

If Selection.End = Selection.Start Then

If AscW(Selection) = 8201 Then nonTextColour = otherColour

partWord = False

Selection.MoveEnd , 1

myChar = Selection

nonText = (UCase(myChar) = LCase(myChar))

If nonText = True Then

nowColour = Selection.Range.HighlightColorIndex

If nowColour = wdNoHighlight Then

Options.DefaultHighlightColorIndex = nonTextColour

Else

Options.DefaultHighlightColorIndex = wdNoHighlight

End If

Else

If UCase(rng) <> LCase(rng) And rng.HighlightColorIndex > 0 Then

Options.DefaultHighlightColorIndex = wdNoHighlight

Else

nowColour = Selection.Range.HighlightColorIndex

If nowColour = wdNoHighlight Then

Options.DefaultHighlightColorIndex = textColour

Else

Options.DefaultHighlightColorIndex = nowColour

End If

End If

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

Else

partWord = True

nowColour = Selection.Range.HighlightColorIndex

' If nowColour = 0 Then nowColour = nontextColour

If nowColour > 1000 Then

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

rng.End = Selection.Start + 1

nowColour = rng.HighlightColorIndex

End If

Options.DefaultHighlightColorIndex = nowColour

End If

findText = Selection

Select Case Asc(findText)

Case 9: findText = "^t"

Case 30: findText = "^~": partWord = True ' non-breaking hyphen

End Select

Selection.Collapse wdCollapseStart

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = findText

.MatchCase = True

.Forward = True

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Wrap = wdFindContinue

If partWord = False And nonText = False Then .MatchWholeWord = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Restore to original state

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = oldColour

ActiveDocument.TrackRevisions = nowTrack

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchCase = False

.MatchWholeWord = False

.Replacement.Highlight = False

End With

If Selection.End = Selection.Start Then

myChar = Selection

If UCase(myChar) <> LCase(myChar) Then

Selection.Move , 1

End If

End If

End Sub

**Sub FontColourSame()**

' Paul Beverley - Version 18.06.21

' Colours all occurrences of this text in this colour

textColour = wdColorBlue

' Preserve TC status and existing highlight colour

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

searchChars = " " & ChrW(8217)

Dim v As Variable, nowColour As Long

nowColour = Selection.Font.Color

varsExist = False

For Each v In ActiveDocument.Variables

If v.Name = "selStart" Then varsExist = True: Exit For

Next v

If varsExist Then

wasStart = ActiveDocument.Variables("selStart")

wasEnd = ActiveDocument.Variables("selEnd")

If Selection.Start > wasStart - 1 And Selection.End < wasEnd + 1 Then

Selection.Start = wasStart

Selection.End = wasEnd

End If

End If

Set rng = ActiveDocument.Content

rng.Start = Selection.Start - 1

rng.End = Selection.Start

If Selection.End = Selection.Start Then

partWord = False

Selection.MoveEnd , 1

myChar = Selection

nonText = (UCase(myChar) = LCase(myChar))

If nonText = True Then

nowColour = Selection.Font.Color

If nowColour = wdNoHighlight Then

newColour = nonTextColour

Else

newColour = wdNoHighlight

End If

Else

If UCase(rng) <> LCase(rng) And rng.HighlightColorIndex > 0 Then

newColour = wdNoHighlight

Else

nowColour = Selection.Font.Color

If nowColour = wdNoHighlight Then

newColour = textColour

Else

newColour = textColour

End If

End If

Selection.Expand wdWord

Do While InStr(searchChars, Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

Else

partWord = True

nowColour = Selection.Font.Color

' If nowColour = 0 Then nowColour = nontextColour

If nowColour > 1000 Then

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

rng.End = Selection.Start + 1

nowColour = rng.Font.Color

End If

newColour = nowColour

End If

findText = Selection

Select Case Asc(findText)

Case 9: findText = "^t"

Case 30: findText = "^~": partWord = True ' non-breaking hyphen

End Select

Selection.Collapse wdCollapseStart

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = findText

.MatchCase = True

.Forward = True

.Replacement.Text = "^&"

.Replacement.Font.Color = newColour

.Wrap = wdFindContinue

If partWord = False And nonText = False Then .MatchWholeWord = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Restore to original state

oldColour = newColour

newColour = oldColour

ActiveDocument.TrackRevisions = nowTrack

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchCase = False

.MatchWholeWord = False

.Replacement.Highlight = False

End With

If Selection.End = Selection.Start Then

myChar = Selection

If UCase(myChar) <> LCase(myChar) Then

Selection.Move , 1

End If

End If

End Sub

**Sub HighlightLister()**

**'** Paul Beverley - Version 26.05.11

' List all the highlight colours used

allHighs = ""

Selection.WholeStory

Selection.Copy

Documents.Add

Selection.Paste

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Highlight = False

.Text = ""

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

wasCol = 0

For Each ch In ActiveDocument.Characters

thisCol = ch.HighlightColorIndex

If thisCol <> wasCol Then

Select Case thisCol

Case 0: 'Do nowt

Case wdYellow: col = "Yellow"

Case wdBrightGreen: col = "BrightGreen"

Case wdGreen: col = " Green"

Case wdPink: col = "Pink"

Case wdRed: col = "Red"

Case wdBlue: col = "Blue"

Case wdGray25: col = "Gray25"

Case wdGray50: col = "Gray50"

Case wdTurquoise: col = "Turquoise"

Case wdTeal: col = "Teal"

Case wdDarkBlue: col = "DarkBlue"

Case wdDarkYellow: col = "DarkYellow"

Case wdDarkRed: col = "DarkRed"

Case wdViolet: col = "Violet"

Case Else

ch.Select

col = "A colour not on the list!"

End Select

If InStr(allHighs, col) = 0 Then allHighs = allHighs & col & ","

End If

wasCol = thisCol

i = i + 1

If i Mod 50 = 0 Then StatusBar = "Checked: " & Str(i)

Next ch

ActiveDocument.Close SaveChanges:=False

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:=Replace(allHighs, ",", vbCrLf)

Selection.Start = 0

Selection.Style = wdStyleNormal

Selection.Sort SortOrder:=wdSortOrderAscending

Selection.End = 0

End Sub

**Sub HighlightListerDeLuxe()**

**'** Paul Beverley - Version 27.05.11

' List all the highlight colours used

Dim gotCol(16) As Boolean

mixCol = 9999999

' Copy the whole text

Selection.WholeStory

Selection.Copy

Documents.Add

Selection.Paste

' Delete unhighlighted text

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Highlight = False

.Text = ""

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Find which colours are used

For Each wd In rng.Words

thisCol = wd.HighlightColorIndex

If thisCol < mixCol Then

gotCol(thisCol) = True

Else

For Each ch In wd.Characters

thisCol = ch.HighlightColorIndex

gotCol(thisCol) = True

Next ch

End If

Next wd

ActiveDocument.Close SaveChanges:=False

' List colours used and not used

' avoiding 1 = white and 8 = black

Selection.HomeKey Unit:=wdStory

For i = 2 To 16

' Make the used colour stand out

If gotCol(i) = True Then Selection.TypeText Text:=vbTab & vbTab & "aaa"

Select Case i

Case wdWhite: col = "White"

Case wdBlack: col = "Black"

Case wdYellow: col = "Yellow"

Case wdBrightGreen: col = "BrightGreen"

Case wdGreen: col = "Green"

Case wdPink: col = "Pink"

Case wdRed: col = "Red"

Case wdBlue: col = "Blue"

Case wdGray25: col = "Gray25"

Case wdGray50: col = "Gray50"

Case wdTurquoise: col = "Turquoise"

Case wdTeal: col = "Teal"

Case wdDarkBlue: col = "DarkBlue"

Case wdDarkYellow: col = "DarkYellow"

Case wdDarkRed: col = "DarkRed"

Case wdViolet: col = "Violet"

End Select

If i <> 8 Then Selection.InsertBefore Text:=col & vbCrLf

Selection.Range.HighlightColorIndex = i

Selection.Start = Selection.End

Next i

Selection.Start = 0

Selection.Style = wdStyleNormal

Selection.Sort SortOrder:=wdSortOrderAscending

' Remove the 'used' markers

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Highlight = False

.Text = "aaa"

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

End Sub

**Sub** **BackgroundColourOff()**

' Paul Beverley - Version 11.04.20

' Removes background colour

If InStr(Selection, Chr(13)) > 0 Then

For Each myPara In Selection.Paragraphs

Set rng = myPara.Range

rng.End = rng.End - 1

If rng.Information(wdWithInTable) = False Then

rng.Shading.BackgroundPatternColor = wdColorAutomatic

End If

DoEvents

Next myPara

Else

Selection.Shading.BackgroundPatternColor = wdColorAutomatic

End If

End Sub

**Sub** **BackgroundColourAllOff()**

' Paul Beverley - Version 11.04.20

' Removes all background colours

Set rng = ActiveDocument.Content

rng.Shading.BackgroundPatternColor = wdColorAutomatic

With rng.Font.Shading

.Texture = wdTextureNone

.ForegroundPatternColor = wdColorAutomatic

.BackgroundPatternColor = wdColorAutomatic

End With

rng.Shading.BackgroundPatternColor = wdColorAutomatic

ActiveDocument.Background.Fill.Visible = msoFalse

End Sub

**Sub PunctuationFormatChecker()**

**'** Paul Beverley - Version 07.03.24

' Variously highlight italic/roman punctuation

markAllItalicCommas = True

markAllItalicCommas = False

italColour = wdGray25

markAllBoldPeriods = True

boldPeriodColour = wdTurquoise

markBoldHeadwordColons = True

boldColonColour = wdTurquoise

markRomanHeadwordColons = True

romanColonColour = wdRed

' Colour for all the rest:

mainPunctuationColour = wdBrightGreen

markCommas = True

markPeriods = True

markFullStops = True

markColons = True

markSemicolons = True

markParens = True

markQMs = True

markEMs = True

markSquBkts = True

markSingleQuotes = True

markDoubleQuotes = True

p = ""

If markCommas = True Then p = p & ","

If markColons = True Then p = p & "\:"

If markPeriods = True Or markFullStops = True Then p = p & "."

If markSemicolons = True Then p = p & "\;"

If markParens = True Then p = p & "\(\)"

If markQMs = True Then p = p & "\?"

If markEMs = True Then p = p & "\!"

If markSquBkts = True Then p = p & "\[\]"

If markSingleQuotes = True Then p = p & "'" & ChrW(8216) & ChrW(8217)

If markDoubleQuotes = True Then p = p & """" & ChrW(8220) & ChrW(8221)

myFind = "[" & p & "]"

If markAllItalicCommas = True Then

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = italColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ","

.Font.Italic = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

End If

If markAllBoldPeriods = True Then

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = boldPeriodColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "."

.Font.Bold = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

End If

' Check italic punctuation

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Font.Italic = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

allNumbers = "0123456789"

Do While rng.Find.Found = True

rng2.Start = rng.Start

rng2.End = rng.Start

isAlpha = False

isNumber = False

Do

rng2.MoveStart , -1

isAlpha = UCase(Left(rng2, 1)) <> LCase(Left(rng2, 1))

isNumber = (InStr(allNumbers, Left(rng2, 1)) > 0)

Loop Until isAlpha Or isNumber

markStart = rng2.Start

preItalic = rng2.Font.Italic

preBold = rng2.Font.Bold

rng2.Start = rng.Start + 1

rng2.End = rng.Start + 1

isAlpha = False

isNumber = False

Do

rng2.MoveEnd , 1

isAlpha = UCase(Right(rng2, 1)) <> LCase(Right(rng2, 1))

isNumber = (InStr(allNumbers, Right(rng2, 1)) > 0)

Loop Until isAlpha Or isNumber

postItalic = rng2.Font.Italic

postBold = rng2.Font.Bold

rng2.Start = markStart

doHighlight = False

If preItalic = Not (postItalic) Then

doHighlight = True

End If

If InStr(rng2, Chr(13)) > 0 Then doHighlight = False

If doHighlight = True Then

rng2.HighlightColorIndex = mainPunctuationColour

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' Check bold punctuation

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Font.Bold = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

allNumbers = "0123456789"

Do While rng.Find.Found = True

rng2.Start = rng.Start

rng2.End = rng.Start

isAlpha = False

isNumber = False

Do

rng2.MoveStart , -1

isAlpha = UCase(Left(rng2, 1)) <> LCase(Left(rng2, 1))

isNumber = (InStr(allNumbers, Left(rng2, 1)) > 0)

Loop Until isAlpha Or isNumber

markStart = rng2.Start

preBold = rng2.Font.Bold

preBold = rng2.Font.Bold

rng2.Start = rng.Start + 1

rng2.End = rng.Start + 1

isAlpha = False

isNumber = False

Do

rng2.MoveEnd , 1

isAlpha = UCase(Right(rng2, 1)) <> LCase(Right(rng2, 1))

isNumber = (InStr(allNumbers, Right(rng2, 1)) > 0)

DoEvents

Loop Until isAlpha Or isNumber

postBold = rng2.Font.Bold

postBold = rng2.Font.Bold

rng2.Start = markStart

doHighlight = False

If preBold = Not (postBold) Then

doHighlight = True

End If

If InStr(rng2, Chr(13)) > 0 Then doHighlight = False

If doHighlight = True Then

rng2.HighlightColorIndex = mainPunctuationColour

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

myMsg = "Finished!"

' Check headword colons

If markBoldHeadwordColons Or markRomanHeadwordColons Then

For Each myPar In ActiveDocument.Paragraphs

i = myPar.Range.Words.Count

Set rng = myPar.Range.Words(1)

firstBold = rng.Font.Bold

Set rng = myPar.Range.Words(i)

lastBold = rng.Font.Bold

If firstBold And Not (lastBold) Then

colonPos = InStr(myPar, ":")

If colonPos > 0 And colonPos < Len(myPar) - 6 Then

Set rng = myPar.Range.Characters(colonPos)

colonBold = rng.Font.Bold

Set rng2 = myPar.Range.Characters(colonPos - 1)

previousBold = rng2.Font.Bold

Set rng2 = myPar.Range.Characters(colonPos + 3)

followingBold = rng2.Font.Bold

rng.MoveStart , -1

rng.MoveEnd , 2

If colonBold And previousBold Then

If markBoldHeadwordColons Then

If Not followingBold Then rng.HighlightColorIndex = boldColonColour

Else

If Not followingBold Then rng.HighlightColorIndex = wdNoHighlight

End If

Else

If markRomanHeadwordColons And previousBold Then

If Not followingBold Then rng.HighlightColorIndex = romanColonColour

End If

End If

End If

End If

DoEvents

Next myPar

' Prepare prompt

Select Case boldColonColour

Case wdYellow: boldColour = "Yellow"

Case wdRed: boldColour = "Red"

Case wdTurquoise: boldColour = "Turquoise"

Case wdBrightGreen: boldColour = "Bright green"

Case wdGreen: boldColour = "Green"

Case wdGray25: boldColour = "Light grey"

Case wdGray50: boldColour = "Dark grey"

Case wdDarkYellow: boldColour = "Dark yellow"

Case wdPink: boldColour = "Pink"

Case Else: boldColour = "Unknown colour"

End Select

Select Case romanColonColour

Case wdYellow: romanColour = "Yellow"

Case wdRed: romanColour = "Red"

Case wdTurquoise: romanColour = "Turquoise"

Case wdBrightGreen: romanColour = "Bright green"

Case wdGreen: romanColour = "Green"

Case wdGray25: romanColour = "Light grey"

Case wdGray50: romanColour = "Dark grey"

Case wdDarkYellow: romanColour = "Dark yellow"

Case wdPink: romanColour = "Pink"

Case Else: romanColour = "Unknown colour"

End Select

myMsg = myMsg & vbCr & vbCr

If markBoldHeadwordColons Then

myMsg = myMsg & boldColour & " = bold " & vbCr

End If

If markRomanHeadwordColons Then

myMsg = myMsg & romanColour & " = roman " & vbCr

End If

End If

Beep

MsgBox (myMsg)

End Sub

**Sub HighlightOddPunctuationFormat()**

' Paul Beverley - Version 13.12.16

' Highlights oddly formatted punctuation marks

checkColons = True

myColour0 = wdBrightGreen

wantColonBold = False

wantColonItalic = False

checkCommas = True

myColour1 = wdBrightGreen

checkFontSize = True

myColour2 = wdYellow

checkFontName = True

myColour3 = wdTurquoise

highlightItalicCommasAnyway = True

myColour4 = wdGray25

Dim b(4) As Boolean

Dim i(4) As Boolean

Dim n(4) As String

Dim char(4) As String

myJump = 5

If checkColons = True Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ":"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myStart = rng.Start - 1

For j = 1 To 4

rng.Start = myStart + j - 1

rng.End = myStart + j

i(j) = rng.Italic

b(j) = rng.Bold

n(j) = rng.Font.Name

char(j) = rng.Text

Next j

rng.Start = myStart

rng.End = myStart + 4

If checkFontSize = True And rng.Font.Size > 999 Then

rng.HighlightColorIndex = myColour2

End If

If checkFontName = True Then

If n(1) <> n(2) Or n(2) <> n(4) Then \_

rng.HighlightColorIndex = myColour3

End If

'number of characters to be highlighted

m = 0

If wantColonBold Then

If b(1) And Not (b(2)) Then m = m + 1

' If the space is bold

If b(1) And b(2) And b(3) And Not (b(4)) \_

And char(3) <> vbCr Then

m = m + 1

If b(1) Then myStart = myStart + 1

End If

Else

' Want a roman colon, i.e. b(2) should be False

If b(1) And b(2) And (Not (b(3)) Or Not (b(4))) Then m = m + 1

End If

If wantColonItalic Then

If i(1) And Not (i(2)) Then m = m + 1

Else

' Want an italic colon, i.e. i(2) should be False

If i(1) And i(2) And (Not (i(3)) Or Not (i(4))) Then m = m + 1

End If

If m > 0 Then

rng.Start = myStart + 1

rng.End = myStart + m + 1

rng.HighlightColorIndex = myColour0

myCount = myCount + 1

If myCount Mod myJump = 1 Then rng.Select

End If

' Find the next one

rng.Start = myStart + 2

rng.End = myStart + 2

rng.Find.Execute

Loop

End If

If checkCommas = True Then

If highlightItalicCommasAnyway = True Then

Set rng = ActiveDocument.Content

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour4

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ","

.Font.Italic = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

End If

' Now check commas individually

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ","

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myStart = rng.Start - 1

For j = 1 To 4

rng.Start = myStart + j - 1

rng.End = myStart + j

i(j) = rng.Italic

b(j) = rng.Bold

n(j) = rng.Font.Name

Next j

rng.Start = myStart - 10

rng.End = myStart + 10

localItalic = rng.Font.Italic

rng.Start = myStart

rng.End = myStart + 4

If checkFontSize = True And rng.Font.Size > 999 Then

rng.HighlightColorIndex = myColour2

End If

If checkFontName = True Then

If n(1) <> n(2) Or n(2) <> n(4) Then \_

rng.HighlightColorIndex = myColour3

End If

' number of characters to be highlighted

m = 0

If i(1) And i(2) And Not (i(3)) Then m = m + 1

If i(1) And i(2) And Not (i(4)) Then m = m + 1

If i(2) And localItalic > 99 Then m = m + 1

If b(1) And b(2) And Not (b(3)) Then m = m + 1

If b(1) And b(2) And Not (b(4)) Then m = m + 1

If m > 0 Then

rng.Start = myStart + 1

rng.End = myStart + m + 1

rng.HighlightColorIndex = myColour0

myCount = myCount + 1

If myCount Mod myJump = 1 Then rng.Select

End If

' Find the next one

rng.Start = myStart + 2

rng.End = myStart + 2

rng.Find.Execute

Loop

End If

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub RomanPunctuationHighlight()**

**'** Paul Beverley - Version 13.06.23

' Finds roman punctuation that follows italic text

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = wdFindStop

.Execute

End With

Do While rng.Find.Found = True

rng.Collapse wdCollapseEnd

rng.MoveEnd , 1

If LCase(rng) = UCase(rng) And rng <> " " Then

rng.HighlightColorIndex = wdGray25

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

End Sub

**Sub HighlightWordList()**

**'** Paul Beverley - Version 19.12.24

' Highlights (and/or colours) all the words/phrases given in a list

doWholeWordsOnly = True

doMatchCase = False

Set rng = ActiveDocument.Content

Set myDoc = ActiveDocument

gottaList = False

For Each myWnd In Application.Windows

thisName = myWnd.Document.Name

If InStr(LCase(thisName), "list") > 0 Then

myWnd.Document.Activate

myResponse = MsgBox("Is this your list?" & vbCr & vbCr \_

& ">>> " & thisName & " <<<", \_

vbQuestion + vbYesNoCancel, "HighlightWordList")

If myResponse = vbYes Then

gottaList = True

Exit For

End If

End If

Next myWnd

If gottaList = False Then

Beep

MsgBox "Can't find a word list."

Exit Sub

End If

nowTrack = myDoc.TrackRevisions

myDoc.TrackRevisions = False

Set myList = myWnd.Document.Content

If rng.Text = myList.Text Then

Beep

MsgBox "Please place the cursor in the text to be" & vbCr & "highlighted and rerun the macro."

Exit Sub

End If

With myList.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^11"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

If myDoc.Footnotes.count > 0 Then

Set rngFoot = myDoc.StoryRanges(wdFootnotesStory)

End If

If myDoc.Endnotes.count > 0 Then

Set rngEnd = myDoc.StoryRanges(wdEndnotesStory)

End If

For Each myPara In myWnd.Document.Paragraphs

myText = Replace(myPara.Range.Text, vbCr, "")

If Left(myText, 1) = "#" Then

Beep

myDoc.TrackRevisions = nowTrack

Exit Sub

End If

dotPos = InStr(myText, " . .")

If dotPos > 0 Then myText = Left(myText, dotPos - 1)

tabPos = InStr(myText, vbTab)

If tabPos > 0 Then myText = Mid(myText, tabPos + 1)

If Len(myText) > 1 And Left(myText, 1) <> "|" Then

thisHighlightColour = myPara.Range.Characters(tabPos + 2).HighlightColorIndex

Options.DefaultHighlightColorIndex = thisHighlightColour

thisTextColour = myPara.Range.Characters(tabPos + 2).Font.Color

myFind = myText

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = myFind

.Font.StrikeThrough = False

.Replacement.Text = "^&"

If thisHighlightColour > 0 Then .Replacement.Highlight = True

If thisTextColour > 0 Then .Replacement.Font.Color = thisTextColour

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = doWholeWordsOnly

If (thisHighlightColour > 0) Or (thisTextColour > 0) Then

.Execute Replace:=wdReplaceAll

End If

End With

If myDoc.Footnotes.count > 0 Then

With rngFoot.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = myFind

.Font.StrikeThrough = False

.Replacement.Text = "^&"

If thisHighlightColour > 0 Then .Replacement.Highlight = True

If thisTextColour > 0 Then .Replacement.Font.Color = thisTextColour

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = doWholeWordsOnly

If (thisHighlightColour > 0) Or (thisTextColour > 0) Then

.Execute Replace:=wdReplaceAll

End If

End With

End If

If myDoc.Endnotes.count > 0 Then

With rngEnd.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = myFind

.Font.StrikeThrough = False

.Replacement.Text = "^&"

If thisHighlightColour > 0 Then .Replacement.Highlight = True

If thisTextColour > 0 Then .Replacement.Font.Color = thisTextColour

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = doWholeWordsOnly

If (thisHighlightColour > 0) Or (thisTextColour > 0) Then

.Execute Replace:=wdReplaceAll

End If

End With

End If

End If

DoEvents

Next myPara

Beep

myDoc.Activate

myDoc.TrackRevisions = nowTrack

End Sub

**Sub RevisionsHighlight()**

' Paul Beverley - Version 19.02.20

' Highlights all the edits in a text

myRange = 3

' = how many characters either side of the edit to highlight

myHighlight = wdBrightGreen

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

For Each rv In ActiveDocument.Revisions

Set rng = rv.range

rng.Start = rng.Start - myRange

rng.End = rng.End + myRange

rng.HighlightColorIndex = myHighlight

Next rv

ActiveDocument.TrackRevisions = myTrack

rng.Select

Beep

End Sub

**Sub BordersAddToText()**

' Paul Beverley - Version 26.05.18

' Changes underlined+highlighted text to coloured borders

For i = 1 To 3

Select Case i

Case 1

gotRange = (ActiveDocument.Footnotes.Count > 0)

If gotRange Then Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.Content

gotRange = (ActiveDocument.Endnotes.Count > 0)

If gotRange Then Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content: gotRange = True

End Select

If gotRange Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.Execute

End With

Do While rng.Find.Found = True

Select Case rng.HighlightColorIndex

Case wdBrightGreen:

With Options

.DefaultBorderLineStyle = wdLineStyleSingle

.DefaultBorderLineWidth = wdLineWidth150pt

.DefaultBorderColor = wdColorBrightGreen

End With

Case wdRed:

With Options

.DefaultBorderLineStyle = wdLineStyleSingle

.DefaultBorderLineWidth = wdLineWidth150pt

.DefaultBorderColor = wdColorRed

End With

Case wdTurquoise:

With Options

.DefaultBorderLineStyle = wdLineStyleSingle

.DefaultBorderLineWidth = wdLineWidth150pt

.DefaultBorderColor = wdColorBlue

End With

Case wdPink:

With Options

.DefaultBorderLineStyle = wdLineStyleSingle

.DefaultBorderLineWidth = wdLineWidth300pt

.DefaultBorderColor = wdColorPink

End With

Case wdYellow:

With Options

.DefaultBorderLineStyle = wdLineStyleSingleWavy

' Sorry, you can't make the line thicker, I don't think

.DefaultBorderColor = wdColorBlack

End With

Case wdGreen:

With Options

.DefaultBorderLineStyle = wdLineStyleDouble

.DefaultBorderLineWidth = wdLineWidth150pt

.DefaultBorderColor = wdColorTurquoise

End With

Case wdGray25:

With Options

.DefaultBorderLineStyle = wdLineStyleDot

.DefaultBorderLineWidth = wdLineWidth225pt

.DefaultBorderColor = wdColorRed

End With

Case wdGray50:

With Options

.DefaultBorderLineStyle = wdLineStyleEmboss3D

.DefaultBorderColor = wdColorTurquoise

End With

Case Else

With Options

.DefaultBorderLineStyle = wdLineStyleSingle

.DefaultBorderLineWidth = wdLineWidth150pt

.DefaultBorderColor = wdColorBlack

End With

End Select

With rng.Font.Borders(1)

.LineStyle = Options.DefaultBorderLineStyle

.LineWidth = Options.DefaultBorderLineWidth

.Color = Options.DefaultBorderColor

End With

rng.Underline = False

rng.HighlightColorIndex = wdNoHighlight

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

End If

Next i

End Sub

**Sub BordersAllOff()**

' Paul Beverley - Version 26.05.18

' Removes border attributes from some or all text

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If Selection.Start = Selection.End Then

myResponse = MsgBox("Remove borders from whole text?", vbQuestion \_

+ vbYesNoCancel, "BordersAllOff")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

Else

Set rng = Selection.range.Duplicate

End If

rng.Font.Borders(1).LineStyle = wdLineStyleNone

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub NavPaneCustomize()**

' Paul Beverley - Version 12.01.21

' Opens the navigation pane where and how you want

h = 600

w = 400

' doSetUp = True

doSetUp = False

If doSetUp = True Then

w = Application.CommandBars("Navigation").Width

h = Application.CommandBars("Navigation").Height

MsgBox "H: " & h & vbCr & vbCr & "W: " & w

Exit Sub

End If

If Application.CommandBars("Navigation").Visible = False Then

Application.CommandBars("Navigation").Visible = True

'Application.CommandBars("Navigation").Position = msoBarRight

Application.CommandBars("Navigation").Position = msoBarLeft

'Application.CommandBars("Navigation").Position = msoBarFloating

'Application.CommandBars("Navigation").Height = h

Application.CommandBars("Navigation").Width = w

Else

Application.CommandBars("Navigation").Visible = False

End If

End Sub

**Sub BookmarkTempAdd()**

' Paul Beverley - Version 12.01.16

' Add temporary marker

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.Collapse wdCollapseStart

ActiveDocument.Bookmarks.Add Name:="myTempMark"

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub BookmarkTempFind()**

' Paul Beverley - Version 29.04.20

' Go to temporary marker

forceToTop = True

findMark = "myTempMark"

If Selection.Bookmarks.Exists(findMark) Then \_

findMark = "myTempMark2"

If ActiveDocument.Bookmarks.Exists(findMark) Then

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

ActiveDocument.Bookmarks(findMark).Select

If forceToTop = True Then

Set wasSelected = Selection.range

Application.ScreenUpdating = False

Selection.EndKey Unit:=wdStory

wasSelected.Select

Application.ScreenUpdating = True

ActiveDocument.ActiveWindow.SmallScroll Down:=1

End If

ActiveDocument.TrackRevisions = myTrack

Selection.Collapse wdCollapseEnd

Selection.MoveRight , 1

Else

If ActiveDocument.Bookmarks.Exists("myTempMark2") Then

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

ActiveDocument.Bookmarks("myTempMark2").Select

If forceToTop = True Then

Set wasSelected = Selection.range

Application.ScreenUpdating = False

Selection.EndKey Unit:=wdStory

wasSelected.Select

Application.ScreenUpdating = True

ActiveDocument.ActiveWindow.SmallScroll Down:=1

End If

ActiveDocument.TrackRevisions = myTrack

Selection.Collapse wdCollapseEnd

Selection.MoveRight , 1

Else

Beep

If ActiveDocument.Bookmarks.Exists("myTempMark") Then

MsgBox "You're already at the bookmark!"

Selection.MoveLeft , 1

Selection.MoveRight , 1

Else

Selection.Collapse wdCollapseEnd

MsgBox "No temporary bookmarks found"

End If

End If

End If

ActiveDocument.ActiveWindow.SmallScroll Down:=1

Selection.MoveLeft , 1

End Sub

**Sub BookmarkToCursorSelect()**

' Paul Beverley - Version 12.01.16

' Select from temporary marker to cursor

cursorPosn = Selection.Start

findMark = "myTempMark"

If ActiveDocument.Bookmarks.Exists(findMark) Then

ActiveDocument.Bookmarks(findMark).Select

Else

Selection.Collapse wdCollapseEnd

MsgBox "Temporary bookmarks not found"

Exit Sub

End If

bkmkPosn = Selection.Start

Selection.Start = cursorPosn

Selection.End = cursorPosn

Selection.MoveRight , 1

Selection.MoveLeft , 1

If cursorPosn < bkmkPosn Then

Selection.Start = cursorPosn

Selection.End = bkmkPosn

Else

Selection.End = cursorPosn

Selection.Start = bkmkPosn

End If

End Sub

**Sub BookmarkTempClear()**

' Paul Beverley - Version 12.01.16

' Delete temporary markers

If ActiveDocument.Bookmarks.Exists("myTempMark2") Then \_

ActiveDocument.Bookmarks("myTempMark2").Delete

If ActiveDocument.Bookmarks.Exists("myTempMark") Then

ActiveDocument.Bookmarks("myTempMark").Delete

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Else

MsgBox "Temporary bookmark not found"

End If

End Sub

**Sub InstantFindDown()**

**'** Paul Beverley - Version 14.03.24

' Finds selected text downwards

doTrim = True

addBookmark = True

butNotTheseFiles = "zzSwitchList,ComputerTools4Eds,TheMacros,5\_Library,VideoList"

Set rng = Selection.Range.Duplicate

If rng.Start = rng.End Then

rng.MoveEnd , 1

If rng.Text = vbCr Then

rng.MoveEnd , -2

Else

rng.Collapse wdCollapseStart

End If

rng.Expand wdWord

For i = 1 To 3

If InStr(ChrW(8217) & " '", Right(rng.Text, 1)) > 0 Then \_

rng.End = rng.End - 1

Next i

End If

thisBit = rng.Text

endNow = Selection.End

Selection.Collapse wdCollapseStart

myName = ActiveDocument.Name

dotPos = InStr(myName, ".")

If dotPos > 0 Then myName = Left(myName, dotPos - 1)

noMarker = (InStr(LCase(butNotTheseFiles), LCase(myName)) > 0)

If InStr(myName, "Ch") > 0 Then noMarker = True

If addBookmark = True And noMarker = False Then \_

ActiveDocument.Bookmarks.Add Name:="myTempMark2"

Selection.Start = endNow

If Asc(thisBit) <> 32 And doTrim Then thisBit = Trim(thisBit)

thisBit = Replace(thisBit, "^", "^^")

thisBit = Replace(thisBit, vbCr, "^p")

thisBit = Replace(thisBit, vbTab, "^t")

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = thisBit

.Replacement.Text = thisBit

.MatchWildcards = False

.MatchCase = False

If UCase(thisBit) = thisBit Then .MatchCase = True

.MatchWholeWord = False

.Forward = True

.Execute

End With

If Selection.End = endNow And \_

Selection.Find.Found = False Then Beep

' Leaves F&R dialogue in a sensible state

Selection.Find.Wrap = wdFindContinue

End Sub

**Sub InstantFindDownWild()**

' Paul Beverley - Version 23.02.23

' Finds selected text downwards wildcards set

selectWholeLine = True

addBookmark1 = True

addBookmark2 = False

On Error GoTo ReportIt

If Selection.Start = Selection.End And selectWholeLine = True \_

Then Selection.Expand wdParagraph

If Right(Selection.Text, 1) = vbCr Then Selection.MoveEnd , -1

thisBit = Selection

thisBit = Replace(thisBit, vbCr, "^13")

thisBit = Replace(thisBit, vbTab, "^t")

wordEnd = Selection.End

Selection.Collapse wdCollapseStart

If addBookmark1 = True Then ActiveDocument.Bookmarks.Add Name:="myTempMark"

If addBookmark2 = True Then ActiveDocument.Bookmarks.Add Name:="myTempMark2"

Selection.Start = wordEnd

Selection.Start = Selection.End

Set rng = Selection.Range.Duplicate

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = thisBit

.Replacement.Text = ""

.MatchWildcards = True

.MatchWholeWord = False

.MatchCase = False

.Forward = True

.Execute

.Wrap = wdFindContinue

End With

If Selection.Find.Found = False Then

Beep

Else

If Selection.End = 0 Then

rng.Select

Beep

myResponse = MsgBox("Sorry, Word's Find facility is playing sillies!" \_

& vbCr & vbCr & "Try searching in a text-only copy.", \_

vbOKOnly, "InstantFindDownWild")

End If

End If

Exit Sub

ReportIt:

asdfgsdf = Err.Number

If Err.Number = 5560 Then

Beep

MsgBox ("Bad pattern match!")

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub InstantFindUp()**

' Paul Beverley - Version 07.01.22

' Finds selected text upwards

doTrim = True

addBookmark = True

butNotTheseFiles = "zzSwitchList,ComputerTools4Eds,TheMacros,5\_Library"

Set rng = Selection.Range.Duplicate

If rng.Start = rng.End Then

rng.Expand wdWord

For i = 1 To 3

If InStr(ChrW(8217) & " '", Right(rng.Text, 1)) > 0 Then \_

rng.End = rng.End - 1

Next i

End If

thisBit = rng.Text

hereNow = Selection.Start

makeWild = False

If Right(thisBit, 1) = ">" Then makeWild = True

If Left(thisBit, 1) = "<" Then makeWild = True

wordEnd = Selection.End

Selection.Collapse wdCollapseStart

myName = ActiveDocument.Name

dotPos = InStr(myName, ".")

If dotPos > 0 Then myName = Left(myName, dotPos - 1)

noMarker = (InStr(butNotTheseFiles, myName) > 0)

If InStr(myName, "Ch") > 0 Then noMarker = True

If addBookmark = True And noMarker = False Then \_

ActiveDocument.Bookmarks.Add Name:="myTempMark2"

If Asc(thisBit) <> 32 And doTrim Then thisBit = Trim(thisBit)

thisBit = Replace(thisBit, "^", "^^")

If makeWild = True Then

thisBit = Replace(thisBit, vbCr, "^p")

Else

thisBit = Replace(thisBit, vbCr, "^13")

End If

thisBit = Replace(thisBit, vbTab, "^t")

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Forward = False

.Text = thisBit

.Replacement.Text = thisBit

.MatchCase = False

If UCase(thisBit) = thisBit Then .MatchCase = True

.MatchWildcards = makeWild

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If Selection.Start = hereNow And \_

Selection.Find.Found = False Then Beep

' Leaves F&R dialogue in a sensible state

With Selection.Find

.Wrap = wdFindContinue

.Forward = True

End With

End Sub

**Sub InstantFindTop()**

**'** Paul Beverley - Version 18.11.10

' Find this from the top

If Selection.Start = Selection.End Then

Selection.MoveRight Unit:=wdWord, Count:=1

Selection.MoveLeft Unit:=wdWord, Count:=1, Extend:=wdExtend

End If

thisBit = Trim(Selection)

Selection.Start = Selection.End

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = False

.Text = thisBit

.Replacement.Text = thisBit

.MatchWildcards = False

.MatchWholeWord = False

.MatchCase = False

.Forward = True

.Execute

End With

' Move the screen display down a couple of lines

Selection.MoveUp Unit:=wdLine, Count:=2

Selection.MoveDown Unit:=wdLine, Count:=2

Selection.Find.Execute

' Leaves F&R dialogue in a sensible state

Selection.Find.Wrap = wdFindContinue

End Sub

**Sub InstantFindBottom()**

**'** Paul Beverley - Version 25.02.11

' I couldn't possibly comment!

If Selection.Start = Selection.End Then

Selection.MoveRight Unit:=wdWord, Count:=1

Selection.MoveLeft Unit:=wdWord, Count:=1, Extend:=wdExtend

End If

thisBit = Trim(Selection)

Selection.Start = Selection.End

Selection.EndKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = False

.Text = thisBit

.Replacement.Text = thisBit

.MatchWildcards = False

.MatchWholeWord = False

.MatchCase = False

.Forward = False

.Execute

End With

' Move the screen display down a couple of lines

Selection.MoveUp Unit:=wdLine, Count:=2

Selection.MoveDown Unit:=wdLine, Count:=2

Selection.Find.Execute

' Leaves F&R dialogue in a sensible state

With Selection.Find

.Wrap = wdFindContinue

.Forward = True

End With

End Sub

**Sub FindFwd()**

**'** Paul Beverley - Version 20.12.23

' Finds next match forwards, case insensitively

mySpaces = " "

For i = 1 To 3

mySpaces = mySpaces & mySpaces

Next i

mySch = Selection.Find.Text

thisIsWild = ((InStr(mySch, "]") + InStr(mySch, "<") \_

+ InStr(mySch, ">") > 0))

If thisIsWild Then StatusBar = mySpaces & "Using WILDCARD find"

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseEnd

With Selection.Find

.Wrap = wdFindStop

.Forward = True

.MatchCase = False

.MatchWildcards = thisIsWild

.Execute

.Wrap = wdFindStop

End With

If Selection.Find.Found = False Then

Beep

Else

If Selection.End = 0 Then

rng.Select

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Selection.EndKey Unit:=wdStory

With Selection.Find

.Wrap = wdFindStop

.Forward = False

.Execute

' Leave F&R dialogue in a sensible state

.Wrap = wdFindContinue

.Forward = True

End With

StatusBar = "Sorry, Word's Find facility is playing sillies!"

Else

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseStart

Selection.MoveUp , 1

rng.Select

End If

End If

End Sub

**Sub FindFwdAll()**

' Paul Beverley - Version 27.06.20

' Moves forward to next match: text > notes > comments

Selection.Collapse wdCollapseEnd

hereNow = Selection.End

With Selection.Find

.Wrap = wdFindStop

.Forward = True

.MatchCase = False

.Execute

End With

If Selection.Start = hereNow Then

inFootnotes = Selection.Information(wdInFootnote)

inEndnotes = Selection.Information(wdInEndnote)

inComments = Selection.Information(wdInCommentPane)

inText = Not (inFootnotes Or inEndnotes Or inComments)

anyComments = (ActiveDocument.Comments.Count > 0)

anyFootnotes = (ActiveDocument.Footnotes.Count > 0)

anyEndnotes = (ActiveDocument.Endnotes.Count > 0)

If Selection.Find.Found = False Then

If inText And anyFootnotes Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

rng.Select

Selection.Collapse wdCollapseStart

With Selection.Find

.Wrap = wdFindStop

.Forward = True

.MatchCase = False

.Execute

End With

If Selection.Find.Found = True Then

GoTo CleanUp

End If

inFootnotes = True

Else

inFootnotes = True

End If

If inFootnotes And anyEndnotes Then

windowNow = ActiveWindow.WindowNumber

ActiveDocument.Endnotes(1).Range.Select

ActiveDocument.Windows(windowNow).Activate

Selection.Collapse wdCollapseStart

Selection.Find.Execute

If Selection.Find.Found = True Then GoTo CleanUp

inEndnotes = True

Else

inEndnotes = True

End If

If inEndnotes And anyComments Then

If inComments Then GoTo CleanUp

ActiveDocument.Comments(1).Edit

Selection.WholeStory

Selection.Collapse wdCollapseStart

With Selection.Find

.Wrap = wdFindStop

.Forward = True

.MatchCase = False

.Execute

End With

End If

End If

End If

CleanUp:

If Selection.Find.Found = False Then Beep

' Leave F&R dialogue in a sensible state

Selection.Find.Wrap = wdFindContinue

End Sub

**Sub FindFwdCase()**

**'** Paul Beverley - Version 16.06.23

' Next case-sensitive find forwards

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseEnd

With Selection.Find

.Wrap = wdFindStop

.Forward = True

.MatchCase = True

.MatchWildcards = (InStr(.Text, "]") + InStr(.Text, "<") + \_

InStr(.Text, ">") > 0)

.Execute

' Leave F&R dialogue in a sensible state

.Wrap = wdFindContinue

End With

If Selection.Find.Found = False Then

Beep

Else

If Selection.End = 0 Then

rng.Select

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Selection.EndKey Unit:=wdStory

With Selection.Find

.Wrap = wdFindStop

.Forward = False

.Execute

' Leave F&R dialogue in a sensible state

.Wrap = wdFindContinue

.Forward = True

End With

StatusBar = "Sorry, Word's Find facility is playing sillies!"

Else

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseStart

Selection.MoveUp , 1

rng.Select

End If

End If

End Sub

**Sub FindBack()**

**'** Paul Beverley - Version 16.06.23

' Next find backwards

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseStart

With Selection.Find

.Wrap = wdFindStop

.Forward = False

.MatchCase = False

If InStr(.Text, "]") + InStr(.Text, "<") + \_

InStr(.Text, ">") > 0 Then .MatchWildcards = True

.Execute

' Leave F&R dialogue in a sensible state

.Wrap = wdFindContinue

.Forward = True

End With

If Selection.Find.Found = False Then

Beep

Else

If Selection.Start = rng.Start Then

rng.Select

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

StatusBar = "Sorry, Word's Find facility is playing sillies!"

Else

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseStart

Selection.MoveDown , 1

rng.Select

End If

End If

End Sub

**Sub FindBackCase()**

**'** Paul Beverley - Version 16.06.21

' Next case-sensitive find backwards

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseStart

With Selection.Find

.Wrap = wdFindStop

.Forward = False

.MatchCase = True

If InStr(.Text, "]") + InStr(.Text, "<") + \_

InStr(.Text, ">") > 0 Then .MatchWildcards = True

.Execute

' Leave F&R dialogue in a sensible state

.Wrap = wdFindContinue

.Forward = True

End With

If Selection.Find.Found = False Then

Beep

Else

If Selection.Start = rng.Start Then

rng.Select

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

StatusBar = "Sorry, Word's Find facility is playing sillies!"

Else

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseStart

Selection.MoveDown , 1

rng.Select

End If

End If

End Sub

**Sub InstantJumpUp()**

' Paul Beverley - Version 13.01.21

' Find selected text upwards

addBookmark = False

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

apoPos = InStr(Selection, ChrW(8217))

If apoPos > 0 Then Selection.End = Selection.Start + apoPos - 1

End If

thisBit = Selection

wordEnd = Selection.End

Selection.Collapse wdCollapseStart

If addBookmark = True Then ActiveDocument.Bookmarks.Add Name:="myTempMark2"

If Asc(thisBit) <> 32 Then thisBit = Trim(thisBit)

thisBit = Replace(thisBit, "^", "^^")

Selection.End = Selection.Start

hereNow = Selection.Start

oldFind = Selection.Find.Text

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Forward = False

.Text = thisBit

.MatchCase = False

.MatchWildcards = False

.Execute

End With

' Leave F&R dialogue in a sensible state

Selection.Find.Forward = True

Selection.Find.Wrap = wdFindContinue

If Selection.Start = hereNow And \_

Selection.Find.Found = False Then Beep

Selection.Find.Text = oldFind

End Sub

**Sub InstantJumpDown()**

' Paul Beverley - Version 12.01.21

' Find selected text downwards

addBookmark = False

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

apoPos = InStr(Selection, ChrW(8217))

If apoPos > 0 Then Selection.End = Selection.Start + apoPos - 1

End If

thisBit = Selection

wordEnd = Selection.End

Selection.Collapse wdCollapseEnd

If addBookmark = True Then ActiveDocument.Bookmarks.Add Name:="myTempMark2"

If Asc(thisBit) <> 32 Then thisBit = Trim(thisBit)

thisBit = Replace(thisBit, "^", "^^")

Selection.End = Selection.Start

hereNow = Selection.Start

oldFind = Selection.Find.Text

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Forward = True

.Text = thisBit

.MatchCase = False

.MatchWildcards = False

.Execute

End With

If Selection.Start = hereNow And \_

Selection.Find.Found = False Then Beep

Selection.Find.Text = oldFind

End Sub

**Sub FindReplaceGo()**

' Paul Beverley - Version 12.01.21

' Finds, replaces and moves to the next match

hereNow = Selection.End

If Selection.End = Selection.Start Then

With Selection.Find

.Wrap = wdFindStop

.Forward = True

.Execute

End With

Else

Selection.End = Selection.Start

With Selection.Find

.Execute Replace:=wdReplaceOne

.Wrap = wdFindStop

.Forward = True

End With

Selection.Start = Selection.End

Selection.Find.Execute

End If

If Selection.Start = hereNow And \_

Selection.Find.Found = False Then Beep

' Leave F&R dialogue in a sensible state

Selection.Find.Wrap = wdFindContinue

End Sub

**Sub FindReplaceStay()**

' Paul Beverley - Version 09.03.17

' Finds and replaces but don't move to next

Selection.Collapse wdCollapseStart

Selection.Find.Execute Replace:=wdReplaceOne

End Sub

**Sub FindClip()**

' Paul Beverley - Version 12.01.21

' Find whatever is in the clipboard

' Move up this many lines first, before starting the search

moveLinesUp = 20

moveLinesUp = 10

Application.ScreenUpdating = False

If moveLinesUp > 0 Then Selection.MoveUp , moveLinesUp

hereNow = Selection.Start

Selection.Paste

Selection.Start = hereNow

strClip = Trim(Selection)

WordBasic.EditUndo

Application.ScreenUpdating = True

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Text = strClip

.Execute

End With

If Selection.Start = hereNow And \_

Selection.Find.Found = False Then Beep

'Add this to leave F&R dialogue in a sensible state

Selection.Find.Wrap = wdFindContinue

End Sub

**Sub FindClipTop()**

' Paul Beverley - Version 12.01.21

' Jumps to the top, and finds whatever is in the clipboard

Application.ScreenUpdating = False

Selection.HomeKey Unit:=wdStory

Selection.Paste

Selection.Start = 0

strClip = Trim(Selection)

WordBasic.EditUndo

Application.ScreenUpdating = True

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Forward = True

.Text = strClip

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If Selection.Start = hereNow And \_

Selection.Find.Found = False Then Beep

Selection.Find.Wrap = wdFindContinue

End Sub

**Sub InstantFindFormatDown()**

**'** Paul Beverley - Version 10.09.10

' Find format similar to this

hereNow = Selection.Start

isSuper = Selection.Font.Superscript

thisBit = Trim(Selection)

If Selection.End = Selection.Start Then

thisBit = ""

Selection.MoveEnd , 1

End If

isSuper = Selection.Font.Superscript

isSub = Selection.Font.Subscript

isItalic = Selection.Font.Italic

isBold = Selection.Font.Bold

Selection.MoveStart , 10

Selection.End = Selection.Start

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = False

If isSuper Then .Font.Superscript = True

If isSub Then .Font.Subscript = True

If isItalic Then .Font.Italic = True

If isBold Then .Font.Bold = True

.Text = thisBit

.Replacement.Text = thisBit

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.MatchCase = False

.Forward = True

.Execute

End With

If Selection.End = hereNow Then beep

If Selection.Start = hereNow + 10 Then

beep

Selection.Start = hereNow

Selection.End = hereNow

End If

' Leave F&R dialogue in a sensible state

Selection.Find.Wrap = wdFindContinue

End Sub

**Sub InstantFindFormatUp()**

**'** Paul Beverley - Version 10.09.10

' Find format similar to this

hereNow = Selection.Start

thisBit = Trim(Selection)

If Selection.End = Selection.Start Then

thisBit = ""

Selection.MoveEnd , 1

End If

isSuper = Selection.Font.Superscript

isSub = Selection.Font.Subscript

isItalic = Selection.Font.Italic

isBold = Selection.Font.Bold

Selection.MoveStart , -10

Selection.End = Selection.Start

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = False

.Forward = False

If isSuper Then .Font.Superscript = True

If isSub Then .Font.Subscript = True

If isItalic Then .Font.Italic = True

If isBold Then .Font.Bold = True

.Text = thisBit

.Replacement.Text = thisBit

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If Selection.Start = hereNow Then beep

If Selection.Start = hereNow - 10 Then

beep

Selection.Start = hereNow

Selection.End = hereNow

End If

'Add these two to leave F&R dialogue in a sensible state

Selection.Find.Forward = True

Selection.Find.Wrap = wdFindContinue

End Sub

**Sub FindSamePlaceBack()**

**'** Paul Beverley - Version 19.12.24

' Finds the same place in another open file

selectCurrentWord = False

selectCurrentWord = True

doMatchCase = True

doWholeWordSearch = False

' Select whole words

' wholeWordsSelect = True

wholeWordsSelect = False

' onlyLookInTheseFiles = "\_PB"

onlyLookInTheseFiles = ""

notThisFile = "zzSwitchList"

notThisFile = "zzSw"

notThisFileEnd = "XX"

alphaOrderUp = False

' alphaOrderUp = True

preserveOriginalFind = False

myStep = 10

minLength = 15

myWidth = 1000

myHeight = 500

nowFind = Selection.Find.Text

Selection.Find.MatchWholeWord = False

If Selection.start = Selection.End Then

If selectCurrentWord = True Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", \_

Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

hereNow = Selection.start

Selection.HomeKey Unit:=wdLine

cursorPos = hereNow - Selection.start

Selection.MoveDown Unit:=wdLine, count:=1, \_

Extend:=wdExtend

repeatedSearch = (Len(Selection) > 2 \* myStep)

End If

Else

If wholeWordsSelect = True Then

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.start = Selection.start

rng.Select

End If

End If

dotsPos = InStr(Selection, " . . . ")

If dotsPos > 0 Then

Selection.Collapse wdCollapseStart

Selection.MoveRight , dotsPos - 2

Selection.Expand wdWord

Do While InStr(" ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

dotsPos = InStr(Selection, " . . ")

If dotsPos > 0 Then

Selection.Collapse wdCollapseStart

Selection.MoveEndUntil cset:=".", count:=wdForward

Selection.MoveEnd , -1

End If

mySearch = Trim(Replace(Selection, Chr(13), ""))

Selection.Collapse wdCollapseEnd

Set thisDoc = ActiveDocument

Set rng = ActiveDocument.Content

rng.End = Selection.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[[[[[ "

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = False

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

If rng.Find.Found Then

rng.Expand wdParagraph

rng.End = rng.End - 1

tryThisName = rng

tryThisName = Replace(tryThisName, " ]]]]]", "")

tryThisName = Replace(tryThisName, "[[[[[ ", "")

Else

tryThisName = ""

End If

totWnds = Application.Windows.count

' What's this for? I dunno!

' Something to do with having multiple windows

' open on a single file?

For i = 1 To totWnds

Set myDoc = Application.Windows(i).Document

If myDoc.FullName = thisDoc.FullName Then

thisFileNum = i

myDocNum = i

Exit For

End If

Next i

If alphaOrderUp = True Then

s = 1

Else

s = -1

End If

myPtr = totWnds + myDocNum - 1

For i = 1 To totWnds - 1

myPtr = myPtr + s

n = myPtr Mod totWnds + 1

Set myWnd = Application.Windows(n)

winState = myWnd.WindowState

Set myDoc = Application.Windows(n).Document

endBit = Right(myDoc.Content, 10)

doThis = (InStr(endBit, notThisFileEnd) = 0)

thisName = myDoc.Name

If onlyLookInTheseFiles > "" Then

doThis = (InStr(thisName, onlyLookInTheseFiles) > 0)

End If

If myDoc.FullName <> thisDoc.FullName And doThis And \_

InStr(myDoc.Name, notThisFile) = 0 Then

If thisName = tryThisName Then tryThisName = ""

Set rng = Documents(myDoc).Content

Debug.Print myDoc.Name

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchCase = doMatchCase

.MatchWildcards = False

If selectCurrentWord = True And \_

doWholeWordSearch = True Then

.MatchWholeWord = True

Else

.MatchWholeWord = False

End If

.Text = mySearch

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

myDoc.Activate

If myWnd.WindowState = 2 Then \_

myWnd.WindowState = wdWindowStateNormal

rng.Select

Selection.MoveLeft , 1

rng.Select

If tryThisName > "" Then

GoTo LoadFile

Else

GoTo myEnd

End If

Else

numFoots = myDoc.Footnotes.count

numEnds = myDoc.Endnotes.count

If numFoots + numEnds > 0 Then

If numFoots > 0 Then Set rng = \_

myDoc.StoryRanges(wdFootnotesStory)

If numEnds > 0 Then Set rng = \_

myDoc.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchCase = doMatchCase

.MatchWildcards = False

If selectCurrentWord = True Then

.MatchWholeWord = True

Else

.MatchWholeWord = False

End If

.Text = mySearch

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

rng.Select

If myWnd.WindowState = 2 Then \_

myWnd.WindowState = wdWindowStateNormal

GoTo myEnd

End If

End If

End If

End If

myWnd.WindowState = winState

Next i

LoadFile:

' Now try to find the search text in the "[[[[[" file

On Error Resume Next

If tryThisName > "" Then

Documents.Open tryThisName

If Err.Number = 5174 Then

MsgBox ("Can't find file: " & tryThisName)

Err.Clear

Else

Application.Resize Width:=myWidth, Height:=myHeight

On Error GoTo 0

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchCase = doMatchCase

.MatchWildcards = False

If selectCurrentWord = True Then

.MatchWholeWord = True

Else

.MatchWholeWord = False

End If

.Text = mySearch

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

If myWnd.WindowState = 2 Then \_

myWnd.WindowState = wdWindowStateNormal

rng.Select

GoTo myEnd

Else

numFoots = myDoc.Footnotes.count

numEnds = myDoc.Endnotes.count

If numFoots + numEnds > 0 Then

If numFoots > 0 Then Set rng = \_

ActiveDocument.StoryRanges(wdFootnotesStory)

If numEnds > 0 Then Set rng = \_

ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchCase = doMatchCase

.MatchWildcards = False

If selectCurrentWord = True Then

.MatchWholeWord = True

Else

.MatchWholeWord = False

End If

.Text = mySearch

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

rng.Select

If myWnd.WindowState = 2 Then \_

myWnd.WindowState = wdWindowStateNormal

GoTo myEnd

End If

End If

End If

End If

End If

' Now try looking for a subset of the search text

If repeatedSearch = True Then

Do

If Len(mySearch) > 2 \* cursorPos Then

mySearch = Left(mySearch, Len(mySearch) - myStep)

Else

mySearch = Right(mySearch, Len(mySearch) - myStep)

cursorPos = cursorPos - myStep

End If

myPtr = totWnds + myDocNum - 1

For i = 1 To totWnds - 1

myPtr = myPtr + s

n = myPtr Mod totWnds + 1

Set myWnd = Application.Windows(n)

Set myDoc = Application.Windows(n).Document

If myDoc.FullName <> thisDoc.FullName And \_

InStr(myDoc.Name, notThisFile & ".doc") = 0 Then

If thisName = tryThisName Then tryThisName = ""

myDoc.Activate

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = False

.Text = mySearch

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

If myWnd.WindowState = 2 Then \_

myWnd.WindowState = wdWindowStateNormal

rng.Select

GoTo myEnd

End If

End If

Next i

Loop Until Len(mySearch) < minLength

End If

Beep

thisDoc.Activate

myEnd:

If rng.Find.Found = True Then myDoc.Activate

If preserveOriginalFind = True Then

Selection.Find.Text = nowFind

Else

Selection.Find.Text = mySearch

End If

End Sub

**Sub FindSamePlace()**

**'** Paul Beverley - Version 19.12.24

' Finds the same place in another open file

selectCurrentWord = False

selectCurrentWord = True

doMatchCase = True

doWholeWordSearch = False

' Select whole words

' wholeWordsSelect = True

wholeWordsSelect = False

' onlyLookInTheseFiles = "\_PB"

onlyLookInTheseFiles = ""

notThisFile = "zzSwitchList"

notThisFile = "zzSw"

notThisFileEnd = "XX"

' alphaOrderUp = False

alphaOrderUp = True

preserveOriginalFind = False

myStep = 10

minLength = 15

myWidth = 1000

myHeight = 500

nowFind = Selection.Find.Text

Selection.Find.MatchWholeWord = False

If Selection.start = Selection.End Then

If selectCurrentWord = True Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", \_

Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

hereNow = Selection.start

Selection.HomeKey Unit:=wdLine

cursorPos = hereNow - Selection.start

Selection.MoveDown Unit:=wdLine, count:=1, \_

Extend:=wdExtend

repeatedSearch = (Len(Selection) > 2 \* myStep)

End If

Else

If wholeWordsSelect = True Then

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.start = Selection.start

rng.Select

End If

End If

dotsPos = InStr(Selection, " . . . ")

If dotsPos > 0 Then

Selection.Collapse wdCollapseStart

Selection.MoveRight , dotsPos - 2

Selection.Expand wdWord

Do While InStr(" ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

dotsPos = InStr(Selection, " . . ")

If dotsPos > 0 Then

Selection.Collapse wdCollapseStart

Selection.MoveEndUntil cset:=".", count:=wdForward

Selection.MoveEnd , -1

End If

mySearch = Trim(Replace(Selection, Chr(13), ""))

Selection.Collapse wdCollapseEnd

Set thisDoc = ActiveDocument

Set rng = ActiveDocument.Content

rng.End = Selection.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[[[[[ "

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = False

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

If rng.Find.Found Then

rng.Expand wdParagraph

rng.End = rng.End - 1

tryThisName = rng

tryThisName = Replace(tryThisName, " ]]]]]", "")

tryThisName = Replace(tryThisName, "[[[[[ ", "")

Else

tryThisName = ""

End If

totWnds = Application.Windows.count

' What's this for? I dunno!

' Something to do with having multiple windows

' open on a single file?

For i = 1 To totWnds

Set myDoc = Application.Windows(i).Document

If myDoc.FullName = thisDoc.FullName Then

thisFileNum = i

myDocNum = i

Exit For

End If

Next i

If alphaOrderUp = True Then

s = 1

Else

s = -1

End If

myPtr = totWnds + myDocNum - 1

For i = 1 To totWnds - 1

myPtr = myPtr + s

n = myPtr Mod totWnds + 1

Set myWnd = Application.Windows(n)

winState = myWnd.WindowState

Set myDoc = Application.Windows(n).Document

endBit = Right(myDoc.Content, 10)

doThis = (InStr(endBit, notThisFileEnd) = 0)

thisName = myDoc.Name

If onlyLookInTheseFiles > "" Then

doThis = (InStr(thisName, onlyLookInTheseFiles) > 0)

End If

If myDoc.FullName <> thisDoc.FullName And doThis And \_

InStr(myDoc.Name, notThisFile) = 0 Then

If thisName = tryThisName Then tryThisName = ""

Set rng = Documents(myDoc).Content

Debug.Print myDoc.Name

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchCase = doMatchCase

.MatchWildcards = False

If selectCurrentWord = True And \_

doWholeWordSearch = True Then

.MatchWholeWord = True

Else

.MatchWholeWord = False

End If

.Text = mySearch

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

myDoc.Activate

If myWnd.WindowState = 2 Then \_

myWnd.WindowState = wdWindowStateNormal

rng.Select

Selection.MoveLeft , 1

rng.Select

If tryThisName > "" Then

GoTo LoadFile

Else

GoTo myEnd

End If

Else

numFoots = myDoc.Footnotes.count

numEnds = myDoc.Endnotes.count

If numFoots + numEnds > 0 Then

If numFoots > 0 Then Set rng = \_

myDoc.StoryRanges(wdFootnotesStory)

If numEnds > 0 Then Set rng = \_

myDoc.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchCase = doMatchCase

.MatchWildcards = False

If selectCurrentWord = True Then

.MatchWholeWord = True

Else

.MatchWholeWord = False

End If

.Text = mySearch

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

rng.Select

If myWnd.WindowState = 2 Then \_

myWnd.WindowState = wdWindowStateNormal

GoTo myEnd

End If

End If

End If

End If

myWnd.WindowState = winState

Next i

LoadFile:

' Now try to find the search text in the "[[[[[" file

On Error Resume Next

If tryThisName > "" Then

Documents.Open tryThisName

If Err.Number = 5174 Then

MsgBox ("Can't find file: " & tryThisName)

Err.Clear

Else

Application.Resize Width:=myWidth, Height:=myHeight

On Error GoTo 0

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchCase = doMatchCase

.MatchWildcards = False

If selectCurrentWord = True Then

.MatchWholeWord = True

Else

.MatchWholeWord = False

End If

.Text = mySearch

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

If myWnd.WindowState = 2 Then \_

myWnd.WindowState = wdWindowStateNormal

rng.Select

GoTo myEnd

Else

numFoots = myDoc.Footnotes.count

numEnds = myDoc.Endnotes.count

If numFoots + numEnds > 0 Then

If numFoots > 0 Then Set rng = \_

ActiveDocument.StoryRanges(wdFootnotesStory)

If numEnds > 0 Then Set rng = \_

ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchCase = doMatchCase

.MatchWildcards = False

If selectCurrentWord = True Then

.MatchWholeWord = True

Else

.MatchWholeWord = False

End If

.Text = mySearch

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

rng.Select

If myWnd.WindowState = 2 Then \_

myWnd.WindowState = wdWindowStateNormal

GoTo myEnd

End If

End If

End If

End If

End If

' Now try looking for a subset of the search text

If repeatedSearch = True Then

Do

If Len(mySearch) > 2 \* cursorPos Then

mySearch = Left(mySearch, Len(mySearch) - myStep)

Else

mySearch = Right(mySearch, Len(mySearch) - myStep)

cursorPos = cursorPos - myStep

End If

myPtr = totWnds + myDocNum - 1

For i = 1 To totWnds - 1

myPtr = myPtr + s

n = myPtr Mod totWnds + 1

Set myWnd = Application.Windows(n)

Set myDoc = Application.Windows(n).Document

If myDoc.FullName <> thisDoc.FullName And \_

InStr(myDoc.Name, notThisFile & ".doc") = 0 Then

If thisName = tryThisName Then tryThisName = ""

myDoc.Activate

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = False

.Text = mySearch

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

If myWnd.WindowState = 2 Then \_

myWnd.WindowState = wdWindowStateNormal

rng.Select

GoTo myEnd

End If

End If

Next i

Loop Until Len(mySearch) < minLength

End If

Beep

thisDoc.Activate

myEnd:

If rng.Find.Found = True Then myDoc.Activate

If preserveOriginalFind = True Then

Selection.Find.Text = nowFind

Else

Selection.Find.Text = mySearch

End If

End Sub

**Sub FindInProximity()**

' Paul Beverley - Version 12.02.22

' Finds certain words within a given word range

mainWord = "diff"

nearWord1 = "size"

nearWord2 = ""

distance = 20

' Check file for alternative variable settings

myFileName = "zzSwitchList"

Set nowDoc = ActiveDocument

For Each myWnd In Application.Windows

If InStr(myWnd.Document.Name, myFileName) Then

myWnd.Activate

allText = ActiveDocument.Content

' First variable

myVariable = "main = "

myPos = InStr(LCase(allText), myVariable)

If myPos > 0 Then

myData = Mid(allText, myPos + Len(myVariable) + 1)

endPos = InStr(myData, Chr(34)) - 1

If endPos >= 0 Then myData = Left(myData, endPos)

mainWord = myData

End If

' Second variable

myVariable = "near1 = "

myPos = InStr(LCase(allText), myVariable)

If myPos > 0 Then

myData = Mid(LCase(allText), myPos + Len(myVariable) + 1)

endPos = InStr(myData, Chr(34)) - 1

If endPos >= 0 Then myData = Left(myData, endPos)

nearWord1 = myData

End If

' Third variable

myVariable = "near2 = "

myPos = InStr(LCase(allText), myVariable)

If myPos > 0 Then

myData = Mid(allText, myPos + Len(myVariable) + 1)

endPos = InStr(myData, Chr(34)) - 1

If endPos >= 0 Then myData = Left(myData, endPos)

nearWord2 = myData

End If

' Fourth (number) variable

myVariable = "distance = "

myPos = InStr(allText, myVariable)

If myPos > 0 Then distance = Val(Mid(allText, myPos + Len(myVariable)))

End If

Next myWnd

nowDoc.Activate

' Is the selection in a field; if so, escape

numFields = Selection.Range.Fields.Count

If numFields > 0 Then

Selection.Range.Fields(numFields).Select

End If

Selection.Collapse wdCollapseEnd

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

rng.Start = Selection.End

' Go and find the first occurrence

With rng.Find

.Text = mainWord

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

If LCase(mainWord) = mainWord Then

.MatchCase = False

Else

.MatchCase = True

End If

.Execute

End With

Do While rng.Find.Found = True

Set rng2 = rng.Duplicate

rng2.MoveEnd wdWord, distance

rng2.MoveStart wdWord, -distance

If LCase(nearWord1) = nearWord1 Then

wordPos1 = InStr(LCase(rng2), nearWord1)

Else

wordPos1 = InStr(rng2, nearWord1)

End If

If LCase(nearWord2) = nearWord2 Then

wordPos2 = InStr(LCase(rng2), nearWord2)

Else

wordPos2 = InStr(rng2, nearWord2)

End If

found1 = (wordPos1 > 0) Or (nearWord1 = "")

found2 = (wordPos2 > 0) Or (nearWord2 = "")

' If found then select it...

' ...but if either word is null, don't alter the selection

If nearWord1 = "" Then wordPos1 = rng.Start - rng2.Start

If nearWord2 = "" Then wordPos2 = rng.Start - rng2.Start

If found1 And found2 Then

nearWord1Start = rng2.Start + wordPos1

nearWord2Start = rng2.Start + wordPos2

If nearWord1Start < rng.Start Then rng.Start = nearWord1Start - 1

If nearWord2Start < rng.Start Then rng.Start = nearWord2Start - 1

nearWord1End = rng2.Start + wordPos1 + Len(nearWord1)

nearWord2End = rng2.Start + wordPos2 + Len(nearWord2)

If nearWord1End > rng.End Then rng.End = nearWord1End - 1

If nearWord2End > rng.End Then rng.End = nearWord2End - 1

If rng.Fields.Count > 0 Then

myTest = LCase(rng.Text)

myWdPos1 = InStr(myTest, LCase(nearWord1))

myWdPosMain = InStr(myTest, LCase(mainWord))

If myWdPos1 > 1 Then

rng.Start = rng.Start + myWdPos1 + 1

Else

If myWdPosMain > 1 Then rng.Start = rng.Start + myWdPosMain + 1

End If

End If

rng.Select

Exit Sub

End If

rng.Find.Execute

Loop

Beep

rng.Select

End Sub

**Sub FindInContextLoad()**

' Paul Beverley - Version 04.10.18

' Loads name and date ready for FindInContext macro

myDistance = "12"

assumeWholeWords = False

listName = "zzSwitchList"

Set nowFile = ActiveDocument

myEnd = Selection.End

Selection.Collapse wdCollapseStart

If assumeWholeWords = True Then

Selection.Expand wdWord

mainWord = Trim(Selection)

Selection.Start = myEnd

Selection.Expand wdWord

nearWord1 = Selection

Selection.Collapse wdCollapseEnd

Else

myStart = Selection.Start

Selection.Expand wdWord

Selection.Start = myStart

mainWord = Trim(Selection)

Selection.Start = myEnd

Selection.Expand wdWord

Selection.End = myEnd

nearWord1 = Selection

Selection.Collapse wdCollapseEnd

End If

myDistance = InputBox("Search distance?", \_

"FindInContextLoad", myDistance)

On Error Resume Next

defaultLoaded = False

Set thisDoc = ActiveDocument.ActiveWindow

paneNumber = thisDoc.ActiveWindow.Caption

dirName = ActiveDocument.Path

' Go and look for the list file

gottaList = False

For i = 1 To Application.Windows.Count

If InStr(Application.Windows(i).Document.Name, \_

listName) > 0 Then

Set listDoc = Application.Windows(i).Document

gottaList = True

End If

Next i

If gottaList = False Then

Documents.Open dirName & "\" & listName

If Err.Number = 5174 Then

Err.Clear

Documents.Open defaultList

If Err.Number = 5174 Then

Err.Clear

Documents.Open defaultList & ".docx"

If Err.Number = 5174 Then

Err.Clear

defaultList = Replace(defaultList, ".", \_

" [Compatibility Mode].")

Application.Windows(defaultList).Activate

End If

End If

If Err.Number = 5174 Then GoTo ReportIt

If doAnErrorBeep = True Then Beep

defaultLoaded = True

Else

If Err.Number > 0 And Err.Number <> 438 Then GoTo ReportIt

End If

Else

listDoc.Activate

End If

If Documents.Count = 1 Then Exit Sub

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "mainWord = " & Chr$(34)

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

Selection.Collapse wdCollapseEnd

Selection.MoveEndUntil cset:=Chr$(34), Count:=wdForward

Selection.TypeText Text:=mainWord

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "nearWord1 = " & Chr$(34)

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

Selection.Collapse wdCollapseEnd

Selection.MoveEndUntil cset:=Chr$(34), Count:=wdForward

Selection.TypeText Text:=nearWord1

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "distance = "

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

rng.Select

Selection.Collapse wdCollapseEnd

Selection.Expand wdWord

Selection.TypeText Text:=myDistance

nowFile.Activate

Exit Sub

ReportIt:

If Err.Number = 5174 Then

MsgBox ("Couldn't find file: " & listName)

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub MultiSearch()**

' Paul Beverley - Version 28.05.22

' Searches for a set of words in proximity

myRange = 15

searchUp = False

Dim freq(8) As Integer

CR = vbCr: CR2 = CR & CR

Application.ScreenUpdating = False

On Error GoTo ReportIt

restart:

Set rng = Selection.Range.Duplicate

If rng.Start = rng.End Then rng.Expand wdParagraph

DoEvents

posPlus = InStr(rng, "+")

If posPlus > 1 And rng.Words.Count < 20 Then

myTest = Mid(rng, posPlus - 1, 3)

If LCase(myTest) <> UCase(myTest) Then

pbMultiSearch = Replace(rng.Text, vbCr, "")

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

End If

posOR = InStr(rng, "\_")

If posOR > 1 And rng.Words.Count < 20 Then

myTest = Mid(rng, posOR - 1, 3)

If LCase(myTest) <> UCase(myTest) Then

If Right(pbMultiSearch, 1) = "\_" And \_

LCase(theMultiSearch) = theMultiSearch Then

Beep

myResponse = MsgBox("Your criterion is all lowercase." \_

& CR2 & "Only capital letters are case-checked.", vbOKOnly, "MultiSearch")

Exit Sub

End If

pbMultiSearch = Replace(rng.Text, vbCr, "")

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

End If

' Check for a null search criterion

If pbMultiSearch = "" Then

Beep

myResponse = MsgBox("Sorry, I've forgotten what I was looking for!" \_

& CR2 & "Please reload.", vbOKOnly, "MultiSearch")

Application.ScreenUpdating = True

Exit Sub

End If

theMultiSearch = pbMultiSearch

If Right(pbMultiSearch, 1) = "\_" Then

theMultiSearch = Left(pbMultiSearch, Len(pbMultiSearch) - 1)

checkCaps = True

If LCase(theMultiSearch) = theMultiSearch Then

Beep

myResponse = MsgBox("Your criterion is all lowercase." \_

& CR2 & "Only capital letters are case-checked.", vbOKOnly, "pbMultiSearch")

End If

Else

checkCaps = False

End If

' Search for the words

sch = Split("+" & theMultiSearch, "+")

numSch = UBound(sch)

inText = True

If rng.Information(wdInEndnote) = True Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

inText = False

End If

If rng.Information(wdInFootnote) = True Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

inText = False

End If

If inText = True Then Set rng = ActiveDocument.Content

If searchUp = True Then

rng.End = Selection.Start

allTextToEnd = LCase(rng.Text)

totLen = Len(allTextToEnd)

Else

rng.Start = Selection.End

allTextToEnd = LCase(rng.Text)

totLen = Len(allTextToEnd)

End If

gottaSingle = False

doSch = 1

lowestFreq = 99999

numORs = 0

For i = 1 To numSch

sc = LCase(sch(i))

freq(i) = Len(Replace(allTextToEnd, sc, sc & "!")) - totLen

If InStr(sc, "\_") = 0 Then

gottaSingle = True

If freq(i) = 0 Then

Beep

Application.ScreenUpdating = True

Exit Sub

End If

If freq(i) <= lowestFreq Then lowestFreq = freq(i): doSch = i

Else

numORs = numORs + 1

If numORs > 1 Then

Beep

Application.ScreenUpdating = True

myResponse = MsgBox("Sorry, the search criterion is too complex:" \_

& CR2 & pbMultiSearch, vbOKOnly, "MultiSearch")

Exit Sub

End If

End If

DoEvents

Next i

mySearch = sch(doSch)

inText = True

If rng.Information(wdInEndnote) = True Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

inText = False

End If

If rng.Information(wdInFootnote) = True Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

inText = False

End If

If inText = True Then Set rng = ActiveDocument.Content

If searchUp = True Then

rng.End = Selection.Start

Else

rng.Start = Selection.End

End If

selStartNow = Selection.Start

selEndNow = Selection.End

veryEnd = rng.End

' Search just as an 'OR' function

If gottaSingle = False Then

schOR = Split("\_" & theMultiSearch, "\_")

numSchOR = UBound(schOR)

If searchUp = True Then

bestFind = 0

bestFindWas = 0

Else

bestFind = veryEnd

bestFindWas = veryEnd

End If

For i = 1 To numSchOR

Set rng2 = rng.Duplicate

gotOne = False

Do While gotOne = False

gotOne = True

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = schOR(i)

.Wrap = wdFindStop

.Replacement.Text = ""

If searchUp = True Then

.Forward = False

Else

.Forward = True

End If

.MatchWildcards = False

.MatchCase = False

.MatchWholeWord = False

.Execute

End With

If checkCaps = True Then

For k = 1 To Len(schOR(i))

c = Mid(schOR(i), k, 1)

If UCase(c) = c Then

If c <> Mid(rng2.Text, k, 1) Then gotOne = False \_

: Exit For

End If

Next k

End If

DoEvents

Loop

If gotOne = True Then

If searchUp = True Then

If rng2.Start > bestFind Then

bestFind = rng2.Start

nearestItem = i

nearestItemStart = rng2.Start

End If

Else

If rng2.Start < bestFind Then

bestFind = rng2.Start

nearestItem = i

nearestItemStart = rng2.Start

End If

End If

End If

Next i

If bestFind = bestFindWas Then

Beep

Else

Selection.Start = nearestItemStart

Selection.End = Selection.Start + Len(schOR(nearestItem))

End If

Application.ScreenUpdating = True

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseEnd

rng.Select

Exit Sub

Else

' Search using the least common 'AND' item

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySearch

.Wrap = wdFindStop

.Replacement.Text = ""

If searchUp = True Then

.Forward = False

Else

.Forward = True

End If

.MatchWildcards = False

.MatchCase = False

.MatchWholeWord = False

.Execute

DoEvents

End With

Do While rng.Find.Found = True

rng.MoveStart wdWord, -myRange

rng.MoveEnd wdWord, myRange

thisText = rng.Text

deffoGood = True

' Check each of the AND items,

' e.g. (Smith\_Brown)+(2004)+(et al)

For i = 1 To numSch

sc = sch(i)

If InStr(sc, "\_") = 0 Then

' This is an AND item

If checkCaps = False Then

If InStr(LCase(thisText), LCase(sch(i))) = 0 Then

deffoGood = False

Exit For

End If

Else

' Where is this item in the selected text?

myPos = InStr(LCase(thisText), LCase(sch(i)))

For k = 1 To Len(sch(i))

c = Mid(sch(i), k, 1)

If UCase(c) = c And myPos + k > 1 Then

' If the capital letter isn't capital in the text

' then it's a no-no

If c <> Mid(thisText, myPos + k - 1, 1) Then

deffoGood = False

Exit For

End If

End If

Next k

End If

Else

' This is an OR item, e.g. "Smith\_Brown"

gotOne = False

wd = Split("\_" & sch(i), "\_")

numWds = UBound(wd)

For j = 1 To numWds

If checkCaps = False Then

If InStr(LCase(thisText), LCase(wd(j))) > 0 Then

gotOne = True

Exit For

End If

Else

' Check the caps match

myPos = InStr(LCase(thisText), LCase(wd(j)))

If myPos > 0 Then

gotOne = True

For k = 1 To Len(wd(j))

c = Mid(wd(j), k, 1)

If UCase(c) = c And myPos + k > 1 Then

If c <> Mid(thisText, myPos + k - 1, 1) Then

gotOne = False

Exit For

End If

End If

Next k

End If

End If

Next j

If gotOne = False Then deffoGood = False

End If

Next i

If deffoGood = True Then

' Now decide what extent of the text to display selected

mk = Split("\_" & Replace(LCase(theMultiSearch), "+", "\_"), "\_")

numWds = UBound(mk)

allText = LCase(rng)

firstChar = Len(rng)

lastChar = 0

For j = 1 To numWds

myLeft = InStr(allText, LCase(mk(j)))

myRight = 0

If myLeft > 0 Then

myRight = myLeft + Len(mk(j))

If myLeft < firstChar Then firstChar = myLeft

If myRight > lastChar Then lastChar = myRight

End If

DoEvents

Next j

rng.End = rng.Start + lastChar - 1

rng.MoveStart , firstChar - 1

rng.Select

If selStartNow = Selection.Start \_

And selEndNow = Selection.End Then

If searchUp = True Then

Selection.Collapse wdCollapseStart

Selection.MoveEnd wdWord, -myRange - 1

Else

Selection.Collapse wdCollapseEnd

Selection.MoveStart wdWord, myRange + 1

End If

GoTo restart

End If

Exit Sub

End If

If searchUp = True Then

rng.Collapse wdCollapseStart

Else

rng.Collapse wdCollapseEnd

End If

wasStart = rng.Start

rng.Find.Execute

DoEvents

If rng.Start = wasStart Then

If searchUp = True Then

rng.MoveStart wdWord, -1

rng.Collapse wdCollapseStart

Else

rng.MoveEnd wdWord, 1

rng.Collapse wdCollapseEnd

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

End If

lastTimeStart = rng.Start

Loop

End If

Beep

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub MultiSearchUp()**

' Paul Beverley - Version 28.05.22

' Searches for a set of words in proximity

myRange = 15

searchUp = True

Dim freq(8) As Integer

CR = vbCr: CR2 = CR & CR

Application.ScreenUpdating = False

On Error GoTo ReportIt

restart:

Set rng = Selection.Range.Duplicate

If rng.Start = rng.End Then rng.Expand wdParagraph

DoEvents

posPlus = InStr(rng, "+")

If posPlus > 1 And rng.Words.Count < 20 Then

myTest = Mid(rng, posPlus - 1, 3)

If LCase(myTest) <> UCase(myTest) Then

pbMultiSearch = Replace(rng.Text, vbCr, "")

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

End If

posOR = InStr(rng, "\_")

If posOR > 1 And rng.Words.Count < 20 Then

myTest = Mid(rng, posOR - 1, 3)

If LCase(myTest) <> UCase(myTest) Then

If Right(pbMultiSearch, 1) = "\_" And \_

LCase(theMultiSearch) = theMultiSearch Then

Beep

myResponse = MsgBox("Your criterion is all lowercase." \_

& CR2 & "Only capital letters are case-checked.", vbOKOnly, "MultiSearch")

Exit Sub

End If

pbMultiSearch = Replace(rng.Text, vbCr, "")

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

End If

' Check for a null search criterion

If pbMultiSearch = "" Then

Beep

myResponse = MsgBox("Sorry, I've forgotten what I was looking for!" \_

& CR2 & "Please reload.", vbOKOnly, "MultiSearch")

Application.ScreenUpdating = True

Exit Sub

End If

theMultiSearch = pbMultiSearch

If Right(pbMultiSearch, 1) = "\_" Then

theMultiSearch = Left(pbMultiSearch, Len(pbMultiSearch) - 1)

checkCaps = True

If LCase(theMultiSearch) = theMultiSearch Then

Beep

myResponse = MsgBox("Your criterion is all lowercase." \_

& CR2 & "Only capital letters are case-checked.", vbOKOnly, "pbMultiSearch")

End If

Else

checkCaps = False

End If

' Search for the words

sch = Split("+" & theMultiSearch, "+")

numSch = UBound(sch)

inText = True

If rng.Information(wdInEndnote) = True Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

inText = False

End If

If rng.Information(wdInFootnote) = True Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

inText = False

End If

If inText = True Then Set rng = ActiveDocument.Content

If searchUp = True Then

rng.End = Selection.Start

allTextToEnd = LCase(rng.Text)

totLen = Len(allTextToEnd)

Else

rng.Start = Selection.End

allTextToEnd = LCase(rng.Text)

totLen = Len(allTextToEnd)

End If

gottaSingle = False

doSch = 1

lowestFreq = 99999

numORs = 0

For i = 1 To numSch

sc = LCase(sch(i))

freq(i) = Len(Replace(allTextToEnd, sc, sc & "!")) - totLen

If InStr(sc, "\_") = 0 Then

gottaSingle = True

If freq(i) = 0 Then

Beep

Application.ScreenUpdating = True

Exit Sub

End If

If freq(i) <= lowestFreq Then lowestFreq = freq(i): doSch = i

Else

numORs = numORs + 1

If numORs > 1 Then

Beep

Application.ScreenUpdating = True

myResponse = MsgBox("Sorry, the search criterion is too complex:" \_

& CR2 & pbMultiSearch, vbOKOnly, "MultiSearch")

Exit Sub

End If

End If

DoEvents

Next i

mySearch = sch(doSch)

inText = True

If rng.Information(wdInEndnote) = True Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

inText = False

End If

If rng.Information(wdInFootnote) = True Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

inText = False

End If

If inText = True Then Set rng = ActiveDocument.Content

If searchUp = True Then

rng.End = Selection.Start

Else

rng.Start = Selection.End

End If

selStartNow = Selection.Start

selEndNow = Selection.End

veryEnd = rng.End

' Search just as an 'OR' function

If gottaSingle = False Then

schOR = Split("\_" & theMultiSearch, "\_")

numSchOR = UBound(schOR)

If searchUp = True Then

bestFind = 0

bestFindWas = 0

Else

bestFind = veryEnd

bestFindWas = veryEnd

End If

For i = 1 To numSchOR

Set rng2 = rng.Duplicate

gotOne = False

Do While gotOne = False

gotOne = True

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = schOR(i)

.Wrap = wdFindStop

.Replacement.Text = ""

If searchUp = True Then

.Forward = False

Else

.Forward = True

End If

.MatchWildcards = False

.MatchCase = False

.MatchWholeWord = False

.Execute

End With

If checkCaps = True Then

For k = 1 To Len(schOR(i))

c = Mid(schOR(i), k, 1)

If UCase(c) = c Then

If c <> Mid(rng2.Text, k, 1) Then gotOne = False \_

: Exit For

End If

Next k

End If

DoEvents

Loop

If gotOne = True Then

If searchUp = True Then

If rng2.Start > bestFind Then

bestFind = rng2.Start

nearestItem = i

nearestItemStart = rng2.Start

End If

Else

If rng2.Start < bestFind Then

bestFind = rng2.Start

nearestItem = i

nearestItemStart = rng2.Start

End If

End If

End If

Next i

If bestFind = bestFindWas Then

Beep

Else

Selection.Start = nearestItemStart

Selection.End = Selection.Start + Len(schOR(nearestItem))

End If

Application.ScreenUpdating = True

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseEnd

rng.Select

Exit Sub

Else

' Search using the least common 'AND' item

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySearch

.Wrap = wdFindStop

.Replacement.Text = ""

If searchUp = True Then

.Forward = False

Else

.Forward = True

End If

.MatchWildcards = False

.MatchCase = False

.MatchWholeWord = False

.Execute

DoEvents

End With

Do While rng.Find.Found = True

rng.MoveStart wdWord, -myRange

rng.MoveEnd wdWord, myRange

thisText = rng.Text

deffoGood = True

' Check each of the AND items,

' e.g. (Smith\_Brown)+(2004)+(et al)

For i = 1 To numSch

sc = sch(i)

If InStr(sc, "\_") = 0 Then

' This is an AND item

If checkCaps = False Then

If InStr(LCase(thisText), LCase(sch(i))) = 0 Then

deffoGood = False

Exit For

End If

Else

' Where is this item in the selected text?

myPos = InStr(LCase(thisText), LCase(sch(i)))

For k = 1 To Len(sch(i))

c = Mid(sch(i), k, 1)

If UCase(c) = c And myPos + k > 1 Then

' If the capital letter isn't capital in the text

' then it's a no-no

If c <> Mid(thisText, myPos + k - 1, 1) Then

deffoGood = False

Exit For

End If

End If

Next k

End If

Else

' This is an OR item, e.g. "Smith\_Brown"

gotOne = False

wd = Split("\_" & sch(i), "\_")

numWds = UBound(wd)

For j = 1 To numWds

If checkCaps = False Then

If InStr(LCase(thisText), LCase(wd(j))) > 0 Then

gotOne = True

Exit For

End If

Else

' Check the caps match

myPos = InStr(LCase(thisText), LCase(wd(j)))

If myPos > 0 Then

gotOne = True

For k = 1 To Len(wd(j))

c = Mid(wd(j), k, 1)

If UCase(c) = c And myPos + k > 1 Then

If c <> Mid(thisText, myPos + k - 1, 1) Then

gotOne = False

Exit For

End If

End If

Next k

End If

End If

Next j

If gotOne = False Then deffoGood = False

End If

Next i

If deffoGood = True Then

' Now decide what extent of the text to display selected

mk = Split("\_" & Replace(LCase(theMultiSearch), "+", "\_"), "\_")

numWds = UBound(mk)

allText = LCase(rng)

firstChar = Len(rng)

lastChar = 0

For j = 1 To numWds

myLeft = InStr(allText, LCase(mk(j)))

myRight = 0

If myLeft > 0 Then

myRight = myLeft + Len(mk(j))

If myLeft < firstChar Then firstChar = myLeft

If myRight > lastChar Then lastChar = myRight

End If

DoEvents

Next j

rng.End = rng.Start + lastChar - 1

rng.MoveStart , firstChar - 1

rng.Select

If selStartNow = Selection.Start \_

And selEndNow = Selection.End Then

If searchUp = True Then

Selection.Collapse wdCollapseStart

Selection.MoveEnd wdWord, -myRange - 1

Else

Selection.Collapse wdCollapseEnd

Selection.MoveStart wdWord, myRange + 1

End If

GoTo restart

End If

Exit Sub

End If

If searchUp = True Then

rng.Collapse wdCollapseStart

Else

rng.Collapse wdCollapseEnd

End If

wasStart = rng.Start

rng.Find.Execute

DoEvents

If rng.Start = wasStart Then

If searchUp = True Then

rng.MoveStart wdWord, -1

rng.Collapse wdCollapseStart

Else

rng.MoveEnd wdWord, 1

rng.Collapse wdCollapseEnd

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

End If

lastTimeStart = rng.Start

Loop

End If

Beep

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub MultiSearchLoader()**

' Paul Beverley - Version 31.05.22

' Loads text segments into MultiSearch

restartKeys = "1."

ANDkeys = "2+"

ORkeys = "3-/"

capsKeys = "6\*"

editKeys = "0"

myListName = "zzzSwitchList"

Set rng = Selection.Range.Duplicate

If rng.Start = rng.End Then rng.Expand wdParagraph

posPlus = InStr(rng, "+")

If posOR > 1 And rng.Words.Count < 20 Then

myTest = Mid(rng, posOR - 1, 3)

If LCase(myTest) <> UCase(myTest) Then

If Right(pbMultiSearch, 1) = "\_" And \_

LCase(theMultiSearch) = theMultiSearch Then

Beep

myResponse = MsgBox("Your criterion is all lowercase." \_

& CR2 & "Only capital letters are case-checked.", vbOKOnly, "MultiSearch")

Exit Sub

End If

pbMultiSearch = Replace(rng.Text, vbCr, "")

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

End If

posOR = InStr(rng, "\_")

If posOR > 1 And rng.Words.Count < 20 Then

myTest = Mid(rng, posOR - 1, 3)

If LCase(myTest) <> UCase(myTest) Then

pbMultiSearch = Replace(rng.Text, vbCr, "")

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

End If

theMultiSearch = pbMultiSearch

If Right(pbMultiSearch, 1) = "\_" Then

theMultiSearch = Left(pbMultiSearch, Len(pbMultiSearch) - 1)

checkCaps = True

capsText = "Ignore caps ["

Else

checkCaps = False

capsText = "Check Caps ["

End If

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

CR = vbCr: CR2 = CR & CR

Dim sch(4) As String

newText = LCase(Selection.Text)

Do

myText = InputBox("RESTART [" & restartKeys & "]" \_

& CR2 & "AND [" & ANDkeys & "]" & CR2 & "OR [" & \_

ORkeys & "]" & CR2 & capsText & capsKeys & "]" & \_

CR2 & "EDIT [" & editKeys & "]" & CR, \_

"MultiSearchLoader", pbMultiSearch)

DoEvents

If myText = "" Then Beep: Exit Sub

Loop Until InStr(restartKeys & ANDkeys & ORkeys & capsKeys \_

& editKeys, myText) > 0 Or Len(myText) > 3

If Len(myText) = 1 Then

If InStr(ANDkeys, myText) > 0 Then pbMultiSearch = pbMultiSearch & \_

"+" & newText

If InStr(ORkeys, myText) > 0 Then pbMultiSearch = pbMultiSearch & \_

"\_" & newText

If InStr(restartKeys, myText) > 0 Then pbMultiSearch = newText

If InStr(capsKeys, myText) > 0 Then

If checkCaps = True Then

If Right(pbMultiSearch, 1) = "\_" Then \_

pbMultiSearch = Left(pbMultiSearch, Len(pbMultiSearch) - 1)

Else

pbMultiSearch = pbMultiSearch & "\_"

End If

If pbMultiSearch = LCase(pbMultiSearch) And checkCaps = False Then

myResponse = \_

MsgBox("But your search criterion has no capital letters!" \_

& CR2 & "Edit the criterion?", \_

vbQuestion + vbYesNo, pbMultiSearch)

If myResponse <> vbYes Then Exit Sub

myText = editKeys

Else

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

End If

Exit Sub

End If

If InStr(editKeys, myText) > 0 Then

' Try to find a suitable file

' Find the zzSwitchList file

gottaList = False

For i = 1 To Documents.Count

Set dcu = Documents(i)

If InStr(dcu.Name, myListName) > 0 Then

gottaList = True

dcu.Activate

Exit For

End If

If Len(dcu.Content) < 200 Then

gottaList = True

dcu.Activate

Exit For

End If

Next i

If gottaList = False Then Documents.Add

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:=pbMultiSearch & vbCr

Selection.MoveLeft , 1

End If

Exit Sub

End If

If Len(myText) > 3 Then

pbMultiSearch = myText

Call MultiSearch

Else

Beep

End If

End Sub

**Sub FindThisOrThat()**

' Paul Beverley - Version 24.02.14

' Find the next occurrence of certain specific words

Dim v As Variable

varsExist = False

For Each v In ActiveDocument.Variables

If v.Name = "findText" Then varsExist = True: Exit For

Next v

If varsExist = False Then

Call FindThisOrThatSetUp

Exit Sub

End If

listWds = Trim(ActiveDocument.Variables("findText"))

If InStr(listWds, "|") > 0 Then

de = "|"

Else

de = ","

End If

numWds = Len(listWds) - Len(Replace(listWds, de, "")) + 1

ReDim findWds(numWds) As String

findWds = Split(listWds, de)

Selection.Collapse wdCollapseEnd

Set rng = ActiveDocument.Content

hereStart = Selection.Start

' Find the nearest of the required words

nearest = ActiveDocument.Range.End

wdNum = -1

For i = 0 To numWds - 1

rng.Start = hereStart

rng.End = nearest

w = findWds(i)

If Left(w, 1) = "~" Then

w = Mid(w, 2)

wc = True

Else

wc = False

End If

If Left(w, 1) = "$" Then

w = Mid(w, 2)

sens = True

Else

sens = False

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = w

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchCase = sens

.MatchWildcards = wc

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

wdPos = rng.Start

If wdPos < nearest And wdPos <> hereStart Then

nearest = wdPos

wdNum = i

End If

Next i

If wdNum >= 0 Then

w = findWds(wdNum)

If Left(w, 1) = "~" Then

w = Mid(w, 2)

wc = True

Else

wc = False

End If

If Left(w, 1) = "$" Then

w = Mid(w, 2)

sens = True

Else

sens = False

End If

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = w

.Wrap = False

.Replacement.Text = ""

.MatchCase = sens

.MatchWildcards = wc

.Forward = True

.Execute

End With

Else

Beep

End If

End Sub

**Sub FindThisOrThatSetUp()**

' Paul Beverley - Version 22.02.14

' Set-up for finding the next occurrence of certain specific words

Dim v As Variable

varsExist = False

For Each v In ActiveDocument.Variables

If v.Name = "findText" Then varsExist = True: Exit For

Next v

If varsExist = False Then

ActiveDocument.Variables.Add "findText", " "

textNow = ""

Else

textNow = ActiveDocument.Variables("findText")

End If

t = InputBox("Search for?", "Find This Or That", textNow)

If t = "" Then Exit Sub

ActiveDocument.Variables("findText") = t

Call FindThisOrThat

End Sub

**Sub PrepareToReplaceDown()**

' Paul Beverley - Version 10.04.21

' Copies selected text into the F&R box

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

myText = Selection

myText = Replace(myText, vbCr, "^p")

myText = Replace(myText, vbTab, "^t")

If Asc(myText) <> 32 Then myText = Trim(myText)

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText

.Forward = True

.Replacement.Text = myText

.MatchWildcards = False

.MatchWholeWord = False

.MatchCase = True

End With

Selection.End = Selection.Start

On Error Resume Next

Application.Run MacroName:="EditReplace"

End Sub

**Sub PrepareToReplaceDownTCoff()**

' Paul Beverley - Version 10.04.21

' Copy text into the F&R box

tcColour = True

myColour = 200

ActiveDocument.TrackRevisions = False

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

myText = Selection

If Asc(myText) <> 32 Then myText = Trim(myText)

myText = Replace(myText, vbCr, "^p")

myText = Replace(myText, vbTab, "^t")

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Forward = True

.Text = myText

.MatchWildcards = False

.MatchWholeWord = False

.Replacement.Text = myText

.MatchCase = True

End With

Selection.End = Selection.Start

If tcColour = True Then

With ActiveDocument.Background.Fill

.ForeColor.RGB = RGB(255, 255, myColour)

.Visible = msoTrue

.Solid

End With

End If

On Error Resume Next

Application.Run MacroName:="EditReplace"

End Sub

**Sub PrepareToReplaceFromTop()**

**'** Paul Beverley - Version 10.04.21

' Copy text into the F&R box from top

myText = Selection

If Asc(myText) <> 32 Then myText = Trim(myText)

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText

.Replacement.Text = myText

.MatchWildcards = False

.MatchWholeWord = False

End With

Selection.HomeKey Unit:=wdStory

On Error Resume Next

Application.Run MacroName:="EditReplace"

End Sub

**Sub PrepareToReplaceWithMarker()**

' Paul Beverley - Version 10.04.21

' Copy text into the F&R box from top leaving marker

myText = Selection

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

If Asc(myText) <> 32 Then myText = Trim(myText)

Selection.Start = Selection.End

Selection.TypeText "[[["

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText

.Forward = True

.Replacement.Text = myText

.MatchWildcards = False

.MatchWholeWord = False

End With

Selection.HomeKey Unit:=wdStory

On Error Resume Next

Application.Run MacroName:="EditReplace"

End Sub

**Sub FindInDeletedText()**

' Paul Beverley - Version 07.09.17

' Searches only the deleted (track changed) text

myText = InputBox("Find?", "Revision finder", Selection.Find.Text)

Selection.Find.Text = myText

Set rng = ActiveDocument.range(0, Selection.End)

thisRV = rng.Revisions.Count

If thisRV = 0 Then thisRV = 1

For j = thisRV To ActiveDocument.Revisions.Count

Set rv = ActiveDocument.Revisions(j)

If rv.Type = wdRevisionDelete Then

If InStr(LCase(rv.range.Text), LCase(myText)) > 0 Then

rv.range.Select

Exit Sub

End If

End If

DoEvents

Next j

Beep

End Sub

**Sub SmartFinder()**

' Paul Beverley - Version 15.01.21

' Finds this text/note/page/etc, etc immediately.

A = ""

A = A & "w = web address|"

A = A & "a = acronym|"

A = A & "d/D = date|"

A = A & "h = heading|"

A = A & "|"

A = A & "p = postcode|"

A = A & "Z = zipcode|"

A = A & "y = year [2014]|"

A = A & "< = tag <A>, <Cap>|"

A = A & " = |"

A = A & " = |"

A = A & " = |"

A = A & " = |"

endPos = InStr(A, "= |")

If endPos > 0 Then A = Left(A, endPos - 3)

myPrompt = Replace(A, "|", vbCr)

existingFind = Selection.Find.Text

startMyText = Selection.Start

endMyText = Selection.End

selText = Selection

' If nothing selected, pick up the current word

noTextSelected = (Selection.End = Selection.Start)

If noTextSelected Then

cursorPosn = Selection.Start

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

selText = Selection

' Ends of the current word

startMyText = Selection.Start

endMyText = Selection.End

' Put the cursor back where it was

Selection.End = cursorPosn

Selection.Start = cursorPosn

End If

selText = Trim(selText)

' For safety, first clear all 'funny' finds down

With Selection.Find

.Wrap = wdFindStop

.Forward = True

.Text = ""

.MatchAllWordForms = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

End With

inputCommand:

myText = InputBox(myPrompt, "Smart finder", selText)

If Len(myText) = 0 Then Exit Sub

myText = Replace(myText, Chr(13), "")

' Strip accidental typing of '#' before CR

If Right(myText, 1) = "#" Then myText = Replace(myText, "#", "")

' Default settings for all searches

mvStart = 0

mvEnd = 0

myReplace = ""

myStyle = ""

isBold = False

isItalic = False

isSuper = False

isSub = False

isSmalls = False

thisSize = 0

nmlSize = 0

thisFont = ""

nmlFont = ""

thisColour = 0

nmlColour = 0

myCase = False

sndsLike = False

wholeWds = False

allForms = False

If Len(myText) > 1 Then GoTo moreText

' Single-letter commands

wc = True

myReplace = ""

Select Case myText

Case "a": myFind = "[A-Z]{2,}" ' acronyms (see also x and X)

Case "B": myFind = "[ABC][DCE]" ' BC/AD/CE/BCE

Case "b": myFind = "[abc][dce]" ' BC/AD/CE/BCE in small caps

wasSmalls = 0: isSmalls = True

Case "d": myFind = "[123][0-9] [A-O][a-z]@ [0-9]{2,4}" ' dates

Case "D": myFind = "[0-9]{1,2}.[0-9]{1,2}.[0-9]{2,4}" ' dates

Case "e" ' Do nowt and then later jump back to input function

' f is for font attributes - see below

' h is for headings - see below

Case "E": myFind = "[A-Z][ \-,][0-9]{4}"

Case "i": myFind = "[A-Z]. [A-Z][a-z]" ' People's initials dotted

Case "I": myFind = "[A-Z]{1,} [A-Z][a-z]" ' People's initials, no dots

Case "m": myFind = "[ .^13][a-zA-Z]@\@[a-zA-Z]@.[a-zA-Z]{1,}" ' emails

Case "n": myFind = "[0-9]{1,}" ' numbers

' M is macro select, below

Case "s": myFind = "^13[0-9]{1,}.[0-9]{1,}": mvStart = 1 ' section numbers

Case "p": myFind = "[A-Z]{1,2}[0-9]{1,2} [0-9][A-Z]{2}" ' postcodes

Case "u": myFind = "[0-9 ^0160" & ChrW(8201) & "][kcmM][NJAVmg]>" ' units

Case "w": myFind = "[wt]{2}[wp][.:][a-z/]{2,}" ' web addresses

' x is expand acronym below

' X is expand acronym, case insensitively below

Case "y": myFind = "[12][0-9]{3}[!0-9]": mvEnd = -1 ' years

Case "z": myFind = "<[0-9]{5}>"

Case "Z": myFind = "[A-Z][0-9][A-Z] [0-9][A-Z][0-9]>"

Case "<": myFind = "\<[ABDCapHN]{1,3}\>"

Case "(": myFind = "\([0-9.]@\)" ' Number in brackets: (3.16), (12.2)

Case ")": myFind = "^13([0-9]{1,2}). ": myReplace = "^p\1^t"

Case "[": myFind = "[[[[[": wc = False

Case "-":

hyphenPos = InStr(selText, "-")

spacePos = InStr(selText, " ")

enPos = InStr(selText, ChrW(8211))

markerPos = hyphenPos + spacePos + enPos

If markerPos = 0 Then

If Len(Selection) = 1 Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

selText = Selection

End If

Beep

wd1 = InputBox("First word?", "FindAnything", selText)

If wd1 = selText Then

Selection.MoveRight wdWord, 1

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

wd2 = Selection

If wd2 = "-" Or wd2 = ChrW(8211) Then

Selection.MoveRight wdWord, 1

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

wd2 = Selection

End If

Else

wd2 = Mid(selText, Len(wd1) + 1)

End If

Else

wd1 = Left(selText, markerPos - 1)

wd2 = Mid(selText, markerPos + 1)

End If

If Len(wd1) = 0 Then

Beep

Exit Sub

End If

myFind = Left(wd1, Len(wd1) - 1) & "[" & Right(wd1, 1) \_

& Left(wd2, 1) & "\- ^=]{2,3}" & Mid(wd2, 2)

wc = True

Case "A": myFind = Trim(Replace(selText, " ", ""))

allForms = True: wc = False

Case "S": myFind = Trim(selText): sndsLike = True: wc = False

Case "W": myFind = Trim(selText): wholeWds = True: wc = False

Case "x" ' Xpand, i.e. look for acronym expansion

myFind = "<"

abbrLength = Len(selText)

For i = 1 To abbrLength

myFind = myFind & Mid(selText, i, 1) & "[a-z]{1,}^32"

Next i

myFind = Left(myFind, Len(myFind) - 3)

Case "X" ' Xpand, i.e. look for acronym expansion, either case

myFind = "<"

abbrLength = Len(selText)

For i = 1 To abbrLength

If i = 4 Then Exit For

myChar = Mid(selText, i, 1)

myChar = "[" & LCase(myChar) & UCase(myChar) & "]"

myFind = myFind & myChar & "[a-z]{1,}^32"

Next i

myFind = Left(myFind, Len(myFind) - 3)

Case "h": wc = False: myFind = "" ' A heading

myStyle = Selection.Range.Style

If myStyle = "Normal" Then

myStyle = ""

Selection.End = Selection.Start + 1

Selection.Font.Reset

nmlSize = Selection.Font.Size

nmlFont = Selection.Font.Name

nmlColour = Selection.Font.Color

wasBold = Selection.Font.Bold

wasItalic = Selection.Font.Italic

' undo the change, i.e. restore the text's original attributes

WordBasic.EditUndo

' Check the text's emphasis now

isBold = Selection.Font.Bold

isItalic = Selection.Font.Italic

thisSize = Selection.Font.Size

thisFont = Selection.Font.Name

thisColour = Selection.Font.Color

End If

Selection.Expand wdParagraph

If myStyle = "" And wasBold = isBold And wasItalic = isItalic \_

And thisSize = nmlSize And thisFont = nmlFont \_

And thisColour = nmlColour Then

myFind = "^p" & Left(Selection, 1)

End If

Selection.Start = Selection.End

Case "f": wc = False: myFind = "" ' Font attributes

If Selection.End = Selection.Start Then Selection.MoveEnd wdCharacter, 1

Selection.Font.Reset

nmlSize = Selection.Font.Size

nmlFont = Selection.Font.Name

nmlColour = Selection.Font.Color

wasBold = Selection.Font.Bold

wasItalic = Selection.Font.Italic

wasSuper = Selection.Font.Superscript

wasSub = Selection.Font.Subscript

wasSmalls = Selection.Font.SmallCaps

' undo the change, i.e. restore the text's original attributes

WordBasic.EditUndo

' Check the text's emphasis now

isBold = Selection.Font.Bold

isItalic = Selection.Font.Italic

isSuper = Selection.Font.Superscript

isSub = Selection.Font.Subscript

isSmalls = Selection.Font.SmallCaps

isStrike = Selection.Font.StrikeThrough

isDStrike = Selection.Font.DoubleStrikeThrough

isUnderline = Selection.Font.Underline

thisSize = Selection.Font.Size

thisFont = Selection.Font.Name

thisColour = Selection.Font.Color

Case Else

' A number 1 to 9

myNumber = Val(Chr(Asc(myText)))

If myNumber > 0 And myNumber < 10 Then

' A caption/heading with the same first three characters

Selection.HomeKey Unit:=wdLine

Selection.MoveEnd wdCharacter, myNumber

myFind = "[^t^11^12^13]" & Replace(Selection, "<", "\<")

mvStart = 1

wc = True

Else

' Absolutely every other input gives a straight non-WC find

myFind = myText

wc = False

End If

End Select

If myText = "e" Then selText = Trim(existingFind): GoTo inputCommand

GoTo goFind

' See if it's a page, fnote, enote or section number

moreText:

Debug.Print selText

Debug.Print myText

startChar = Left(LCase(myText), 1)

lastChar = Asc(Right(myText, 1))

If lastChar > 47 And lastChar < 58 And Len(myText) <= 4 Then

' The final chacter is a number

isACode = True

Select Case startChar

Case "p"

Selection.GoTo what:=wdGoToPage, Count:=Mid(myText, 2)

Case "n"

Selection.GoTo what:=wdGoToFootnote, Count:=Mid(myText, 2)

Case "f"

Selection.GoTo what:=wdGoToFootnote, Count:=Mid(myText, 2)

Case "e"

Selection.GoTo what:=wdGoToEndnote, Count:=Mid(myText, 2)

Case "c"

Selection.GoTo what:=wdGoToComment, Count:=Mid(myText, 2)

Case "s"

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13\>^t]" & Mid(myText, 2)

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Wrap = wdFindStop

.Forward = True

.Execute

End With

If rng.Find.Found = True Then

rng.Select

Selection.MoveStart wdCharacter, 1

Else

Beep

Selection.Start = endMyText

End If

Case Else

isACode = False

End Select

If isACode = True Then

If ActiveDocument.Footnotes.Count >= 1 And InStr("fn", startChar) > 0 \_

Then ActiveDocument.ActiveWindow.View.SeekView = wdSeekFootnotes

If ActiveDocument.Endnotes.Count >= 1 And startChar = "e" Then \_

ActiveDocument.ActiveWindow.View.SeekView = wdSeekEndnotes

Exit Sub

End If

End If

' If not a special command, then it's ordinary text to find

If Len(myText) = 2 Then

myCase = True

Select Case myText

Case "ap": myText = "Appendix "

Case "re": myText = "References^p": myCase = False

Case "fi": myText = "Figure"

Case "[": myText = "[[[[["

Case "se": myText = "| Spelling Errors"

Case "mm": myText = "1.^0160Bookmarks"

Case "ii": myText = "(Preparing your brief or style guide)"

Case "op": myText = "When I want to change"

Case "st": myText = "What is this document?"

End Select

End If

myFind = myText

If myReplace > "" Then myReplace = myText

wc = False

goFind:

Selection.Start = Selection.End

' Go and look for the word

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

If Len(myStyle) > 0 And myStyle <> "Normal" Then .Style = myStyle

.Replacement.Text = myReplace

.Forward = True

.MatchCase = myCase

.MatchWildcards = wc

.MatchWholeWord = wholeWds

.MatchSoundsLike = sndsLike

.MatchAllWordForms = allForms

If isBold Then .Font.Bold = True

If isItalic Then .Font.Italic = True

If isSuper Then .Font.Superscript = True

If isSub Then .Font.Subscript = True

If isSmalls Then .Font.SmallCaps = True

If isStrike Then .Font.StrikeThrough = True

If isDStrike Then .Font.DoubleStrikeThrough = True

If isUnderline Then .Font.Underline = True

If thisSize <> nmlSize Then .Font.Size = thisSize

If thisFont <> nmlFont Then .Font.Name = thisFont

If thisColour <> nmlColour Then .Font.Color = thisColour

.Wrap = wdFindStop

.Forward = True

.Execute

End With

If (Selection.Start = 0 Or Selection.Start = endMyText Or \_

Selection.End = cursorPosn) And wc = False Then

'If Selection.Start = endMyText Or Selection.End = cursorPosn Then

Beep

Selection.End = startMyText

Selection.Start = startMyText

With Selection.Find

.Forward = False

.Execute

End With

Else

With Selection.Find

.Forward = True

.Execute

End With

If Selection.Start > hereNow Then

With Selection.Find

.Forward = False

.Execute

End With

End If

End If

If mvStart > 0 Then Selection.MoveStart wdCharacter, mvStart

If mvEnd > 0 Then Selection.MoveEnd wdCharacter, mvEnd

With Selection.Find

If myText = "[" Then .Text = existingFind

.Forward = True

.Wrap = wdFindContinue

End With

End Sub

**Sub NavPaneWidth()**

' Paul Beverley - Version 12.01.21

' Widens the navigation pane

myWidth = 400

Application.CommandBars("Navigation").Width = myWidth

End Sub

**Sub HighlightFindUp()**

' Paul Beverley - Version 27.01.22

' Selects the previous piece of highlighted text

nowColour = Selection.Range.HighlightColorIndex

mixCol = 9999999

myScreenOff = True

If nowColour = mixCol Then

Set rng = Selection.Range.Duplicate

rng.End = rng.Start + 1

nowColour = rng.HighlightColorIndex

End If

If Selection.Start <> Selection.End And nowColour = wdNoHighlight Then \_

Selection.Collapse wdCollapseStart

If Selection.Start <> Selection.End Then

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

' Find this colour only

findAny = False

Set rng = Selection.Range.Duplicate

rng.End = rng.Start + 1

myColour = rng.HighlightColorIndex

Do

rng.Start = rng.Start - 1

rng.End = rng.End - 1

nowColour = rng.HighlightColorIndex

DoEvents

Loop Until nowColour <> myColour Or rng.Start = 0

i = 1

doneBeep = False

Do

rng.Start = rng.Start - 1

rng.End = rng.End - 1

nowColour = rng.HighlightColorIndex

DoEvents

i = i + 1

If i Mod 100 = 0 Then

DoEvents

End If

If i > 1000 Then

If doneBeep = False Then

Beep

doneBeep = True

End If

If i Mod 100 = 0 Then StatusBar = "This takes time, sorry: " & Str(i)

End If

Loop Until nowColour = myColour Or rng.Start = 0

' Find where this colour stops

If nowColour = myColour Then

en = rng.End

Do

rng.Start = rng.Start - 1

rng.End = rng.End - 1

nowColour = rng.HighlightColorIndex

DoEvents

Loop Until nowColour <> myColour

rng.Start = rng.Start + 1

adsgf = rng.Start

rng.End = en

rng.Select

End If

Else

' Find any colour

Application.ScreenUpdating = False

findAny = True

Set rng = Selection.Range.Duplicate

Selection.MoveStart , -1

nowColour = Selection.Range.HighlightColorIndex

If nowColour > 0 Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindStop

.Forward = False

.Highlight = False

.Replacement.Text = ""

.Execute

DoEvents

If .Found = False Then

Beep

rng.Start = myEnd

End If

End With

DoEvents

rng.Collapse wdCollapseStart

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindStop

.Forward = False

.Highlight = True

.Replacement.Text = ""

.Execute

DoEvents

If .Found = False Then

Beep

rng.End = 0

End If

End With

DoEvents

End If

rng.Select

' Flash the range

Application.ScreenUpdating = True

For i = 1 To 2

DoEvents

myTime = Timer

Do

Loop Until Timer > myTime + 0.08

Selection.Collapse wdCollapseStart

DoEvents

myTime = Timer

Do

Loop Until Timer > myTime + 0.08

rng.Select

Next i

If findAny = True Then

Selection.Collapse wdCollapseStart

Else

rng.Select

End If

If rng.Start = 0 Then

Beep

Selection.Collapse wdCollapseStart

End If

' Clear up the F&R

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindContinue

End With

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub HighlightFindDown()**

' Paul Beverley - Version 27.01.22

' Selects the next piece of highlighted text

nowColour = Selection.Range.HighlightColorIndex

mixCol = 9999999

myEnd = ActiveDocument.Content.End

myScreenOff = True

If nowColour = mixCol Then

Set rng = Selection.Range.Duplicate

rng.End = rng.Start + 1

nowColour = rng.HighlightColorIndex

End If

If Selection.Start <> Selection.End And nowColour = wdNoHighlight Then \_

Selection.Collapse wdCollapseEnd

If Selection.Start <> Selection.End Then

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

' Find this colour only

findAny = False

Set rng = Selection.Range.Duplicate

rng.End = rng.Start + 1

myColour = rng.HighlightColorIndex

Do

rng.End = rng.End + 1

rng.Start = rng.Start + 1

nowColour = rng.HighlightColorIndex

DoEvents

Loop Until nowColour <> myColour Or rng.End = myEnd

i = 1

doneBeep = False

Do

rng.End = rng.End + 1

rng.Start = rng.Start + 1

nowColour = rng.HighlightColorIndex

i = i + 1

If i Mod 100 = 0 Then

DoEvents

End If

If i > 1000 Then

If doneBeep = False Then

Beep

doneBeep = True

End If

If i Mod 100 = 0 Then StatusBar = "This takes time, sorry: " & Str(i)

End If

Loop Until nowColour = myColour Or rng.End = myEnd

' Find where this colour stops

If nowColour = myColour Then

st = rng.Start

Do

rng.End = rng.End + 1

rng.Start = rng.Start + 1

nowColour = rng.HighlightColorIndex

DoEvents

Loop Until nowColour <> myColour

rng.End = rng.End - 1

rng.Start = st

rng.Select

End If

Else

' Find any colour

findAny = True

Set rng = Selection.Range.Duplicate

Selection.MoveEnd , 1

nowColour = Selection.Range.HighlightColorIndex

If nowColour > 0 Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindStop

.Forward = True

.Highlight = False

.Replacement.Text = ""

.Execute

DoEvents

If .Found = False Then

Beep

rng.Start = myEnd

End If

DoEvents

End With

rng.Collapse wdCollapseStart

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindStop

.Forward = True

.Highlight = True

.Replacement.Text = ""

.Execute

DoEvents

If .Found = False Then

Beep

rng.Start = myEnd

End If

DoEvents

End With

End If

rng.Select

' Flash the range

Application.ScreenUpdating = True

For i = 1 To 2

DoEvents

myTime = Timer

Do

Loop Until Timer > myTime + 0.08

Selection.Collapse wdCollapseStart

DoEvents

myTime = Timer

Do

Loop Until Timer > myTime + 0.08

rng.Select

Next i

If findAny = True Then Selection.Collapse wdCollapseEnd

' Clear up the F&R

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindContinue

End With

StatusBar = ""

Exit Sub

ReportIt:

On Error GoTo 0

Resume

End Sub

**Sub SelectNextHighlight()**

' Paul Beverley - Version 13.09.19

' Selects the next piece of highlighted text

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Replacement.Text = ""

.Forward = True

.Highlight = True

.MatchWildcards = False

.Execute

End With

If Selection.Find.Found = False Then

Selection.Collapse wdCollapseEnd

Beep

End If

End Sub

**Sub SelectPreviousHighlight()**

' Paul Beverley - Version 13.09.19

' Selects the previous piece of higlighted text

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Replacement.Text = ""

.Forward = False

.Highlight = True

.MatchWildcards = False

.Execute

End With

If Selection.Find.Found = False Then

Selection.Collapse wdCollapseEnd

Beep

End If

End Sub

**Sub FindColouredText()**

' Paul Beverley - Version 03.02.12

' Finds coloured text

Dim v As Variable, rng As Range

' Check what the background (black) colour is

Set rng = ActiveDocument.Content

rng.Collapse wdCollapseEnd

myBlack = rng.Font.Color

' Check for search colour variable

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "tColour" Then varExists = True: Exit For

Next v

If varExists = False Then ActiveDocument.Variables.Add "tColour", 0

searchColour = ActiveDocument.Variables("tColour")

initialPosition = Selection.End

' If no text is selected, search for next coloured bit

If Selection.Start = Selection.End Then GoTo FindNext

' If some text is selected, see what colour it is;

' then go find more text of that colour.

searchColour = Selection.Font.Color

If searchColour < 0 Then searchColour = 0

ActiveDocument.Variables("tColour") = searchColour

If searchColour > 0 Then GoTo FindNext

Selection.Start = Selection.End

' Go and find the next non-black colour

Set rng = ActiveDocument.Content

theEnd = rng.End

Do

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

rng.Start = Selection.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = myBlack

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

' Examine the character after the end of the find

rng2.Start = rng.End

rng2.End = rng.End + 1

' If the next bit is still black, find the end of that

Do While rng2.Font.Color = myBlack

rng.Collapse wdCollapseEnd

rng.Find.Execute

rng2.Start = rng.End

rng2.End = rng.End + 1

Loop

rng.Collapse wdCollapseEnd

rng.Find.Execute

rng.Collapse wdCollapseStart

rng.Start = rng.Start - 1

colourHere = rng.Font.Color

rng.Select

myResponse = MsgBox("This colour? (Cancel = any colour)", vbQuestion + vbYesNoCancel)

If myResponse = vbCancel Then

Selection.Collapse wdCollapseEnd

ActiveDocument.Variables("tColour") = 0

Exit Sub

End If

If rng.End = theEnd Then Exit Sub

Loop Until myResponse = vbYes

ActiveDocument.Variables("tColour") = colourHere

GoTo finish

FindNext:

If searchColour = 0 Then

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

rng.Start = Selection.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = myBlack

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If rng.Find.Found = True Then

' Examine the character after the end of the find

rng2.Start = rng.End

rng2.End = rng.End + 1

' If the next bit is still black, find the end of that

Do While rng2.Font.Color = myBlack

rng.Collapse wdCollapseEnd

rng.Find.Execute

rng2.Start = rng.End

rng2.End = rng.End + 1

Loop

rng.Collapse wdCollapseEnd

rng.Find.Execute

rng.Collapse wdCollapseStart

End If

GoTo finish

Else

Set rng = ActiveDocument.Content

rng.Start = Selection.End

foundHlight = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = searchColour

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

End If

rng.Select

finish:

If searchColour = 0 Then

rng2.End = rng.Start

rng2.Select

Else

Set rng2 = rng.Duplicate

End If

' Flash the range

For i = 1 To 3

DoEvents

myTime = Timer

Do

Loop Until Timer > myTime + 0.06

Selection.Collapse wdCollapseStart

DoEvents

myTime = Timer

Do

Loop Until Timer > myTime + 0.08

rng2.Select

Next i

Selection.Collapse wdCollapseEnd

If Selection.End = initialPosition Then Beep

End Sub

**Sub FindColouredTextUp()**

' Paul Beverley - Version 03.02.22

' Finds coloured text upwards

Dim v As Variable, rng As Range

' Check what the background (black) colour is

Set rng = ActiveDocument.Content

rng.Collapse wdCollapseEnd

myBlack = rng.Font.Color

' Check for search colour variable

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "tColour" Then varExists = True: Exit For

Next v

If varExists = False Then ActiveDocument.Variables.Add "tColour", 0

searchColour = ActiveDocument.Variables("tColour")

initialPosition = Selection.Start

' If no text is selected, search for next coloured bit

If Selection.Start = Selection.End Then GoTo FindNext

' If some text is selected, see what colour it is;

' then go find more text of that colour.

searchColour = Selection.Font.Color

If searchColour < 0 Then searchColour = 0

ActiveDocument.Variables("tColour") = searchColour

If searchColour > 0 Then GoTo FindNext

Selection.Start = Selection.End

' Go and find the previous non-black colour

Set rng = ActiveDocument.Content

theEnd = rng.End

Do

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

rng.End = Selection.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = myBlack

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

' Examine the character before the beginning of the find

rng2.Start = rng.Start - 1

rng2.End = rng.Start

' If the next bit is still black, find the end of that

Do While rng2.Font.Color = myBlack

rng.Collapse wdCollapseStart

rng.Find.Execute

rng2.Start = rng.Start - 1

rng2.End = rng.Start

Loop

rng.Collapse wdCollapseStart

rng.Find.Execute

rng.Collapse wdCollapseEnd

rng.End = rng.Start + 1

colourHere = rng.Font.Color

rng.Select

myResponse = MsgBox("This colour? (Cancel = any colour)", vbQuestion + vbYesNoCancel)

Selection.Collapse wdCollapseStart

If myResponse = vbCancel Then

ActiveDocument.Variables("tColour") = 0

Exit Sub

End If

If rng.End = theEnd Then Exit Sub

Loop Until myResponse = vbYes

ActiveDocument.Variables("tColour") = colourHere

GoTo finish

FindNext:

If searchColour = 0 Then

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

rng.End = Selection.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = myBlack

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = False

.MatchWildcards = False

.Execute

End With

If rng.Find.Found = True Then

' Examine the character after the end of the find

rng2.Start = rng.Start - 1

rng2.End = rng.Start

' If the next bit is still black, find the end of that

Do While rng2.Font.Color = myBlack

rng.Collapse wdCollapseStart

rng.Find.Execute

rng2.Start = rng.Start - 1

rng2.End = rng.Start

Loop

rng.Collapse wdCollapseStart

rng.Find.Execute

rng.Collapse wdCollapseEnd

End If

GoTo finish

Else

Set rng = ActiveDocument.Content

rng.End = Selection.End

foundHlight = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = searchColour

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = False

.MatchWildcards = False

.Execute

End With

End If

finish:

If searchColour = 0 Then

rng2.Start = rng.End

rng2.Select

Else

Set rng2 = rng.Duplicate

End If

' Flash the range

For i = 1 To 3

DoEvents

myTime = Timer

Do

Loop Until Timer > myTime + 0.06

Selection.Collapse wdCollapseStart

DoEvents

myTime = Timer

Do

Loop Until Timer > myTime + 0.08

rng2.Select

Next i

rng.Collapse wdCollapseStart

rng.Select

If Selection.Start = initialPosition Then Beep

End Sub

**Sub FindStyleOld()**

**'** Paul Beverley - Version 17.11.10

' Find text in this style

' Alt-O

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

targetStyle = Selection.Range.Style

Do

Selection.MoveDown Unit:=wdParagraph, Count:=1

If Selection.Start = 0 Then Beep: Exit Sub

thisStyle = Selection.Range.Style

Loop Until thisStyle <> targetStyle

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Wrap = False

.Text = ""

.Replacement.Text = ""

.Style = targetStyle

.Execute

End With

Selection.Start = Selection.End

Selection.MoveLeft Unit:=wdCharacter, Count:=1

Selection.MoveUp Unit:=wdLine, Count:=3

Selection.MoveDown Unit:=wdLine, Count:=3

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.Wrap = wdFindContinue

End With

End Sub

**Sub FindStyleOldUp()**

**'** Paul Beverley - Version 16.11.10

' Find text in this style

' Alt-I

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

targetStyle = Selection.Range.Style

Do

Selection.MoveUp Unit:=wdParagraph, Count:=1

thisStyle = Selection.Range.Style

Loop Until thisStyle <> targetStyle

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = ""

.Forward = False

.Wrap = False

.Replacement.Text = ""

.Style = targetStyle

.Execute

End With

Selection.Start = Selection.End

Selection.MoveLeft Unit:=wdCharacter, Count:=1

Selection.MoveUp Unit:=wdLine, Count:=3

Selection.MoveDown Unit:=wdLine, Count:=3

With Selection.Find

.Text = oldFind

.Wrap = wdFindContinue

.Replacement.Text = oldReplace

.Forward = True

End With

End Sub

**Sub FindStyle()**

' Paul Beverley - Version 30.12.10

' Finds text in this style

' Turn the selected text into Normal style

Selection.Style = wdStyleNormal

' and records its parameters

nmlSize = Selection.Font.Size

nmlFont = Selection.Font.Name

nmlColour = Selection.Font.Color

' undo the change, i.e. restore the text's original style

WordBasic.EditUndo

' Check the text's emphasis

isBold = Selection.Font.Bold

isItalic = Selection.Font.Italic

isSuper = Selection.Font.Superscript

isSub = Selection.Font.Subscript

isSmalls = Selection.Font.SmallCaps

thisSize = Selection.Font.Size

thisFont = Selection.Font.Name

thisColour = Selection.Font.Color

' First find all the existing style

endWas = Selection.End

Selection.Start = Selection.End

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Replacement.Text = ""

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

If isBold = True Then .Font.Bold = True

If isItalic = True Then .Font.Italic = True

If isSuper = True Then .Font.Superscript = True

If isSub = True Then .Font.Subscript = True

If isSmalls = True Then .Font.SmallCaps = True

If thisSize <> nmlSize Then .Font.Size = thisSize

If thisFont <> nmlFont Then .Font.Name = thisFont

If thisColour <> nmlColour Then .Font.Color = thisColour

.Wrap = wdFindStop

.Forward = True

.Execute

End With

If Selection.Start = endWas Then

Selection.Start = Selection.End

Else

Selection.End = Selection.Start

End If

' Now go and find the next one

hereNow = Selection.End

With Selection.Find

' .Forward = False

.Execute

End With

If Selection.Start = hereNow And Selection.End = hereNow Then

If Selection.Find.Found = False Then Beep

Else

' Move the screen display down a couple of lines

hereNow = Selection.Start

Selection.MoveUp Unit:=wdLine, Count:=2

Selection.MoveDown Unit:=wdLine, Count:=2

With Selection.Find

.Forward = True

.Execute

End With

If Selection.Start > hereNow Then

With Selection.Find

.Forward = False

.Execute

End With

End If

End If

With Selection.Find

.Forward = True

.Wrap = wdFindContinue

End With

End Sub

**Sub** **JumpNextAppliedStyle()**

' Paul Beverley - Version 11.08.20

' Jumps to the next applied style

Set rng = Selection.Range.Duplicate

Do

rng.Start = rng.Start + 1

rng.Expand wdParagraph

myStyle = rng.Style

rng.Collapse wdCollapseEnd

DoEvents

Loop Until (myStyle <> "Normal") Or rng.End > ActiveDocument.Content.End - 2

rng.Select

Selection.Collapse wdCollapseStart

Beep

End Sub

**Sub ToCback()**

**'** Paul Beverley - Version 18.07.10

' Jump back to table of contents

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

Selection.HomeKey Unit:=wdLine

Selection.EndKey Unit:=wdLine, Extend:=wdExtend

thisLine = Selection

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = thisLine

.Replacement.Text = ""

.Execute

End With

Selection.MoveUp Unit:=wdLine, Count:=3

Selection.MoveDown Unit:=wdLine, Count:=3

Selection.MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdExtend

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

End With

End Sub

**Sub CommentNext()**

' Paul Beverley - Version 13.11.21

' Goes to next comment

hereNow = Selection.Start

Selection.GoTo What:=wdGoToComment, Which:=wdGoToNext

If Selection.Start = hereNow Then Beep

End Sub

**Sub CommentPrevious()**

' Paul Beverley - Version 14.11.21

' Goes to previous comment

hereNow = Selection.Start

Selection.GoTo What:=wdGoToComment, Which:=wdGoToPrevious

If Selection.Start = hereNow Then Beep

End Sub

**Sub CommentCompose()**

**'** Paul Beverley - Version 26.02.24

' Allows you to create a new (modern) comment

' sentenceSelect = False

sentenceSelect = True

stdName = "Document"

docName = ActiveDocument.Name

dotPos = InStr(docName, ".")

If dotPos > 1 Then

justName = Left(docName, dotPos - 1)

Else

justName = docName

End If

If Left(docName, Len(stdName)) = stdName Then GoTo insertComment

gottaCompo = False

For Each myDoc In Documents

thisName = myDoc.Name

If Left(thisName, Len(stdName)) = stdName And \_

InStr(myDoc.Paragraphs(1).Range.Text, justName) > 0 Then

myDoc.Activate

gottaCompo = True

Exit For

End If

DoEvents

Next myDoc

If gottaCompo = False Then

Documents.Add

Selection.TypeText Text:=docName & vbCr & vbCr

Else

If ActiveDocument.Paragraphs.Count > 2 Then

ActiveDocument.Paragraphs(3).Range.Select

Selection.End = ActiveDocument.Content.End

Else

Selection.EndKey Unit:=wdStory

End If

End If

Exit Sub

insertComment:

Set rng = ActiveDocument.Paragraphs(1).Range

rng.End = rng.End - 1

docName = rng.Text

If ActiveDocument.Paragraphs.Count > 2 Then

ActiveDocument.Paragraphs(3).Range.Select

Selection.End = ActiveDocument.Content.End

Else

myResponse = MsgBox("Please type your comment in here", vbQuestion \_

+ vbOKOnly, "CommentCompose")

Exit Sub

End If

Selection.Copy

For Each myDoc In Documents

thisName = myDoc.Name

If thisName = docName Then

myDoc.Activate

Exit For

End If

DoEvents

Next myDoc

Set myWnd = ActiveDocument.ActiveWindow

If myWnd.WindowState = 2 Then myWnd.WindowState = wdWindowStateNormal

' If no text selected, select the current sentence

If Selection.Start = Selection.End And sentenceSelect = True Then

Selection.Expand wdSentence

If Right(Selection, 4) = "al. " Or Right(Selection, 5) = "al., " \_

Or Right(Selection, 5) = "e.g. " Or Right(Selection, 5) = "i.e. " \_

Or Right(Selection, 6) = "e.g., " Or Right(Selection, 6) = "i.e., " Then

Selection.MoveRight Unit:=wdSentence, Count:=1, Extend:=wdExtend

End If

Do While InStr(" " & vbCr, Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

Dim cmt As Comment

Set cmt = Selection.Comments.Add(Range:=Selection.Range)

Selection.Paste

ActiveWindow.ActivePane.Close

End Sub

**Sub CommentComposeMenu()**

**'** Paul Beverley - Version 26.02.24

' Adds a comment off a menu

stdName = "Document"

' myPrefix = ""

myPrefix = "PB: "

prefixBold = True

' refText = ""

' refText = "p <p> "

refText = "p <p>, ln <l>. "

refBold = True

listName = "zzSwitchList"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff\"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff/"

defaultList = myFolder & listName

Set startDoc = ActiveDocument

docName = ActiveDocument.Name

dotPos = InStr(docName, ".")

If dotPos > 1 Then

justName = Left(docName, dotPos - 1)

Else

justName = docName

End If

If Left(docName, Len(stdName)) = stdName Then GoTo insertComment

' Register the page and line number

Set rng = Selection.Range.Duplicate

rng.End = rng.Start + 1

pNum = rng.Information(wdActiveEndAdjustedPageNumber)

lNum = rng.Information(wdFirstCharacterLineNumber)

' Begin to compose menu-based comment

On Error GoTo ReportIt

Set myDoc = Application.Documents.Open(fileName:=defaultList & ".docx")

On Error GoTo 0

Set menuDoc = ActiveDocument

' Find first comment line

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "]]^p"

.Wrap = wdFindContinue

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

CR = vbCr

Selection.Collapse wdCollapseEnd

Selection.End = ActiveDocument.Content.End

myPrompts = Selection

sqPos = InStr(myPrompts, "[")

myFullPrompt = ""

Do While sqPos > 0

myPrompts = Mid(myPrompts, sqPos + 1)

endPos = InStr(myPrompts, "]")

myFullPrompt = myFullPrompt & Left(myPrompts, endPos - 1) & CR

sqPos = InStr(myPrompts, "[")

DoEvents

Loop

' Choose the comment

codePos = 0

Do While codePos = 0

startDoc.Activate

myText = InputBox(myFullPrompt, "CommentComposeMenu")

menuDoc.Activate

If myText = "" Then

Beep

startDoc.Activate

Exit Sub

End If

myCode = UCase(myText)

codePos = InStr(Selection, "[" & myCode & " ")

Loop

Selection.End = Selection.Start + codePos

Selection.Collapse wdCollapseEnd

Selection.Expand wdParagraph

sqPos = InStr(Selection, "[")

Selection.End = Selection.Start + sqPos - 2

Selection.Copy

gottaCompo = False

For Each myDoc In Documents

thisName = myDoc.Name

If Left(thisName, Len(stdName)) = stdName And \_

InStr(myDoc.Paragraphs(1).Range.Text, justName) > 0 Then

myDoc.Activate

gottaCompo = True

Exit For

End If

DoEvents

Next myDoc

Set myWnd = ActiveDocument.ActiveWindow

If myWnd.WindowState = 2 Then myWnd.WindowState = wdWindowStateNormal

If gottaCompo = False Then

startDoc.Activate

Documents.Add

Selection.TypeText Text:=docName & vbCr & vbCr

Else

If ActiveDocument.Paragraphs.Count > 2 Then

ActiveDocument.Paragraphs(3).Range.Select

Selection.End = ActiveDocument.Content.End

Selection.Delete

Selection.TypeText Text:=vbCr

Else

Selection.EndKey Unit:=wdStory

End If

End If

DoEvents

Selection.Paste

' Go back and get text range to quote

Set compoDoc = ActiveDocument

startDoc.Activate

If Selection.Start <> Selection.End Then Selection.Copy

compoDoc.Activate

' Replace the <> with the quote

quotePos = InStr(ActiveDocument.Content, "<>")

If quotePos > 0 Then

Selection.Start = quotePos - 1

Selection.End = Selection.Start + 2

Selection.Delete

DoEvents

Selection.Paste

End If

quotePos = InStr(ActiveDocument.Content, "{}")

If quotePos > 0 Then

Selection.Start = quotePos - 1

Selection.End = Selection.Start + 2

DoEvents

Selection.Delete

DoEvents

Selection.PasteSpecial DataType:=wdPasteText

End If

' And again, in case there are two!

quotePos = InStr(ActiveDocument.Content, "<>")

If quotePos > 0 Then

Selection.Start = quotePos - 1

Selection.End = Selection.Start + 2

Selection.Delete

DoEvents

Selection.Paste

End If

quotePos = InStr(ActiveDocument.Content, "{}")

If quotePos > 0 Then

Selection.Start = quotePos - 1

Selection.End = Selection.Start + 2

Selection.Delete

DoEvents

Selection.PasteSpecial DataType:=wdPasteText

End If

If refText > "" Then

refText = Replace(refText, "<p>", Trim(Str$(pNum)))

refText = Replace(refText, "<l>", Trim(Str$(lNum)))

ActiveDocument.Paragraphs(3).Range.Select

Selection.Collapse wdCollapseStart

myStart = Selection.Start

Selection.InsertBefore Text:=refText

Selection.Start = myStart

If refBold = True Then Selection.Font.Bold = True

Selection.EndKey Unit:=wdStory

End If

If myPrefix > "" Then

ActiveDocument.Paragraphs(3).Range.Select

Selection.Collapse wdCollapseStart

myStart = Selection.Start

Selection.InsertBefore Text:=myPrefix

Selection.Start = myStart

If prefixBold = True Then Selection.Font.Bold = True

Selection.EndKey Unit:=wdStory

End If

cursorPos = InStr(ActiveDocument.Content, "|")

If cursorPos > 0 Then

Selection.End = cursorPos

Selection.Start = Selection.End - 1

Selection.Delete

End If

Selection.Collapse wdCollapseEnd

Exit Sub

insertComment:

Set rng = ActiveDocument.Paragraphs(1).Range

rng.End = rng.End - 1

docName = rng.Text

If ActiveDocument.Paragraphs.Count > 2 Then

ActiveDocument.Paragraphs(3).Range.Select

Selection.End = ActiveDocument.Content.End

Else

myResponse = MsgBox("Please type your comment in here", vbQuestion \_

+ vbOKOnly, "CommentCompose")

Exit Sub

End If

Selection.Copy

For Each myDoc In Documents

thisName = myDoc.Name

If thisName = docName Then

myDoc.Activate

Exit For

End If

DoEvents

Next myDoc

Set myWnd = ActiveDocument.ActiveWindow

If myWnd.WindowState = 2 Then myWnd.WindowState = wdWindowStateNormal

' If no text selected, select the current sentence

If Selection.Start = Selection.End And sentenceSelect = True Then

Selection.Expand wdSentence

If Right(Selection, 4) = "al. " Or Right(Selection, 5) = "al., " \_

Or Right(Selection, 5) = "e.g. " Or Right(Selection, 5) = "i.e. " \_

Or Right(Selection, 6) = "e.g., " Or Right(Selection, 6) = "i.e., " Then

Selection.MoveRight Unit:=wdSentence, Count:=1, Extend:=wdExtend

End If

Do While InStr(" " & vbCr, Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

Dim cmt As Comment

Set cmt = Selection.Comments.Add(Range:=Selection.Range)

DoEvents

Selection.Paste

ActiveWindow.ActivePane.Close

Exit Sub

ReportIt:

If Err.Number = 5174 Then

MsgBox ("Couldn't find file: " & listName)

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub CommentsModernCollect()**

' Paul Beverley - Version 02.08.21

' Extracts modern comments with format and colouring

myInits = "PB"

myColour = wdBlue

myInits2 = "NH"

myColour2 = wdPink

' For any other initials

myColour3 = wdGreen

Set mainText = ActiveDocument

Documents.Add

CR = vbCr

CR2 = CR & CR

totCmnts = mainText.Comments.Count

If totCmnts = 0 Then

Beep

Exit Sub

End If

For i = 1 To totCmnts

Set cmnt = mainText.Comments(i)

cmntInits = cmnt.Initial

cmnt.Range.Copy

myStart = Selection.Start

Selection.InsertAfter Text:=cmntInits & ": "

Selection.Start = myStart

gotColour = False

Selection.Font.Bold = True

If cmntInits = myInits Then

Selection.Font.ColorIndex = myColour

gotColour = True

End If

If gotColour = False Then

If cmntInits = myInits2 Then

Selection.Font.ColorIndex = myColour2

Else

Selection.Font.ColorIndex = myColour3

End If

End If

Selection.Collapse wdCollapseEnd

Selection.Paste

Selection.TypeText Text:=CR2

Selection.Collapse wdCollapseEnd

DoEvents

Next i

Beep

Selection.TypeText Text:=CR2

Selection.TypeText Text:="Total comments = " & Str(totCmnts)

End Sub

**Sub CommentsModernDelete()**

' Paul Beverley - Version 02.08.21

' Deletes current comment or all comments

If Selection.Start = Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Start = 0

n = rng.Comments.Count + 1

ActiveDocument.Comments(n).Scope.Select

Selection.Collapse wdCollapseEnd

ActiveDocument.Comments(n).DeleteRecursively

Else

numberCmnts = ActiveDocument.Comments.Count

myResponse = MsgBox("Delete all comments?!", vbQuestion \_

+ vbYesNoCancel, "CommentsModernDelete")

If myResponse <> vbYes Then Beep: Exit Sub

If numberCmnts > 0 Then ActiveDocument.DeleteAllComments

MsgBox ("Comments deleted: " & Str(numberCmnts))

Beep

End If

End Sub

**Sub CommentJumpInOut()**

' Paul Beverley - Version 13.06.22

' Jumps in and out of comment text

inCommentBox = Selection.Range.Information(wdInCommentPane)

If inCommentBox Then

Selection.Comments(1).Scope.Select

Selection.Collapse wdCollapseStart

Else

Set rng = ActiveDocument.Range(0, Selection.End)

cmtNum = rng.Comments.Count + 1

If cmtNum > ActiveDocument.Comments.Count Then

cmtNum = ActiveDocument.Comments.Count

End If

ActiveDocument.Comments(cmtNum).Edit

End If

End Sub

**Sub ChangeNext()**

' Paul Beverley - Version 29.08.20

' Finds next change (but not comment)

' Set rng = Selection.Range.Duplicate

On Error GoTo theEnd

foundList = " "

Do

Application.Run macroName:="NextChangeOrComment"

hereNow = Str(Selection.Start)

Debug.Print hereNow

If Selection.Start > 0 Then

If InStr(foundList, " " & hereNow & " ") > 0 Then

Beep

MsgBox "No tracked changes found"

Exit Sub

End If

foundList = foundList & hereNow & " "

End If

If Selection.Start < rng.Start And Not \_

(Selection.Information(wdInCommentPane)) Then

Selection.EndKey Unit:=wdStory

rng.Select

Selection.Collapse wdCollapseEnd

Beep

End If

DoEvents

Debug.Print foundList

Loop While Selection.Information(wdInCommentPane)

Exit Sub

theEnd:

On Error GoTo 0

Beep

End Sub

**Sub ChangePrevious()**

' Paul Beverley - Version 23.10.13

' Finds next change (not! comment)

On Error GoTo theEnd

Do

Application.Run MacroName:="PreviousChangeOrComment"

DoEvents

Loop While Selection.Information(wdInCommentPane)

Exit Sub

theEnd:

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub CompareNow()**

' Paul Beverley - Version 26.10.21

' Creates an instant comparison of two open Word files

myFile1 = "aa1.docx"

myFile2 = "aa2.docx"

ignoreFilesWith = "zzSw"

' showDetail = wdGranularityCharLevel

showDetail = wdGranularityWordLevel

' checkFormatting = False

checkFormatting = True

checkSpaces = False

' checkSpaces = True

' checkCase = False

checkCase = True

checkTables = True

' checkTables = False

showMoves = False

' showMoves = True

CR = vbCr: CR2 = CR & CR

k = "!": allNames = k

Dim myDoc(10) As Document

numDoc = 0

gotFile1 = False

gotFile2 = False

For Each myWnd In Application.Windows

aName = myWnd.Document.Name

If InStr(allNames, k & aName & k) = 0 And \_

Left(aName, Len(ignoreFilesWith)) <> ignoreFilesWith Then

If aName = myFile1 Then

gotFile1 = True

Set doc1 = myWnd.Document

End If

If aName = myFile2 Then

gotFile2 = True

Set doc2 = myWnd.Document

End If

If aName <> myFile1 And aName <> myFile2 Then

numDoc = numDoc + 1

Set myDoc(numDoc) = myWnd.Document

allNames = allNames & aName & k

End If

End If

Next myWnd

For i = 1 To numDoc

myPrompt = myPrompt & Trim(Str(i)) & " – " \_

& myDoc(i).Name & CR

Next i

myPrompt = myPrompt & CR & "Compare which files?" & \_

CR2 & CR & "m - Moves t - Tables s - Spaces c - Case" \_

& CR & "f - Formatting w - Word (or character)"

Do

gotOne = True

myChoice = InputBox(myPrompt, "CompareNow", "12")

If Len(myChoice) = 0 Then Beep: Exit Sub

If Len(myChoice) = 1 Or Val(myChoice) < 12 Then

Beep

MsgBox "Please type two numbers to indicate your file choice."

gotOne = False

End If

befDoc = Val(Left(myChoice, 1))

aftDoc = Val(Mid(myChoice, 2, 1))

If befDoc > numDoc Or aftDoc > numDoc Then

gotOne = False

Beep

MsgBox "Please type two numbers within the range 1 to " & Str(numDoc)

End If

Loop Until gotOne = True

Set beforeDoc = myDoc(befDoc)

beforeDoc.Activate

If Selection.Start <> Selection.End Then

If gotFile1 = False Then

Beep

myResponse = MsgBox("Please open or create a file called " \_

& myFile1, vbQuestion + vbOKOnly, "CompareNow")

Exit Sub

End If

Set rng = Selection.Range.Duplicate

doc1.Activate

Set beforeDoc = doc1

Selection.WholeStory

Selection.Delete

Selection.Range.FormattedText = rng.FormattedText

End If

myDestination = wdCompareDestinationNew

Set afterDoc = myDoc(aftDoc)

afterDoc.Activate

If Selection.Start <> Selection.End Then

If gotFile2 = False Then

Beep

myResponse = MsgBox("Please open or create a file called " \_

& myFile2, vbQuestion + vbOKOnly, "CompareNow")

Exit Sub

End If

Set rng = Selection.Range.Duplicate

doc2.Activate

Set afterDoc = doc2

Selection.WholeStory

Selection.Delete

Selection.Range.FormattedText = rng.FormattedText

myDestination = wdCompareDestinationRevised

End If

myChoice = UCase(Mid(myChoice, 2))

If InStr(myChoice, "M") > 0 Then showMoves = Not (showMoves)

If InStr(myChoice, "T") > 0 Then checkTables = Not (checkTables)

If InStr(myChoice, "S") > 0 Then checkSpaces = Not (checkSpaces)

If InStr(myChoice, "C") > 0 Then checkCase = Not (checkCase)

If InStr(myChoice, "F") > 0 Then checkFormatting = \_

Not (checkFormatting)

If InStr(myChoice, "W") > 0 Then

If showDetail = wdGranularityCharLevel Then

showDetail = wdGranularityWordLevel

Else

showDetail = wdGranularityCharLevel

End If

End If

Application.CompareDocuments \_

OriginalDocument:=beforeDoc, \_

RevisedDocument:=afterDoc, \_

Destination:=myDestination, \_

Granularity:=showDetail, \_

CompareFormatting:=checkFormatting, \_

CompareCaseChanges:=checkCase, \_

CompareWhitespace:=checkSpaces, \_

Comparetables:=checkTables, \_

CompareMoves:=showMoves

Set resultsDoc = ActiveDocument

End Sub

**Sub FandRdespiteTCsGlobal()**

' Paul Beverley - Version 19.09.18

' Does current F&R despite track changes (global)

doGlobal = True

With Selection.Find

myBold = .Replacement.Font.Bold

myItalic = .Replacement.Font.Italic

myReplace = .Replacement.Text

myHighlight = .Replacement.Highlight

If doGlobal = False Then

.Wrap = wdFindStop

Else

.Wrap = wdFindContinue

End If

.Execute

End With

Do While Selection.Find.Found = True

myResponse = vbYes

If doGlobal = False Then

myResponse = MsgBox("Continue?", vbQuestion \_

+ vbYesNoCancel, "F&R despite TCs")

If myResponse = vbCancel Then Exit Sub

End If

With Selection

If myResponse = vbYes Then

startNow = .Start

.TypeText Text:=myReplace

.Start = startNow

If myBold = True Then .Font.Bold = True

If myItalic = True Then .Font.Italic = True

If myHighlight = True Then .range.HighlightColorIndex = \_

Options.DefaultHighlightColorIndex

End If

.Collapse wdCollapseEnd

.Find.Execute

End With

Loop

End Sub

**Sub FandRdespiteTCsSelective()**

' Paul Beverley - Version 19.09.18

' Does current F&R despite track changes (selective)

doGlobal = False

With Selection.Find

myBold = .Replacement.Font.Bold

myItalic = .Replacement.Font.Italic

myReplace = .Replacement.Text

myHighlight = .Replacement.Highlight

If doGlobal = False Then

.Wrap = wdFindStop

Else

.Wrap = wdFindContinue

End If

.Execute

End With

Do While Selection.Find.Found = True

myResponse = vbYes

If doGlobal = False Then

myResponse = MsgBox("Continue?", vbQuestion \_

+ vbYesNoCancel, "F&R despite TCs")

If myResponse = vbCancel Then Exit Sub

End If

With Selection

If myResponse = vbYes Then

startNow = .Start

.TypeText Text:=myReplace

.Start = startNow

If myBold = True Then .Font.Bold = True

If myItalic = True Then .Font.Italic = True

If myHighlight = True Then .range.HighlightColorIndex = \_

Options.DefaultHighlightColorIndex

End If

.Collapse wdCollapseEnd

.Find.Execute

End With

Loop

End Sub

**Sub TrackChangeCounter()**

**'** Paul Beverley - Version 18.01.24

' Reports the numbers of track changes, plus words/characters added/deleted

CR = vbCr

CR2 = CR & CR

numRevs = ActiveDocument.Revisions.Count

For Each rev In ActiveDocument.Revisions

Select Case rev.Type

Case wdRevisionInsert

myAddChars = myAddChars + Len(rev.Range.Text)

myAddWords = myAddWords + rev.Range.Words.Count

Case wdRevisionDelete

myDeletesChars = myDeletesChars + Len(rev.Range.Text)

myDeletesWords = myDeletesWords + rev.Range.Words.Count

End Select

i = i + 1

If i Mod 10 = 0 Then DoEvents

Next rev

myResult = Str(numRevs) & " Revisions" & CR2

myResult = myResult & "Deletions" & CR

myResult = myResult & " Words: " & myDeletesWords & CR

myResult = myResult & " Characters: " & myDeletesChars & CR2

myResult = myResult & "Insertions" & CR

myResult = myResult & " Words: " & myAddWords & CR

myResult = myResult & " Characters: " & myAddChars & CR2

MsgBox myResult

End Sub

**Sub CopyAllEditedSentences()**

' Paul Beverley - Version 20.04.21

' Copies all sentences that have tracked edits in them

ActiveDocument.TrackRevisions = False

Set rngSource = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rngSource.FormattedText

Selection.EndKey Unit:=wdStory

enCount = ActiveDocument.Endnotes.Count

If enCount > 0 Then

ActiveDocument.StoryRanges(wdEndnotesStory).Copy

Selection.Paste

For i = enCount To 1 Step -1

ActiveDocument.Endnotes(i).Delete

Next i

End If

fnCount = ActiveDocument.Footnotes.Count

If fnCount > 0 Then

ActiveDocument.StoryRanges(wdFootnotesStory).Copy

Selection.Paste

For i = fnCount To 1 Step -1

ActiveDocument.Footnotes(i).Delete

Next i

End If

snCount = ActiveDocument.Sentences.Count

For i = snCount To 1 Step -1

Set sn = ActiveDocument.Sentences(i)

snWas = sn.Text

sn.Revisions.RejectAll

If sn.Text = snWas Then

sn.Delete

Else

WordBasic.EditUndo

sn.InsertBefore Text:=vbCr & vbCr

End If

DoEvents

Next i

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub TrackDateTimeList()**

' Paul Beverley - Version 13.05.20

' Lists the date and time of all track changes

Set rng = ActiveDocument.Content

Documents.Add

tot = rng.Revisions.Count

For i = 1 To tot

Selection.TypeText Text:=rng.Revisions(i).Date & vbCr

DoEvents

Next i

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

End Sub

**Sub DictaFRedit()**

' Paul Beverley - Version 06.02.21

' Adds features to Word 365 Dictate, and cleans up it errors

doShowChanges = True

assumeWholeText = False

warnReWholeText = False

doCopy = False

myColour = wdColorBlue

myColour = 0

myHighlight = 0

myHighlight = wdGray25

doUnderline = False

doSelectResult = False

doBeep = True

inFilename = "box"

' inFilename = "Switch"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff"

If doShowChanges = False Then

myColour = 0

myHighlight = 0

doUnderline = False

End If

listName = "zzDictateBox.docx"

CR = vbCr: CR2 = CR & CR

' Decide what range of text to process

doingPara = False

If InStr(ActiveDocument.Name, "zClipStore") > 0 Then

Selection.EndKey Unit:=wdStory

Do

Selection.MoveStart wdParagraph, -1

DoEvents

Loop Until Left(Selection, 1) = "\_"

Selection.MoveStart wdParagraph, 1

End If

If Selection.Start = Selection.End Then

If assumeWholeText = True Then

Selection.WholeStory

If warnReWholeText = True Then

myResponse = MsgBox("Clean up WHOLE file?!", \_

vbYesNoCancel, "DictaFRedit")

If myResponse <> vbYes Then Exit Sub

End If

Else

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=CR

Selection.MoveLeft , 2

Selection.Expand wdParagraph

doingPara = True

End If

End If

Selection.Copy

' If first word of para is macro name, select it

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.MoveStart , -1

If Right(rng.Text, 1) = vbCr Then rng.End = rng.End - 1

rng.Expand wdParagraph

leadword = Trim(LCase(rng.Words(1)))

If InStr("|google|dictionary|translate|map|", \_

"|" & leadword & "|") > 0 Then

doingCall = True

rng.Select

End If

Selection.Copy

originalStart = Selection.Start

' Postcode wording error correction

myPostcodeDebug = False

a = ""

a = a & ";are|R;zero|0;one|1;"

a = a & ";two|2;three|3;for|4;"

a = a & ";four|4;five|5;six|6;oh|O;"

a = a & ";seven|7;eight|8;nine|9;ten|10;"

a = a & "es|S;ke|K;-|;to|2;be|B;a |A ;"

a = a & ";Zedd|Z;November|;|;|;"

a = a & ";|;|;|;|;"

' Correct postcodes

testLength = 26

If myPostcodeDebug = True Then

ActiveDocument.Content.Font.StrikeThrough = False

ActiveDocument.Content.Font.Underline = False

ActiveDocument.Content.HighlightColorIndex = wdNoHighlight

Set rng = Selection.Range.Duplicate

End If

Set wasRng = Selection.Range.Duplicate

Set testRng = Selection.Range.Duplicate

With testRng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "post"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchCase = False

.Execute

End With

myCount = 0

Do While testRng.Find.Found = True

DoEvents

absPos = testRng.Start

endBit = absPos + testLength

endDoc = ActiveDocument.Content.End

If endBit > endDoc Then endBit = endDoc

Set thisRng = ActiveDocument.Range(absPos, endBit)

crPos = InStr(thisRng.Text, vbCr)

If crPos > 0 Then thisRng.End = thisRng.Start + crPos - 1

If myPostcodeDebug = True Then thisRng.Font.Underline = True

dCode = InStr(thisRng.Text, "code")

If dCode > 0 Then

thisRng.Text = Replace(thisRng.Text, " ", " ")

Set rng = ActiveDocument.Range(absPos + dCode + 4, thisRng.End)

myText = rng.Text

myTextWas = myText

If myPostcodeDebug = True Then rng.Font.StrikeThrough = True

a = Replace(a, ";;", ";")

myFR = Split(Trim(a), ";")

For i = 0 To UBound(myFR) - 1

txt = myFR(i)

pPos = InStr(txt, "|")

If pPos > 1 Then

myF = Left(txt, pPos - 1)

myR = Mid(txt, pPos + 1)

myWas = myText

myText = Replace(myText, myF, myR)

If myWas <> myText Then Debug.Print myF, myR, myText

End If

DoEvents

Next i

myText = UCase(myText)

myPattern = ""

For i = 1 To Len(myText)

ch = Mid(myText, i, 1)

If LCase(ch) <> UCase(ch) Then

myPattern = myPattern & "."

Else

If ch = " " Then

myPattern = myPattern & " "

Else

myPattern = myPattern & "#"

End If

End If

DoEvents

Next i

Debug.Print myPattern

patnPos = InStr(myPattern, "#..")

pullBack = Len(myText) - patnPos - 2

If patnPos = 0 Then

patnPos = InStr(myPattern, "# ..")

pullBack = Len(myText) - patnPos - 3

End If

If patnPos = 0 Then

patnPos = InStr(myPattern, "#. .")

pullBack = Len(myText) - patnPos - 3

End If

If patnPos = 0 Then

patnPos = InStr(myPattern, "# . .")

pullBack = Len(myText) - patnPos - 4

End If

rng.Start = absPos

endPattern = Right(myPattern, pullBack)

If InStr(endPattern, "#") > 0 Then

pullBack = pullBack + 1

End If

rng.End = rng.End - pullBack

If myPostcodeDebug = True Then rng.HighlightColorIndex = wdBrightGreen

If Left(myPattern, 6) = "..##.." Then

part1 = Left(myText, 3)

part2 = Mid(myText, 4, 3)

Debug.Print part1, part2, myText

End If

If Left(myPattern, 7) = "..# #.." Then

part1 = Left(myText, 3)

part2 = Mid(myText, 5, 3)

Debug.Print part1, part2, myText

End If

If Left(myPattern, 7) = ". .##.." Then

part1 = Replace(Left(myText, 4), " ", "")

part2 = Mid(myText, 5, 3)

Debug.Print part1, part2, myText

End If

If Left(myPattern, 7) = "..##. ." Then

part1 = Left(myText, 3)

part2 = Replace(Mid(myText, 4, 4), " ", "")

Debug.Print part1, part2, myText

End If

If Left(myPattern, 8) = ". .# #.." Then

part1 = Replace(Left(myText, 4), " ", "")

part2 = Mid(myText, 6, 3)

Debug.Print part1, part2, myText

End If

If Left(myPattern, 9) = ". .# # .." Then

part1 = Replace(Left(myText, 4), " ", "")

part2 = Replace(Mid(myText, 6, 4), " ", "")

Debug.Print part1, part2, myText

End If

If Left(myPattern, 9) = ". ## # .." Then

part1 = Replace(Left(myText, 4), " ", "")

part2 = Replace(Mid(myText, 6, 4), " ", "")

Debug.Print part1, part2, myText

End If

If Left(myPattern, 10) = ".. ## # .." Then

part1 = Replace(Left(myText, 5), " ", "")

part2 = Replace(Mid(myText, 7, 4), " ", "")

Debug.Print part1, part2, myText

End If

If Left(myPattern, 10) = ". .# # . ." Then

part1 = Replace(Left(myText, 4), " ", "")

part2 = Replace(Mid(myText, 6, 5), " ", "")

Debug.Print part1, part2, myText

End If

If Left(myPattern, 10) = "..## # . ." Then

part1 = Replace(Left(myText, 4), " ", "")

part2 = Replace(Mid(myText, 6, 5), " ", "")

Debug.Print part1, part2, myText

End If

If Left(myPattern, 11) = ". . ## # .." Then

part1 = Replace(Left(myText, 6), " ", "")

part2 = Replace(Mid(myText, 8, 4), " ", "")

Debug.Print part1, part2, myText

End If

If Left(myPattern, 8) = "..#. #.." Then

part1 = Left(myText, 4)

part2 = Mid(myText, 6, 3)

Debug.Print part1, part2, myText

End If

If Left(myPattern, 8) = "..## #.." Then

part1 = Left(myText, 4)

part2 = Mid(myText, 6, 3)

Debug.Print part1, part2, myText

End If

myPCode = part1 & " " & part2

rng.Text = myPCode

rng.Collapse wdCollapseEnd

rng.MoveEnd , 2

If rng.Text = "to" Then rng.InsertBefore Text:=" "

End If

If myPostcodeDebug = True Then

rng.Select

wasRng.Select

End If

testRng.Start = rng.End

testRng.End = Selection.End

testRng.Find.Execute

Loop

' Find the conversion file

Set workDoc = ActiveDocument

thisName = LCase(workDoc.Name)

gottaList = False

If InStr(thisName, LCase(inFilename)) > 0 Then

gottaList = True

useThisList = True

Else

useThisList = False

For Each myDoc In Documents

thisName = LCase(myDoc.Name)

myDoc.Activate

If InStr(thisName, LCase(inFilename)) > 0 Then

gottaList = True

Exit For

End If

DoEvents

Next myDoc

If gottaList = False Then

' Load the DictateList/Box file

On Error Resume Next

useThisList = False

Documents.Open myFolder & "/" & listName

If Err.Number = 5174 Then

Err.Clear

Beep

myResponse = MsgBox("Can't find a zzSwitchList file!", \_

vbOKOnly, "DictateExtra")

Exit Sub

Else

On Error GoTo 0

gottaList = True

End If

End If

End If

a = "": x = "": b = "|"

' Create the initial F&R list

i = 0: j = 0

Dim bolda(500) As Boolean

Dim boldx(500) As Boolean

Dim itala(500) As Boolean

Dim italx(500) As Boolean

For Each myPar In ActiveDocument.Paragraphs

myLine = myPar.Range.Text

If Left(myLine, 1) = "#" Then Exit For

If InStr(myLine, "/") > 0 And \_

Left(myLine, 1) <> "/" Then

Set myTest = myPar.Range.Characters(2)

isBold = (myTest.Font.Bold = True)

isItalic = (myTest.Font.Italic = True)

myLine = Replace(myLine, vbCr, "")

myLine = Replace(myLine, "^32", " ")

barPos = InStr(myLine, "/")

myF = Left(myLine, barPos)

myR = Mid(myLine, barPos + 1)

If Left(myR, 1) = "/" Then

myR = Mid(myR, 2)

x = x & myF & myR & b

j = j + 1

boldx(j) = isBold

italx(j) = isItalic

Else

a = a & myF & myR & b

i = i + 1

bolda(i) = isBold

itala(i) = isItalic

End If

End If

DoEvents

Next myPar

If useThisList = False Then workDoc.Activate

' Execute the initial F&R list

oldHighlight = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myHighlight

Set rng = Selection.Range.Duplicate

myFR = Split(Trim(a), "|")

For k = 0 To UBound(myFR) - 1

txt = myFR(k)

pPos = InStr(txt, "/")

myF = Left(txt, pPos - 1)

If Left(myF, 1) = "~" Then

myF = Mid(myF, 2)

goWild = True

Else

goWild = False

If Left(myF, 1) = ChrW(172) Then

myF = Mid(myF, 2)

fixCase = False

Else

fixCase = True

End If

End If

myR = Mid(txt, pPos + 1)

If pPos > 1 Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = goWild

.MatchCase = fixCase

.Wrap = wdFindStop

.Forward = True

If myR <> " " And myR <> "" Then

.Replacement.Highlight = True

.Replacement.Font.Color = myColour

If bolda(k) = True Then

.Replacement.Font.Bold = True

End If

If itala(k) = True Then

.Replacement.Font.Italic = True

End If

If doUnderline = True Then

.Replacement.Font.Underline = True

End If

End If

.Replacement.Font.Color = myColour

.Text = myF

.Replacement.Text = myR

.Execute Replace:=wdReplaceAll

End With

End If

DoEvents

Next k

' Do a load of funnies by global F&R

Set rng = Selection.Range.Duplicate

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Forward = True

.MatchWildcards = True

.Text = "[Ii]talic [Oo]n (\*) [Ii]talic [Oo]ff(?)"

.Replacement.Text = "\1zczc\2czcz"

.Replacement.Font.Italic = True

If myColour > 0 Then .Replacement.Font.Color = myColour

If myHighlight > 0 Then .Replacement.Highlight = True

If doUnderline = True Then .Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

.Replacement.ClearFormatting

.Text = "[Bb]old on (\*) [Bb]old off(?)"

.Replacement.Text = "\1zczc\2czcz"

.Replacement.Font.Bold = True

If myColour > 0 Then .Replacement.Font.Color = myColour

If myHighlight > 0 Then .Replacement.Highlight = True

If doUnderline = True Then .Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

.Replacement.ClearFormatting

.Text = "[Ss]ma[l ]{2,3}[Cc]aps o"

.Replacement.Text = "smallcaps o"

.Execute Replace:=wdReplaceAll

.Text = "smallcaps on (\*) smallcaps off(?)"

.Replacement.Text = "\1zczc\2czcz"

.Replacement.Font.SmallCaps = True

If myColour > 0 Then .Replacement.Font.Color = myColour

If myHighlight > 0 Then .Replacement.Highlight = True

If doUnderline = True Then .Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

.Replacement.ClearFormatting

.Text = "[Aa][l ]{2,3}[Cc]aps o"

.Replacement.Text = "allcaps o"

.Execute Replace:=wdReplaceAll

.Text = "allcaps on (\*) allcaps off(?)"

.Replacement.Text = "\1zczc\2czcz"

.Replacement.Font.AllCaps = True

If myColour > 0 Then .Replacement.Font.Color = myColour

If myHighlight > 0 Then .Replacement.Highlight = True

If doUnderline = True Then .Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

.Replacement.ClearFormatting

.Text = "superscript on (\*) superscript off(?)"

.Replacement.Text = "\1zczc\2czcz"

.Replacement.Font.Superscript = True

If myColour > 0 Then .Replacement.Font.Color = myColour

If myHighlight > 0 Then .Replacement.Highlight = True

If doUnderline = True Then .Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

.Replacement.ClearFormatting

.Text = "subscript on (\*) subscript off(?)"

.Replacement.Text = "\1zczc\2czcz"

.Replacement.Font.Subscript = True

If myColour > 0 Then .Replacement.Font.Color = myColour

If myHighlight > 0 Then .Replacement.Highlight = True

If doUnderline = True Then .Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

' ??????????????????????????????????????????????????????????????????????????

.Replacement.ClearFormatting

.Text = " to the minus (\*)([!0-9a-zA-Z])"

.Replacement.Text = ChrW(8722) & "\1\2"

.Replacement.Font.Superscript = True

If myColour > 0 Then .Replacement.Font.Color = myColour

If myHighlight > 0 Then .Replacement.Highlight = True

If doUnderline = True Then .Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

.Replacement.ClearFormatting

.Text = " to the plus (\*)([!0-9])"

.Replacement.Text = "+\1\2"

.Replacement.Font.Superscript = True

If myColour > 0 Then .Replacement.Font.Color = myColour

If myHighlight > 0 Then .Replacement.Highlight = True

If doUnderline = True Then .Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

.Replacement.ClearFormatting

.Text = " to the power (\*)([!0-9])"

.Replacement.Text = "\1\2"

.Replacement.Font.Superscript = True

If myColour > 0 Then .Replacement.Font.Color = myColour

If myHighlight > 0 Then .Replacement.Highlight = True

If doUnderline = True Then .Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

' Take attribute off following character

.Text = "zczc(?)czcz"

.Replacement.Text = "\1"

.Replacement.Font.SmallCaps = False

.Replacement.Font.AllCaps = False

' .Replacement.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Font.Superscript = False

.Replacement.Font.Subscript = False

.Execute Replace:=wdReplaceAll

End With

myFRx = Split(Trim(x), "|")

Set rng = Selection.Range.Duplicate

For k = 0 To UBound(myFRx) - 1

txt = myFRx(k)

pPos = InStr(txt, "/")

myF = Left(txt, pPos - 1)

If Left(myF, 1) = "~" Then

myF = Mid(myF, 2)

goWild = True

Else

goWild = False

If Left(myF, 1) = ChrW(172) Then

myF = Mid(myF, 2)

fixCase = False

Else

fixCase = True

End If

End If

myR = Mid(txt, pPos + 1)

If pPos > 1 Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = goWild

.MatchCase = fixCase

.Wrap = wdFindStop

.Forward = True

If myR <> " " And myR <> "" Then

.Replacement.Highlight = True

.Replacement.Font.Color = myColour

If boldx(k) = True Then

.Replacement.Font.Bold = True

End If

If italx(k) = True Then

.Replacement.Font.Italic = True

End If

If doUnderline = True Then

.Replacement.Font.Underline = True

End If

.Replacement.Font.Color = myColour

End If

.Text = myF

.Replacement.Text = myR

.Execute Replace:=wdReplaceAll

End With

End If

DoEvents

Next k

Set rng = Selection.Range.Duplicate

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "lowercase "

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchCase = False

.Execute

End With

Do While rng.Find.Found = True

rng.Delete

rng.End = rng.Start + 1

rng.Text = LCase(rng.Text)

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Set rng = Selection.Range.Duplicate

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "uppercase "

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchCase = False

.Execute

End With

Do While rng.Find.Found = True

rng.Delete

rng.End = rng.Start + 1

rng.Text = UCase(rng.Text)

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Set rng = Selection.Range.Duplicate

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " ([,.;:\!\?])"

.Wrap = False

.Replacement.Text = "\1"

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^^p"

.Wrap = False

.Replacement.Text = "^p"

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Options.DefaultHighlightColorIndex = oldHighlight

If doCopy = True Then Selection.Copy

If doingCall = True Then

Selection.MoveStart wdWord, 1

Select Case leadword

Case "google"

Call GoogleFetch

Exit Sub

Case "dictionary"

Call OUPFetch

Exit Sub

Case "translate"

Call GoogleTranslate

Exit Sub

Case "map"

Call GoogleMapFetch

Exit Sub

End Select

Selection.MoveStart wdWord, -1

Selection.Collapse wdCollapseStart

Selection.Expand wdParagraph

Set rng = Selection.Range.Duplicate

rng.End = rng.End - 1

rng.Start = rng.End - 1

lastChar = rng.Text

Selection.Collapse wdCollapseEnd

If lastChar = " " Then

Selection.MoveEnd , -1

Else

Selection.TypeText Text:=" " & vbCr

Selection.MoveLeft , 2

Selection.MoveStart , -1

Selection.Delete

Selection.MoveRight , 1

End If

End If

If doSelectResult = True Then

Selection.Start = originalStart

Else

Selection.Collapse wdCollapseEnd

If paraSelect = True Then Selection.MoveEnd , -11

End If

If doBeep = True Then Beep

End Sub

**Sub** **CountDownVisible()**

' Paul Beverley - Version 03.04.20

' Shows a statusbar and Big Text countdown to zero

myDownTime = 5

doBeep = True

beepAt = 2

doBigText = True

myZoomSize = 400

myFontSize = 144

ActiveWindow.ActivePane.View.Zoom.Percentage = myZoomSize

If doBigText = True Then

Selection.HomeKey Unit:=wdStory

If ActiveDocument.Paragraphs(1).Range.Font.Size \_

<> myFontSize Then

Selection.TypeText Text:=Trim(Str(myDownTime)) & vbCr

Set rng = ActiveDocument.Paragraphs(1).Range

rng.Font.Size = myFontSize

rng.Font.Bold = True

Else

ActiveDocument.Paragraphs(1).Range.Text = \_

Trim(Str(myDownTime)) & vbCr

End If

End If

myTime = Timer

For i = myDownTime To 1 Step -1

StatusBar = myPrompt

If doBigText = True Then

ActiveDocument.Paragraphs(1).Range.Text = Trim(Str(i)) & vbCr

End If

Do

DoEvents

Loop Until Timer > myTime + myDownTime - i + 1

myPrompt = ""

For j = 1 To 25

myPrompt = myPrompt & Str(i) & " "

Next j

Debug.Print i

If doBeep = True And i < beepAt + 1 Then

Beep

thisTime = Timer

Do

Loop Until Timer > thisTime + 0.2

Beep

End If

Next i

StatusBar = ""

If doBigText = True Then ActiveDocument.Paragraphs(1).Range.Delete

End Sub

**Sub** **LoadTheseFiles()**

' Paul Beverley - Version 13.05.20

' Loads all the files listed in a file list

myFileList = "myFileList.docx"

Documents.Open FileName:=myFileList

Set myList = ActiveDocument

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myLine = Replace(myPara.Range.Text, vbCr, "")

If Len(myLine) > 2 Then

If InStr(myLine, "\") > 0 Then myFolder = myLine

If Len(myFolder) > 1 And InStr(myLine, ".doc") > 0 Then

Documents.Open FileName:=myFolder & "\" & myLine

End If

End If

Next myPara

myList.Close SaveChanges:=False

End Sub

**Sub MaggyIt()**

' Paul Beverley - Version 28.10.17

' Creates a Maggied Paul Beverley - Version of the current file

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr & vbCr

Set rng = ActiveDocument.Content

rng.End = rng.End - 2

Documents.Add

Set rngNew = ActiveDocument.Content

rngNew.FormattedText = rng.FormattedText

End Sub

**Sub FootnoteNext()**

**'** Paul Beverley - Version 13.10.10

' Jump to next footnote

Selection.GoTo What:=wdGoToFootnote, Which:=wdGoToNext, \_

Count:=1, Name:=""

End Sub

**Sub FootnoteNextUp()**

**'** Paul Beverley - Version 13.10.10

' Jump to previous footnote

Selection.GoTo What:=wdGoToFootnote, Which:=wdGoToPrevious, \_

Count:=1, Name:=""

End Sub

**Sub NoteJumper()**

' Paul Beverley - Version 21.05.12

' Jump back and forth between notes and main text

myStoryType = Selection.StoryType

If (myStoryType = wdFootnotesStory Or myStoryType = wdEndnotesStory) Then

ActiveWindow.View.SeekView = wdSeekMainDocument

Else

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^e"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

If ActiveDocument.Footnotes.Count >= 1 Then \_

ActiveDocument.ActiveWindow.View.SeekView = wdSeekFootnotes

If ActiveDocument.Endnotes.Count >= 1 Then \_

ActiveDocument.ActiveWindow.View.SeekView = wdSeekEndnotes

End If

End Sub

**Sub NextNumberPlus()**

**'** Paul Beverley - Version 11.10.10

' Find next section number

allowedChars = "0123456789."

theNumber = ""

Selection.End = Selection.Start

startPos = Selection.Start

Selection.Start = startPos - 4

leftBit = Selection

Selection.Start = startPos - 1

If Asc(Selection) < 32 Then leftBit = ""

Selection.Start = startPos

pos = 1

dotPos = 0

Do

thisChar = Selection

theNumber = theNumber + thisChar

If thisChar = "." Then

prevDotPos = dotPos

dotPos = pos

End If

Selection.MoveRight Unit:=wdCharacter, Count:=1

pos = pos + 1

Loop Until InStr(allowedChars, thisChar) = 0

If dotPos > 0 Then

lastNumber = Mid(theNumber, dotPos + 1, pos - dotPos - 2)

If Val(lastNumber) = 0 Then

dotPos = prevDotPos

lastNumber = Mid(theNumber, dotPos + 1, pos - dotPos - 2)

End If

newNumber = Left(theNumber, dotPos) + Trim(Str(Val(lastNumber) + 1))

Else

lastNumber = Left(theNumber, pos - 2)

newNumber = Trim(Str(Val(lastNumber) + 1))

End If

hereNow = Selection.End

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

If leftBit > "" Then

.Text = leftBit & newNumber

Else

.Text = "^p" & newNumber

End If

.Replacement.Text = ""

.Wrap = False

.Forward = True

.MatchWildcards = False

.Execute

End With

Selection.End = Selection.Start

If Selection.End = hereNow Then beep

If leftBit > "" Then

Selection.MoveRight Unit:=wdCharacter, Count:=4

Else

Selection.MoveRight Unit:=wdCharacter, Count:=1

End If

'Add this to leave F&R dialogue in a sensible state

With Selection.Find

.Wrap = wdFindContinue

End With

Selection.End = Selection.Start

End Sub

**Sub NextNumberPlusUp()**

**'** Paul Beverley - Version 11.10.10

' Find previous section number

allowedChars = "0123456789."

theNumber = ""

Selection.End = Selection.Start

startPos = Selection.Start

Selection.Start = startPos - 4

leftBit = Selection

Selection.Start = startPos - 1

If Asc(Selection) < 32 Then leftBit = ""

Selection.Start = startPos

pos = 1

dotPos = 0

Do

thisChar = Selection

theNumber = theNumber + thisChar

If thisChar = "." Then

prevDotPos = dotPos

dotPos = pos

End If

Selection.MoveRight Unit:=wdCharacter, Count:=1

pos = pos + 1

Loop Until InStr(allowedChars, thisChar) = 0

If dotPos > 0 Then

lastNumber = Mid(theNumber, dotPos + 1, pos - dotPos - 2)

If Val(lastNumber) = 0 Then

dotPos = prevDotPos

lastNumber = Mid(theNumber, dotPos + 1, pos - dotPos - 2)

End If

newNumber = Left(theNumber, dotPos) + Trim(Str(Val(lastNumber) - 1))

Else

lastNumber = Left(theNumber, pos - 2)

newNumber = Trim(Str(Val(lastNumber) - 1))

End If

hereNow = Selection.End

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

If leftBit > "" Then

.Text = leftBit & newNumber

Else

.Text = "^p" & newNumber

End If

.Replacement.Text = ""

.Wrap = False

.Forward = False

.MatchWildcards = False

.Execute

End With

Selection.End = Selection.Start

If Selection.End = hereNow Then beep

If leftBit > "" Then

Selection.MoveRight Unit:=wdCharacter, Count:=4

Else

Selection.MoveRight Unit:=wdCharacter, Count:=1

End If

'Add this to leave F&R dialogue in a sensible state

With Selection.Find

.Wrap = wdFindContinue

.Forward = True

End With

Selection.End = Selection.Start

End Sub

**Sub NextNumber()**

**'** Paul Beverley - Version 11.10.10

' Find next section number

allowedChars = "0123456789."

theNumber = ""

Selection.End = Selection.Start

startPos = Selection.Start

Selection.Start = startPos - 4

leftBit = Selection

Selection.Start = startPos

pos = 1

dotPos = 0

Do

thisChar = Selection

theNumber = theNumber + thisChar

If thisChar = "." Then

prevDotPos = dotPos

dotPos = pos

End If

Selection.MoveRight Unit:=wdCharacter, Count:=1

pos = pos + 1

Loop Until InStr(allowedChars, thisChar) = 0

If dotPos > 0 Then

lastNumber = Mid(theNumber, dotPos + 1, pos - dotPos - 2)

If Val(lastNumber) = 0 Then

dotPos = prevDotPos

lastNumber = Mid(theNumber, dotPos + 1, pos - dotPos - 2)

End If

newNumber = Left(theNumber, dotPos) + Trim(Str(Val(lastNumber) + 1))

Else

lastNumber = Left(theNumber, pos - 2)

newNumber = Trim(Str(Val(lastNumber) + 1))

End If

hereNow = Selection.End

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = newNumber

.Replacement.Text = ""

.Wrap = False

.Forward = True

.MatchWildcards = False

.Execute

End With

If Selection.End = hereNow Then

beep

Selection.End = startPos

End If

'Add this to leave F&R dialogue in a sensible state

With Selection.Find

.Wrap = wdFindContinue

End With

Selection.End = Selection.Start

End Sub

**Sub NextNumberUp()**

**'** Paul Beverley - Version 11.10.10

' Find previous section number

allowedChars = "0123456789."

theNumber = ""

Selection.End = Selection.Start

startPos = Selection.Start

Selection.Start = startPos - 4

leftBit = Selection

Selection.Start = startPos

pos = 1

dotPos = 0

Do

thisChar = Selection

theNumber = theNumber + thisChar

If thisChar = "." Then

prevDotPos = dotPos

dotPos = pos

End If

Selection.MoveRight Unit:=wdCharacter, Count:=1

pos = pos + 1

Loop Until InStr(allowedChars, thisChar) = 0

If dotPos > 0 Then

lastNumber = Mid(theNumber, dotPos + 1, pos - dotPos - 2)

If Val(lastNumber) = 0 Then

dotPos = prevDotPos

lastNumber = Mid(theNumber, dotPos + 1, pos - dotPos - 2)

End If

newNumber = Left(theNumber, dotPos) + Trim(Str(Val(lastNumber) - 1))

Else

lastNumber = Left(theNumber, pos - 2)

newNumber = Trim(Str(Val(lastNumber) - 1))

End If

hereNow = Selection.End

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = newNumber

.Replacement.Text = ""

.Wrap = False

.Forward = False

.MatchWildcards = False

.Execute

End With

If Selection.End = hereNow Then

beep

Selection.End = startPos

End If

'Add this to leave F&R dialogue in a sensible state

With Selection.Find

.Wrap = wdFindContinue

.Forward = True

End With

Selection.End = Selection.Start

End Sub

**Sub FindNextNumber()**

' Paul Beverley - Version 09.01.13

' Jump from one number to the next - section, fig, table, etc

Selection.Words(1).Select

If Selection.Start = 0 Then

Selection.InsertBefore Text:=Chr(13)

Selection.MoveStart wdCharacter, 1

End If

wd = Selection

If InStr(wd, "Box") + InStr(wd, "Fig") + InStr(wd, "Tab") > 0 Then

Selection.Collapse wdCollapseEnd

Else

Selection.Collapse wdCollapseStart

End If

Selection.MoveStartUntil cset:=Chr(13) & " " & Chr(9), Count:=wdBackward

Selection.MoveEndUntil cset:=Chr(13) & " :" & Chr(9), Count:=wdForward

myDelay = 0.1

Set rng = ActiveDocument.Content

rng.Start = Selection.Start - 1

rng.End = Selection.Start

b4Text = rng

If b4Text <> Chr(13) Then

' If the first char isn't a return, is there one further back?

gotCR = False

For j = 1 To 10

rng.MoveStart wdCharacter, -1

b4Text = rng

If Asc(rng) = 13 Then

rng.MoveStart wdCharacter, 1

gotCR = True

Exit For

End If

Next j

If gotCR = True Then

b4Text = "^p" & rng.Text

Else

rng.MoveStart wdCharacter, 5

b4Text = rng

If InStr(b4Text, "Fig.") > 0 Then b4Text = "^p" & "Fig. "

End If

Else

rng.MoveStart wdCharacter, 1

b4Text = "^p" & rng.Text

End If

' At this point b4Txt has the preceding text, and

' the number is selected

allText = Selection

txtStart = Selection.Start

myLen = Len(allText)

' See if the next level section number is there

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = b4Text & allText & ".1"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If Selection.Find.Found = True Then GoTo theEnd

' So you've not found the ".1" just find the next heading

gotNumber = False

Do

'go find the next number of this level of heading

If InStr(allText, ".") = 0 Then

getNumber = Trim(Str(Val(allText) + 1)) & ".1"

i = 1

Else

For i = 1 To myLen

myText = Right(allText, i)

If gotNumber = True Then

If Asc(myText) < 48 Or Asc(myText) > 57 Then

theNumber = Val(Right(allText, i - 1))

Exit For

End If

End If

If Asc(myText) > 47 And Asc(myText) < 58 Then gotNumber = True

Next i

thisNumber = Val(theNumber)

getNumber = Left(allText, myLen - i + 1) & Trim(Str(thisNumber + 1))

End If

Selection.Collapse wdCollapseEnd

With Selection.Find

.Text = b4Text & getNumber

.Execute

End With

gotOne = Selection.Find.Found

If gotOne = True Then GoTo theEnd

If Len(b4Text) > 2 Then

' Where a table has got in the way, search again w/o CR

b4Text2 = Replace(b4Text, "^p", "")

With Selection.Find

.Text = b4Text2 & getNumber

.Execute

End With

gotOne = Selection.Find.Found

If gotOne = True Then

Beep

GoTo theEnd

End If

End If

' With section numbering, check for the next number in the sequence

getWas = getNumber

getNumber = Left(allText, myLen - i + 1) & Trim(Str(thisNumber + 2))

With Selection.Find

.Text = b4Text & getNumber

.Execute

End With

If Selection.Find.Found Then

Selection.Start = txtStart - 1

Selection.End = txtStart - 1

Selection.MoveRight Count:=1

b4Text = Replace(b4Text, "^p", "")

If Len(b4Text) = 0 Then b4Text = "Section "

MsgBox (b4Text & getWas & " missing!")

Exit Sub

Else

getWas2 = getNumber

getNumber = Left(allText, myLen - i + 1) & Trim(Str(thisNumber + 3))

With Selection.Find

.Text = b4Text & getNumber

.Execute

End With

If Selection.Find.Found Then

Selection.Start = txtStart - 1

Selection.End = txtStart - 1

Selection.MoveRight Count:=1

b4Text = Replace(b4Text, "^p", "")

If Len(b4Text) = 0 Then b4Text = "Section "

MsgBox (b4Text & getWas & " missing, AND " & getWas2 & "!")

Exit Sub

End If

End If

myLen = myLen - i

If myLen < 1 Then

GoTo theEnd

End If

allText = Left(allText, myLen)

myTime = Timer

Do

Loop Until Timer > myTime + myDelay

Beep

Loop Until gotOne = True

theEnd:

Selection.Collapse wdCollapseEnd

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "whereIwas" Then varExists = True: Exit For

Next v

If varExists = False Then

ActiveDocument.Variables.Add "whereIwas", txtStart

Else

ActiveDocument.Variables("whereIwas") = txtStart

End If

End Sub

**Sub FindPreviousNumber()**

' Paul Beverley - Version 08.01.13

' Jump back to the previous number

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "whereIwas" Then varExists = True: Exit For

Next v

If varExists = False Then

ActiveDocument.Variables.Add "whereIwas", 0

Else

Selection.Start = ActiveDocument.Variables("whereIwas")

Selection.Collapse wdCollapseStart

End If

' Use jumpScroll to set cursor to top of page

startHere = Selection.Start

Application.ScreenUpdating = False

Selection.EndKey Unit:=wdLine

lineEnd = Selection.Start

If Selection.Information(wdInEndnote) = True \_

Then inEndNotes = True

Selection.MoveDown Unit:=wdScreen, Count:=2

' Make sure that you've not dropped into a footnote ...

Do While Selection.Information(wdInFootnote) = True

Selection.MoveDown Unit:=wdLine, Count:=2

Loop

If inEndNotes = False Then

' Make sure that you've not dropped into an endnote ...

Do While Selection.Information(wdInEndnote) = True

Selection.MoveUp Unit:=wdLine, Count:=2

Loop

End If

Selection.End = lineEnd

Selection.MoveRight , 1

Selection.MoveUp , 1

Selection.End = startHere

Selection.Start = startHere

If lineEnd = startHere Then Selection.MoveLeft , 1

Application.ScreenUpdating = True

End Sub

**Sub ListItemFinder()**

' Paul Beverley - Version 28.08.14

' Jump to an auto-list number

myOffset = 0

If Selection.Start = Selection.End Then Selection.Expand wdWord

mySearchText = Trim(Selection)

mySearchNumber = Val(mySearchText)

If mySearchNumber = 0 Then MsgBox "Place the cursor in the citation number": Exit Sub

myDocName = Left(ActiveDocument.Name, InStr(ActiveDocument.Name, ".") - 1)

nowSelection = Selection

Windows(myDocName & ":1").Activate

If Selection = nowSelection Then Windows(myDocName & ":2").Activate

For i = myOffset + mySearchNumber To ActiveDocument.ListParagraphs.Count

num = ActiveDocument.ListParagraphs(i).Range.ListFormat.ListString

If Val(num) = mySearchNumber Then

ActiveDocument.ListParagraphs(i).Range.Select

Exit Sub

End If

Next i

MsgBox "Sorry, I can't find an item of that number."

End Sub

**Sub ListCheckStart()**

' Paul Beverley - Version 28.08.14

' Jump to an auto-list number

hereNow = Selection.Start

For i = 1 To ActiveDocument.ListParagraphs.Count

itemEnd = ActiveDocument.ListParagraphs(i).Range.End

If itemEnd > hereNow Then

MsgBox "Use: myOffset = " & Str(i - 1)

Exit Sub

End If

Next i

End Sub

**Sub TableNext()**

' Paul Beverley - Version 14.08.19

' Steps through tables, one by one

Set rng = ActiveDocument.range(0, Selection.End)

tableNum = rng.Tables.Count

totTables = ActiveDocument.Tables.Count

If tableNum = totTables Then

Beep

Exit Sub

End If

ActiveDocument.Tables(tableNum + 1).Select

Selection.Collapse wdCollapseStart

Set rng = Selection.range.Duplicate

ActiveDocument.ActiveWindow.LargeScroll Down:=1

ActiveDocument.ActiveWindow.SmallScroll Down:=1

Selection.MoveUp wdParagraph, 1

rng.Select

Selection.MoveEnd wdWord, 1

End Sub

**Sub TablePrevious()**

' Paul Beverley - Version 14.08.19

' Steps through tables, one by one

Set rng = ActiveDocument.range(0, Selection.End)

tableNum = rng.Tables.Count

totTables = ActiveDocument.Tables.Count

If tableNum < 2 Then

Beep

Exit Sub

End If

ActiveDocument.Tables(tableNum - 1).Select

Selection.Collapse wdCollapseStart

Set rng = Selection.range.Duplicate

ActiveDocument.ActiveWindow.LargeScroll Down:=1

ActiveDocument.ActiveWindow.SmallScroll Down:=1

Selection.MoveUp wdParagraph, 1

rng.Select

Selection.MoveEnd wdWord, 1

End Sub

**Sub BordersPrevious()**

' Paul Beverley - Version 10.10.19

' Finds previous paragraph with borders

Set rng = ActiveDocument.range(0, Selection.End)

paraNum = rng.Paragraphs.Count

totParas = ActiveDocument.Paragraphs.Count

If paraNum < totParas Then

For i = paraNum To 1 Step -1

If ActiveDocument.Paragraphs(i).range.Font.Borders(1).LineStyle <> \_

wdLineStyleNone Then Exit For

Next i

If i > 0 Then

ActiveDocument.Paragraphs(i).range.Select

Selection.Expand wdParagraph

Else

Beep

End If

End If

Selection.Collapse wdCollapseStart

End Sub

**Sub BordersNext()**

' Paul Beverley - Version 10.10.19

' Finds next paragraph with borders

selectFind = False

Set rng = ActiveDocument.range(0, Selection.End)

paraNum = rng.Paragraphs.Count

If ActiveDocument.Paragraphs(paraNum).range.End = \_

Selection.End Then paraNum = paraNum + 1

totParas = ActiveDocument.Paragraphs.Count

Set pRng = ActiveDocument.Paragraphs(paraNum).range.Duplicate

hasBorder = (pRng.Font.Borders(1).LineStyle <> wdLineStyleNone)

Set sRng = Selection.range.Duplicate

paraEnd = pRng.End

slcnStart = sRng.Start

If hasBorder = True Then

Set rng = ActiveDocument.range(slcnStart, slcnStart + 1)

styleNow = rng.Font.Borders(1).LineStyle

Set rng = ActiveDocument.range(slcnStart, paraEnd)

moreBorderInPara = (rng.Font.Borders(1).LineStyle <> wdLineStyleNone)

If styleNow <> wdLineStyleNone Then

' We are in a border, so find its end

endBorder = paraEnd

For ptr = slcnStart + 1 To paraEnd

Set rng = ActiveDocument.range(ptr, ptr + 1)

styleHere = rng.Font.Borders(1).LineStyle

If styleHere = wdLineStyleNone Then

endBorder = ptr

Exit For

End If

Next ptr

Set rng = ActiveDocument.range(endBorder, paraEnd)

If rng.Font.Borders(1).LineStyle = wdLineStyleNone \_

Then moreBorderInPara = False

End If

If moreBorderInPara = True Then

' ptr is now at a no-border character and there's more

For ptr = endBorder To paraEnd

Set rng = ActiveDocument.range(ptr, ptr + 1)

If rng.Font.Borders(1).LineStyle > 0 Then Exit For

Next ptr

borderStart = ptr

For ptr = borderStart + 1 To ActiveDocument.Content.End

Set rng = ActiveDocument.range(ptr, ptr + 1)

If rng.Font.Borders(1).LineStyle = 0 Then Exit For

Next ptr

borderEnd = ptr

Set rng = ActiveDocument.range(borderStart, borderEnd)

rng.Select

If selectFind = False Then Selection.Collapse wdCollapseStart

Exit Sub

End If

End If

' Check by paragraph

For i = paraNum + 1 To totParas

gotBorders = ActiveDocument.Paragraphs(i).range.Font.Borders(1).LineStyle <> \_

wdLineStyleNone

If gotBorders Then Exit For

Next i

If gotBorders = 0 Then

Beep

Exit Sub

End If

' Find first border bit

paraSt = ActiveDocument.Paragraphs(i).range.Start

paraEnd = ActiveDocument.Paragraphs(i).range.End

For ptr = paraSt To paraEnd

Set rng = ActiveDocument.range(ptr, ptr + 1)

If rng.Font.Borders(1).LineStyle > 0 Then Exit For

Next ptr

borderStart = ptr

For ptr = borderStart + 1 To ActiveDocument.Content.End

Set rng = ActiveDocument.range(ptr, ptr + 1)

If rng.Font.Borders(1).LineStyle = 0 Then Exit For

Next ptr

borderEnd = ptr

Set rng = ActiveDocument.range(borderStart, borderEnd)

rng.Select

If selectFind = False Then Selection.Collapse wdCollapseStart

End Sub

**Sub FigTableBoxLister()**

' Paul Beverley - Version 09.07.20

' Finds figure/table/box elements and their citations

mainDocBackToFront = True

listCaptions = True

capTag = "<Cap>"

' Number of characters of the caption listed

maxLen = 60

' Which letters are searched for as, e.g. "Figure 16.5c/d/e etc"

myLetters = "a-h"

Set mainDoc = ActiveDocument

CR = vbCr

CR2 = CR & CR

tb = Chr(9)

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

' Prepare a range for plurals

Set rng3 = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "F[iI][Gg][uUrReE. ]{1,4}[1-9]"

.Replacement.Text = ""

.Forward = True

.MatchCase = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

isFigure = rng.Find.Found

Set rng = ActiveDocument.Content

With rng.Find

.Text = "T[aA][bB][lL][eE] [1-9]"

.MatchWildcards = True

.Execute

End With

isTable = rng.Find.Found

Set rng = ActiveDocument.Content

With rng.Find

.Text = "Box [1-9]"

.MatchWildcards = True

.Execute

End With

isBox = rng.Find.Found

If isFigure Then

foundList = ""

rng.Start = 0

rng.End = 0

With rng.Find

.Text = "F[iI][Gg][uUrReE. ]{1,4}[0-9.:" & myLetters & "]{1,}"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Set rng2 = rng.Paragraphs(1).Range

pa = rng2.Text

If InStr(pa, ">") > 0 Then

pa = Mid(pa, InStr(pa, ">") + 1)

End If

wd = rng.Text

If InStr(pa, wd) > 2 Then

foundList = foundList & tb

Else

wdLess = wd

If Right(wd, 1) = "." Or Right(wd, 1) = ":" Then wdLess = \_

Left(wd, Len(wd) - 1)

If InStr(foundList, CR & wd & CR) > 0 Or InStr(foundList, CR & \_

wd & ":" & CR) > 0 Or InStr(foundList, CR & wd & "." & CR) \_

> 0 Or InStr(foundList, CR & wdLess & CR) > 0 Then foundList \_

= foundList & tb & tb

End If

foundList = foundList & rng & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

' Now look for "Figures"

pluralList = ""

rng.Start = 0

rng.End = 0

With rng.Find

.Text = "Figures "

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Set rng2 = ActiveDocument.Range(rng.Start, rng.End)

rng2.MoveEnd wdWord, 1

Do

stopNow = False

rng2.MoveEnd wdWord, 1

myText = rng2.Words(rng2.Words.Count)

If LCase(myText) <> UCase(myText) And \_

Len(myText) > 3 Then stopNow = True

If Trim(myText) = "and" Then stopNow = False

If Asc(myText) = 13 Then stopNow = True

Loop Until stopNow

rng2.MoveEnd wdWord, -1

pluralList = pluralList & rng2 & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Documents.Add

Set figDoc = ActiveDocument

Selection.TypeText foundList & CR2

For Each para In ActiveDocument.Paragraphs

pa = Replace(para.Range.Text, Chr(13), "")

If Right(pa, 1) = "." Or Right(pa, 1) = ":" Then pa = \_

Left(pa, Len(pa) - 1)

If InStr(foundList, tb & pa) = 0 And InStr(pa, tb) = 0 Then \_

para.Range.HighlightColorIndex = wdYellow

If InStr(foundList, CR & Replace(pa, Chr(9), "")) = 0 And InStr(pa, tb) \_

> 0 Then para.Range.HighlightColorIndex = wdYellow

Next

If Len(pluralList) > 1 Then

Selection.HomeKey Unit:=wdStory

Selection.InsertBefore Text:=CR2 & "Plurals" & CR2

Selection.MoveStart wdWord, 2

Selection.MoveEnd wdWord, -1

Selection.Font.Bold = True

Selection.Collapse wdCollapseEnd

Selection.MoveRight wdWord, 1

Selection.TypeText pluralList & CR2

End If

Selection.HomeKey Unit:=wdStory

rng.Start = 0: rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = capTag & "Fig"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

rng.Expand wdParagraph

capText = Replace(rng.Text, vbTab, " ")

If Len(capText) > maxLen Then capText = \_

Left(capText, maxLen) & vbCr

Selection.TypeText capText

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Selection.HomeKey Unit:=wdStory

End If

If isTable Then

foundList = ""

rng.Start = 0

rng.End = 0

With rng.Find

.Text = "T[aA][bB][lL][eE] [0-9.:" & myLetters & "]{1,}"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Set rng2 = rng.Paragraphs(1).Range

pa = rng2.Text

If InStr(pa, ">") > 0 Then

pa = Mid(pa, InStr(pa, ">") + 1)

End If

wd = rng.Text

If InStr(pa, wd) > 2 Then

foundList = foundList & tb

Else

wdLess = wd

If Right(wd, 1) = "." Or Right(wd, 1) = ":" Then wdLess = \_

Left(wd, Len(wd) - 1)

If InStr(foundList, CR & wd & CR) > 0 Or InStr(foundList, CR & \_

wd & ":" & CR) > 0 Or InStr(foundList, CR & wd & "." & CR) \_

> 0 Or InStr(foundList, CR & wdLess & CR) > 0 Then foundList \_

= foundList & tb & tb

End If

foundList = foundList & rng & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

pluralList = ""

rng.Start = 0

rng.End = 0

With rng.Find

.Text = "Tables "

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rng3.Start = rng.Start

rng3.End = rng.End

rng3.MoveEnd wdWord, 1

Do

stopNow = False

rng3.MoveEnd wdWord, 1

myText = rng3.Words(rng3.Words.Count)

If LCase(myText) <> UCase(myText) And \_

Len(myText) > 3 Then stopNow = True

If Trim(myText) = "and" Then stopNow = False

If Asc(myText) = 13 Then stopNow = True

Loop Until stopNow

rng3.MoveEnd wdWord, -1

pluralList = pluralList & rng3 & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Documents.Add

Set tableDoc = ActiveDocument

Selection.TypeText foundList & CR2

For Each para In ActiveDocument.Paragraphs

pa = Replace(para.Range.Text, Chr(13), "")

If Right(pa, 1) = "." Or Right(pa, 1) = ":" Then pa = \_

Left(pa, Len(pa) - 1)

If InStr(foundList, tb & pa) = 0 And InStr(pa, tb) = 0 Then \_

para.Range.HighlightColorIndex = wdYellow

If InStr(foundList, CR & Replace(pa, Chr(9), "")) = 0 And InStr(pa, tb) \_

> 0 Then para.Range.HighlightColorIndex = wdYellow

Next

If Len(pluralList) > 2 Then

Selection.HomeKey Unit:=wdStory

Selection.InsertBefore Text:=CR2 & "Plurals" & CR2

Selection.MoveStart wdWord, 2

Selection.MoveEnd wdWord, -1

Selection.Font.Bold = True

Selection.Collapse wdCollapseEnd

Selection.MoveRight wdWord, 1

Selection.TypeText pluralList & CR2

End If

Selection.HomeKey Unit:=wdStory

rng.Start = 0: rng.End = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = capTag & "Tab"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

rng.Expand wdParagraph

capText = Replace(rng.Text, vbTab, " ")

If Len(capText) > maxLen Then capText = \_

Left(capText, maxLen) & vbCr

Selection.TypeText capText

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Selection.HomeKey Unit:=wdStory

End If

If isBox Then

foundList = ""

rng.Start = 0

rng.End = 0

With rng.Find

.Text = "Box [0-9.:" & myLetters & "]{1,}"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

Set rng2 = rng.Paragraphs(1).Range

pa = rng2.Text

If InStr(pa, ">") > 0 Then

pa = Mid(pa, InStr(pa, ">") + 1)

End If

wd = rng.Text

If InStr(pa, wd) > 2 Then

foundList = foundList & tb

Else

wdLess = wd

If Right(wd, 1) = "." Or Right(wd, 1) = ":" Then wdLess = \_

Left(wd, Len(wd) - 1)

If InStr(foundList, CR & wd & CR) > 0 Or InStr(foundList, CR & \_

wd & ":" & CR) > 0 Or InStr(foundList, CR & wd & "." & CR) \_

> 0 Or InStr(foundList, CR & wdLess & CR) > 0 Then foundList \_

= foundList & tb & tb

End If

foundList = foundList & rng & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

rng.Start = 0

rng.End = 0

pluralList = ""

With rng.Find

.Text = "Boxes "

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rng3.Start = rng.Start

rng3.End = rng.End

rng3.MoveEnd wdWord, 1

Do

stopNow = False

rng3.MoveEnd wdWord, 1

myText = rng3.Words(rng3.Words.Count)

If LCase(myText) <> UCase(myText) And \_

Len(myText) > 3 Then stopNow = True

If Trim(myText) = "and" Then stopNow = False

If Asc(myText) = 13 Then stopNow = True

Loop Until stopNow

rng3.MoveEnd wdWord, -1

pluralList = pluralList & rng3 & CR

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Documents.Add

Selection.TypeText foundList

For Each para In ActiveDocument.Paragraphs

pa = Replace(para.Range.Text, Chr(13), "")

If Right(pa, 1) = "." Or Right(pa, 1) = ":" Then pa = \_

Left(pa, Len(pa) - 1)

If InStr(foundList, tb & pa) = 0 And InStr(pa, tb) = 0 Then \_

para.Range.HighlightColorIndex = wdYellow

If InStr(foundList, CR & Replace(pa, Chr(9), "")) = 0 And InStr(pa, tb) \_

> 0 Then para.Range.HighlightColorIndex = wdYellow

Next

If Len(pluralList) > 2 Then

Selection.InsertAfter Text:=CR2 & "Plurals" & CR

Selection.Collapse wdCollapseEnd

Selection.MoveStart wdWord, -2

Selection.MoveEnd wdWord, -1

Selection.Font.Bold = True

Selection.EndKey Unit:=wdStory

Selection.TypeText pluralList

End If

Selection.HomeKey Unit:=wdStory

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z]@>"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If mainDocBackToFront Then

mainDoc.Activate

Else

' bring windows to the front

If isTable Then tableDoc.Activate

If isFigure Then figDoc.Activate

End If

Beep

End Sub

**Sub CaptionsListAll()**

**'** Paul Beverley - Version 25.05.23

' Lists all paragraphs with bold Figure, Table, Box

captionsAreBold = True

maxWords = 40

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = wdYellow

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13[FTB][iao][gbx][ul .]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

rng.MoveStart , 1

rng.Expand wdParagraph

grabThisOne = True

If captionsAreBold = True And rng.Font.Bold \_

= False Then grabThisOne = False

If rng.Words.count > maxWords Then grabThisOne = False

If grabThisOne = True Then rng.HighlightColorIndex = wdNoHighlight

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

For i = rng.Tables.count To 1 Step -1

rng.Tables(i).Delete

Next i

Beep

End Sub

**Sub ParaShort()**

**'** Paul Beverley - Version 06.11.10

' Finds medium-length paragraphs

minLen = 20

maxLen = 150

Do

Selection.MoveDown Unit:=wdParagraph, Count:=1, Extend:=wdExtend

myText = Selection

Selection.Start = Selection.End

Loop Until Len(myText) < 150 And Len(myText) > minLen

Selection.MoveUp Unit:=wdParagraph, Count:=1

startHere = Selection.Start - 1

Selection.MoveDown Unit:=wdScreen, Count:=1

' Make sure that you've not dropped into a footnote ...

Do

Selection.MoveDown Unit:=wdScreen, Count:=1

Loop Until Selection.Start > startHere

Selection.End = startHere

Selection.MoveRight Unit:=wdCharacter, Count:=1

Selection.MoveDown Unit:=wdParagraph, Count:=1

End Sub

**Sub ListHighlighter()**

**'** Paul Beverley - Version 19.10.10

' This highlights all 'short' paragraphs in a text

minLength = 10

MaxLength = 150

MyColour = wdBrightGreen

OldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = MyColour

For Each myPara In ActiveDocument.Paragraphs

paraWords = myPara

lineLen = Len(paraWords)

If lineLen < MaxLength And lineLen > minLength Then

myPara.Range.HighlightColorIndex = MyColour

End If

Next

Options.DefaultHighlightColorIndex = OldColour

Selection.HomeKey Unit:=wdStory

End Sub

**Sub PageHopper()**

**'** Paul Beverley - Version 08.12.10

' Go to page x

' Alt-G

myPage = InputBox("Go to page?", "Page Hopper")

Selection.GoTo What:=wdGoToPage, Count:=myPage

End Sub

**Sub PageHopper2()**

**'** Paul Beverley - Version 18.12.10

' Go to page x

' Alt-G

myPage = InputBox("Go to page?", "Page Hopper")

Selection.GoTo What:=wdGoToPage, Count:=myPage

startHere = Selection.Start - 1

Selection.MoveDown Unit:=wdScreen, Count:=1

' Make sure that you've not dropped into a footnote ...

Do

Selection.MoveDown Unit:=wdScreen, Count:=1

Loop Until Selection.Start > startHere

Selection.End = startHere

Selection.MoveRight Unit:=wdCharacter, Count:=1

End Sub

**Sub JumpScroll()**

' Paul Beverley - Version 12.12.22

' Scrolls current line to the top of the page

endLineNumber = 2

Set rng = Selection.Range.Duplicate

ActiveDocument.ActiveWindow.LargeScroll down:=1

rng.Select

ActiveDocument.ActiveWindow.SmallScroll down:=1

If endLineNumber > 1 Then

Selection.MoveUp wdLine, endLineNumber - 1

Selection.MoveDown wdLine, endLineNumber - 1

End If

End Sub

**Sub ListOfFound()**

' Paul Beverley - Version 16.01.21

' Lists all occurrences of selected text

charsBefore = 50

charsAfter = 50

doBold = False

doHighlight = True

myColour = wdYellow

allowCaseSensitive = True

allowStartOfLine = True

crSymbol = "|"

tabSymbol = "~"

pgSymbol = ""

Set myDoc = ActiveDocument

myDocName = ActiveDocument.Name

dotPosn = InStr(myDocName, ".")

If dotPosn > 0 Then

myDocName = Left(myDocName, InStr(myDocName, ".") - 1)

End If

caseMatch = False

alignList = False

startOfLine = False

Dim v As Variable

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "v\_docName" Then varExists = True

Next v

If varExists Then GoTo jumpBack

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

' In the main text, find the search word(s) then create the list

findString = InputBox(myType, "Search for?", Trim(Selection))

If findString = "" Then Exit Sub

If startOfLine Then findString = "^13" & findString

If allowCaseSensitive Then

myResponse = MsgBox("Allow any case?", vbQuestion \_

+ vbYesNoCancel, "List Of Found")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then caseMatch = True

End If

If allowStartOfLine = True Then

myResponse = MsgBox("Text anywhere in paragraph?", vbQuestion \_

+ vbYesNoCancel, "List Of Found")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then startOfLine = True

End If

' Now find the list if there is one

For Each myWnd In Application.Windows

myWnd.Document.Activate

gotList = (ActiveDocument.Words(1) = "ListOfFound")

If gotList = True Then Exit For

Next myWnd

' No list found, so create it and add document's name as a variable

If gotList = False Then

Documents.Add

ActiveDocument.Variables.Add "v\_docName", myDocName

Else

Selection.WholeStory

Selection.Delete

ActiveDocument.Variables("v\_docName") = myDocName

End If

' Prepare list document

Selection.TypeText "ListOfFound: " & findString & vbCr & vbCr

ActiveDocument.Paragraphs(1).Range.Font.Bold = True

ActiveDocument.Words(1).Font.Bold = False

ActiveDocument.Words(2).Font.Bold = False

Set listDoc = ActiveDocument

myDoc.Activate

Selection.MoveLeft , 1

Selection.MoveRight , 1

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

If startOfLine = True Then

.Text = "^13" & findString

Else

.Text = findString

End If

.Wrap = wdFindStop

.Forward = True

.MatchCase = caseMatch

.MatchWildcards = False

.Execute

End With

If startOfLine = True Then

rng.Start = rng.Start + 1

charsAfter = 50 + charsBefore

charsBefore = 0

End If

inAtable = rng.Information(wdWithInTable)

listDoc.Activate

Selection.EndKey Unit:=wdStory

myCount = 0

myResults = ""

Do While rng.Find.Found = True

myCount = myCount + 1

rng.Start = rng.Start - charsBefore

rng.End = rng.End + charsAfter

rng.MoveStartUntil cset:=ChrW(8216) & ChrW(8220) & " '(""" & Chr(13), Count:=wdBackward

rng.MoveEndUntil cset:=ChrW(8217) & ChrW(8221) & " '""" & Chr(13), Count:=wdForward

myText = rng.Text

If startOfLine = True Then

crPosn = InStr(myText, vbCr)

If crPosn > 0 Then myText = Left(myText, crPosn)

End If

rng.Collapse wdCollapseStart

rng.Start = rng.Start + 10

myText = Replace(myText, vbCr, crSymbol)

myText = Replace(myText, Chr(12), pgSymbol)

myText = Replace(myText, vbTab, tabSymbol)

If inAtable Then myText = "iziz" & myText & "zizi"

' Find search word(s)and embolden it

myOffset = InStr(LCase(myText), LCase(findString))

myResults = myResults & Left(myText, myOffset - 1)

myResults = myResults & "uhuh"

myResults = myResults & Mid(myText, myOffset, Len(findString))

myResults = myResults & "huhu"

myResults = myResults & Mid(myText, myOffset + Len(findString)) & vbCr

rng.Find.Execute

If startOfLine = True Then rng.Start = rng.Start + 1

inAtable = rng.Information(wdWithInTable)

Loop

Selection.TypeText myResults

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "iziz(\*)zizi"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Font.Italic = True

.Forward = True

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "uhuh(\*)huhu"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

If doHighlight = True Then .Replacement.Highlight = True

If doBold = True Then .Replacement.Font.Bold = True

.Replacement.Font.Underline = True

.Forward = True

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

Selection.HomeKey Unit:=wdStory

Selection.MoveDown , 1

Selection.TypeText vbCr & "Found: " & Str(myCount) & vbCr

Selection.MoveLeft , 2

Selection.Expand wdWord

Selection.Font.Bold = True

Selection.HomeKey Unit:=wdStory

Beep

Exit Sub

jumpBack:

' Collect the line of text from the list

findWhole = Selection.Start = Selection.End

If findWhole Then

Selection.Expand wdParagraph

Selection.MoveEnd , -1

myText = Selection

isSomeHighlight = Selection.Range.HighlightColorIndex

If isSomeHighlight > 0 Then

Selection.Collapse wdCollapseStart

Selection.MoveEnd , 1

myOffset = 1

Do

myOffset = myOffset + 1

Selection.MoveStart , 1

Selection.MoveEnd , 1

Loop Until Selection.Range.Font.Underline = 1

myLen = 0

Do

myLen = myLen + 1

Selection.MoveStart , 1

Selection.MoveEnd , 1

Loop Until Selection.Range.Font.Underline = 0

myOffset = myOffset - 1

Else

myOffset = 0

myLen = Len(myText)

End If

Else

myText = Selection

Selection.Expand wdParagraph

myOffset = 0

myLen = Len(myText)

End If

myText = Replace(myText, "^", "^^")

myText = Replace(myText, crSymbol, "^p")

myText = Replace(myText, tabSymbol, "^t")

myText = Replace(myText, pgSymbol, "^m")

' Find the original document

myDocName = ActiveDocument.Variables("v\_docName")

For Each myWnd In Application.Windows

myTitle = myWnd.Document.Name

If InStr(myTitle, myDocName) Then

myWnd.Document.Activate

Exit For

End If

Next myWnd

' Find that text

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute

End With

Selection.Collapse wdCollapseStart

Set wasSelected = Selection.Range

Application.ScreenUpdating = False

Selection.EndKey Unit:=wdStory

wasSelected.Select

Application.ScreenUpdating = True

ActiveDocument.ActiveWindow.SmallScroll down:=1

Selection.MoveRight , 1

If Not (Selection.Find.Found) Then

Beep

Selection.EndKey Unit:=wdStory

Else

Selection.MoveStart , -1

Selection.MoveStart , myOffset

Selection.MoveEnd , myLen

End If

End Sub

**Sub ListofHeadings()**

' Paul Beverley - Version 06.12.18

' Create a list of all headings

Dim myStyle(10)

' How many styles

numStyles = 3

' What are their names

myStyle(1) = "Heading 1"

myStyle(2) = "Heading 2"

myStyle(3) = "Heading 3"

' minimum heading length

minLength = 8

Dim v As Variable, listDoc, textDoc As Document

Dim fText, lineText As String

Dim textArray(500), styleArray(500)

nameNow = ""

' First check if there are any styles specified

endPara = ActiveDocument.Paragraphs.Count

myStyles = 0

For i = 0 To 5

Set rng = ActiveDocument.Paragraphs(endPara - i).range

myTestPara = rng

myTestPara = Left(myTestPara, Len(myTestPara) - 1)

If myTestPara = "Heading" Then

' As long as it's a heading type, record it

myStyles = myStyles + 1

thisStyle = rng.Style

myStyle(myStyles) = thisStyle

' and remember where it starts

headingFirst = rng.Start

End If

Next i

If myStyles > 0 Then numStyles = myStyles

' Are we in the List of headings? If so go to FoldBack

Set rng = ActiveDocument.Paragraphs(1).range

fText = rng

If InStr(fText, "List of headings") > 0 Then GoTo FoldBack

' We are in the text, not the list

findThis = Selection

newList = False

If Selection.Start <> Selection.End Then

' If some text is selected, offer to create a new list

myResponse = MsgBox("Create new list?", vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

newList = True

End If

Set textDoc = ActiveDocument

foundList = False

For Each doc In Documents

If aDoc <> "Normal.dot" Then

dfgsdf = doc.Name

Set rng = doc.Paragraphs(1).range

fText = rng

If InStr(fText, "List of headings") Then

If newList = False Then

Exit Sub

Else

Set listDoc = doc

foundList = True

End If

End If

End If

Next doc

' Go back to text file

asdfasd = textDoc.Name

textDoc.Activate

ActiveDocument.range.Copy

'Create a dummy file to avoid changing the original document

Documents.Add

ActiveDocument.range.Paste

' Collect all the headings

Set rng = ActiveDocument.Content

For n = 1 To numStyles

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Replacement.Text = "abbb^&zxzx^p"

.MatchWildcards = False

.Style = myStyle(n)

.Wrap = False

.Format = True

.Execute Replace:=wdReplaceAll

End With

Next n

i = 0

Do

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "abbb(\*)zxzx"

.Replacement.Text = "\1"

.MatchWildcards = True

.Wrap = False

.Format = True

.Execute Replace:=wdReplaceOne

End With

gotOne = (rng.Find.Found = True)

If gotOne = True Then

i = i + 1

textArray(i) = rng

styleArray(i) = rng.Style

End If

Loop Until gotOne = False

iMax = i

' Get rid of the temporary document

ActiveDocument.Close SaveChanges:=wdDoNotSaveChanges

' Find a List of headings document ...

gottaDoc = False

For Each doc In Documents

doc.Activate

Set rng = ActiveDocument.Paragraphs(1).range

fText = rng

If InStr(fText, "List of headings") Then

gottaDoc = True: Exit For

End If

Next

' ... or open a new one

If foundList = True Then

listDoc.Activate

Selection.WholeStory

Selection.Delete

Else

Documents.Add

Set listDoc = ActiveDocument

textDoc.Activate

Set rng = ActiveDocument.Content

rng.Start = headingFirst

rng.Select

Selection.Copy

Selection.HomeKey Unit:=wdStory

listDoc.Activate

Selection.Paste

End If

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="List of headings" & vbCrLf

Selection.MoveUp Unit:=wdLine, Extend:=wdExtend

Selection.Style = wdStyleNormal

Selection.Start = Selection.End

' Type in the lines of headings text

For i = 1 To iMax

If Len(textArray(i)) > minLength Then

Selection.TypeText Text:=textArray(i)

Selection.MoveUp Unit:=wdLine, Extend:=wdExtend

Selection.Style = styleArray(i)

Selection.Collapse wdCollapseEnd

End If

Next

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^12"

.Replacement.Text = "^p"

.MatchWildcards = False

.Wrap = False

.Format = True

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p^p"

.Replacement.Text = "^p"

.MatchWildcards = False

.Wrap = False

.Format = True

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Heading^p"

.Replacement.Text = ""

.MatchWildcards = False

.Wrap = False

.Format = True

.Execute Replace:=wdReplaceAll

End With

With ActiveDocument.Styles(wdStyleNormal).ParagraphFormat

.KeepWithNext = True

End With

Exit Sub

FoldBack:

' Select the cursor line

Selection.HomeKey Unit:=wdLine

Selection.EndKey Unit:=wdLine, Extend:=wdExtend

Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend

fText = Selection

fStyle = Selection.Style

' Find the right document

Set rng = ActiveDocument.Paragraphs(2).range

firstHeading = rng

DocName = ""

For Each doc In Documents

If doc <> "Normal.dot" Then

doc.Activate

Set rng = ActiveDocument.Paragraphs(1).range

topText = rng

If topText = firstHeading Then

DocName = ActiveDocument.Name

End If

If DocName > "" Then Exit For

End If

Next

If DocName = "" Then

MsgBox ("Can't find file: " & firstHeading)

Exit Sub

End If

' Try to find the searched-for line

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fText

.Replacement.Text = ""

.Style = fStyle

.MatchCase = False

.MatchWildcards = False

.Execute

End With

' select the searched-for heading

If Selection.Find.Found Then

Selection.EndKey Unit:=wdLine

Else

' Somehow we missed it!

If nameNow > "" Then

listDoc.Activate

MsgBox ("For some reason I can't find this heading.")

End If

End If

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fText

.MatchCase = False

End With

End Sub

**Sub ListAllHeadings()**

' Paul Beverley - Version 04.11.22

' Creates a list of all headings by style name

myStyles = "Heading 1,Heading 2,Heading 3"

Set myFile = ActiveDocument

Documents.Add

Set rng = ActiveDocument.Content

For Each pa In myFile.Paragraphs

myStart = pa.Range.Start

myEnd = pa.Range.End

myLen = Len(pa.Range.Text)

If myLen = myEnd - myStart Then

sty = pa.Range.Style

Else

sty = ""

End If

DoEvents

If Len(sty) > 5 Then

If InStr(myStyles, sty) > 0 Then

Debug.Print sty, pa.Range.Text

rng.FormattedText = pa.Range.FormattedText

rng.Collapse wdCollapseEnd

End If

End If

DoEvents

Next pa

End Sub

**Sub StyleDetector()**

' Paul Beverley - Version 14.08.19

' Displays or speaks the current style name

useVoice = True

' useVoice = False

useMessageBox = False

' useMessageBox = True

Set speech = New SpVoice

' Set speech = Nothing

selStyle = Selection.range.Style

Set rng = Selection.range.Duplicate

rng.Expand wdParagraph

paraStyle = rng.Style

myPrompt = "Paragraph style: " & paraStyle

myVoice = paraStyle

Set rng = Selection.range.Duplicate

If rng.Start <> rng.End Then

' Debug.Print rng.Start, rng.End

For i = rng.Start To rng.End - 1

Set rng2 = ActiveDocument.range(i, i + 1)

If rng2.Style <> paraStyle Then

myPrompt = myPrompt & vbCr & vbCr & \_

"Text style: " & rng2.Style

myVoice = myVoice & " and also " & rng2.Style

If useVoice = False Then Beep

Exit For

End If

Next i

Else

If paraStyle <> selStyle Then

If useVoice = False Then Beep

myPrompt = myPrompt & vbCr & vbCr & \_

"Text style: " & selStyle

myVoice = myVoice & " and also " & selStyle

End If

End If

If Selection.Font.SmallCaps = True Then

myVoice = myVoice & " with small caps"

End If

If useVoice = True Then

speech.Speak myVoice, SVSFPurgeBeforeSpeak

StatusBar = myVoice

Else

If useMessageBox Then

MsgBox myVoice

Else

StatusBar = myVoice

End If

End If

End Sub

**Sub FontColourReader()**

' Paul Beverley - Version 14.08.19

' Reads style font colour + any applied colour

showHex = True

showDecimal = True

myMix = 9999999

paraColourIndex = ActiveDocument.Styles(Selection.range.Style).Font.ColorIndex

paraColour = ActiveDocument.Styles(Selection.range.Style).Font.Color

myColourIndex = Selection.range.Font.ColorIndex

myColour = Selection.range.Font.Color

If showHex = True Then

myMessage = ""

myMessage = myMessage & "Style font colour = " & Hex(paraColour) & vbCr

If myColour = myMix Then

myMessage = myMessage & "Mixed colours" & vbCr

Else

If paraColour = myColour Then

myMessage = myMessage & "No applied colour" & vbCr

Else

myMessage = myMessage & "Applied colour = " & Hex(myColour) & vbCr

End If

End If

MsgBox myMessage

End If

If showDecimal = True Then

myMessage = ""

myMessage = myMessage & "Style font colour = " & paraColourIndex & vbCr

If myColourIndex = myMix Then

myMessage = myMessage & "Mixed colours" & vbCr

Else

If paraColourIndex = myColourIndex Then

myMessage = myMessage & "No applied colour" & vbCr

Else

myMessage = myMessage & "Applied colour = " & myColourIndex & vbCr

End If

End If

MsgBox myMessage

End If

End Sub

**Sub SearchTheseWords()**

' Paul Beverley - Version 13.09.19

' Finds the next occurrence of any of a list of words

myWords = ":and:or:but:so:yet:if:"

Set rng = Selection.range.Duplicate

' If only a tiny selection...

If rng.Words.Count < 3 Then rng.Collapse wdCollapseEnd

' or nothing selected, search from cursor to the end of the file

If rng.Start = rng.End Then

rng.End = ActiveDocument.Content.End

End If

myWords = ":" & myWords & ":"

For Each wd In rng.Words

myTest = ":" & LCase(Trim(wd)) & ":"

If InStr(myWords, myTest) > 0 Then

wd.Select

Exit For

End If

DoEvents

Next wd

End Sub

**Sub SplitScreen()**

' Paul Beverley - Version 28.05.21

' Splits the screen

perCent = 60

If ActiveWindow.SplitVertical = perCent Then

ActiveWindow.SplitVertical = 100

Else

ActiveWindow.SplitVertical = perCent

End If

ActiveWindow.Panes(1).Activate

End Sub

**Sub CommentAdd()**

**'** Paul Beverley - Version 26.02.24

' Adds a comment

attachToWord = True

' If False, it attaches to the current sentence

useSingleQuotes = True

' If False, it uses double quotes

useCommentPane = False

paneZoom = 240

copySelectedText = True

attrib1 = ""

attrib2 = ""

attrib1 = ""

attrib2 = ""

postText = ""

addPageNum1 = False

addLineNum1 = False

addPageNum2 = False

addLineNum2 = False

highlightTheText = False

textHighlightColour = wdYellow

colourTheText = False

textColour = wdColorBlue

removeHighlight = True

If Selection.Start = Selection.End Then

If attachToWord = True Then

Selection.Expand wdWord

Else

Selection.Expand wdSentence

End If

End If

myStart = Selection.Start

wasEnd = Selection.End

Set rng = Selection.Range

rng.Collapse wdCollapseEnd

rng.MoveEnd , 1

pageNum = rng.Information(wdActiveEndAdjustedPageNumber)

lineNum = rng.Information(wdFirstCharacterLineNumber)

If myStart <> wasEnd Then

If Right(Selection, 1) = Chr(32) Or Right(Selection, 1) = Chr(13) Then

Selection.MoveEnd wdCharacter, -1

wasEnd = wasEnd - 1

End If

With ActiveWindow.View

showCmnts = .ShowComments

showInsDels = .ShowInsertionsAndDeletions

showFormats = .ShowFormatChanges

trackOnOffState = .ShowRevisionsAndComments

End With

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

' Either highlight it ...

If highlightTheText = True Then Selection.Range.HighlightColorIndex \_

= textHighlightColour

' And/or change the text colour to red

If colourTheText = True Then Selection.Font.Color = textColour

' Now create the comment

Selection.Copy

Selection.Collapse wdCollapseEnd

Set rng = Selection.Range

Do

rng.MoveEnd , 1

DoEvents

Loop Until Right(rng.Text, 1) = Chr(13)

Selection.End = rng.End

Selection.Collapse wdCollapseEnd

extraBitStart = Selection.Start

If addPageNum1 = True Then attrib1 = attrib1 & "(p. " & \_

pageNum & ") "

If addLineNum1 = True Then attrib1 = attrib1 & "(line " & \_

lineNum & ") "

Selection.TypeText attrib1

If copySelectedText = True Then

If useSingleQuotes = True Then

Selection.TypeText ChrW(8216) & ChrW(8217)

Else

Selection.TypeText ChrW(8220) & ChrW(8221)

End If

' Move back to between the close and open quotes

Selection.MoveEnd , -1

' Paste in a copy of the selected text

Selection.Paste

' Move back past the close quote

Selection.MoveRight Count:=1

End If

If postText > "" Then

Selection.TypeText postText

Else

Selection.TypeText " " & ChrW(8211) & " "

End If

Selection.Start = extraBitStart

' If wanted, unhighlight the text clear

If removeHighlight = True Then Selection.Range.HighlightColorIndex = wdAuto

Selection.Range.Revisions.AcceptAll

Selection.Font.Name = ActiveDocument.Styles(wdStyleNormal).Font.Name

Selection.Font.Size = ActiveDocument.Styles(wdStyleNormal).Font.Size

Selection.Cut

Selection.Start = myStart

Selection.End = wasEnd

Dim cmt As Comment

Set cmt = Selection.Comments.Add(Range:=Selection.Range)

Selection.Paste

ActiveDocument.TrackRevisions = myTrack

With ActiveWindow.View

.ShowRevisionsAndComments = trackOnOffState

.ShowComments = showCmnts

.ShowInsertionsAndDeletions = showInsDels

.ShowFormatChanges = showFormats

End With

Else

cmntText = attrib2

If addPageNum2 = True Then cmntText = cmntText & \_

"(p. " & pageNum & ") "

If addLineNum2 = True Then cmntText = cmntText & \_

"(line " & lineNum & ") "

Selection.MoveEnd , 1

Set cmt = Selection.Comments.Add(Range:=Selection.Range)

Selection.TypeText cmntText

Selection.HomeKey Unit:=wdLine

Selection.Fields.Unlink

End If

If useCommentPane = False Then

ActiveWindow.ActivePane.Close

Else

Application.ActiveWindow.View.Zoom.Percentage = paneZoom

End If

Selection.MoveLeft , 2

cmt.Edit

End Sub

**Sub CommentAdd2()**

**'** Paul Beverley - Version 01.06.13

' Adds a comment

attrib1 = "PB: "

' attrib2 = "T/S: "

' postText = " is not on the refs list."

postText = " Will the readers know this acronym?"

' postText = " Undefined acronym " & Chrw(8217) & " "

' postText = " – reference?"

addPageNum1 = True

addLineNum1 = True

addPageNum2 = True

addLineNum2 = True

highlightTheText = False

textHighlightColour = wdYellow

colourTheText = False

textColour = wdColorBlue

keepPaneOpen = False

myStart = Selection.Start

wasEnd = Selection.End

Set rng = Selection.Range

rng.Collapse wdCollapseEnd

rng.MoveEnd , 1

pageNum = rng.Information(wdActiveEndAdjustedPageNumber)

lineNum = rng.Information(wdFirstCharacterLineNumber)

If myStart <> wasEnd Then

If Right(Selection, 1) = Chr(32) Then

Selection.MoveEnd wdCharacter, -1

wasEnd = wasEnd - 1

End If

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

' Either highlight it ...

If highlightTheText = True Then Selection.Range.HighlightColorIndex \_

= textHighlightColour

' And/or change the text colour to red

If colourTheText = True Then Selection.Font.Color = textColour

' Now create the comment

Selection.Copy

Selection.Collapse wdCollapseEnd

If addPageNum1 = True Then attrib1 = attrib1 & "(p. " & \_

pageNum & ") "

If addLineNum1 = True Then attrib1 = attrib1 & "(line " & \_

lineNum & ") "

Selection.TypeText Text:=attrib1 & ChrW(8216) & ChrW(8217)

' Move back to between the close and open quotes

Selection.MoveLeft 1

' Paste in a copy of the selected text

Selection.Paste

' Move back past the close quote

Selection.MoveRight Count:=1

If postText > "" Then

Selection.TypeText Text:=postText

Else

Selection.TypeText Text:=" " & ChrW(8211) & " "

End If

Selection.Start = wasEnd

Selection.Range.Revisions.AcceptAll

' If wanted, unhighlight the text clear

If removeHighlight = True Then Selection.Range.HighlightColorIndex = wdAuto

Selection.Cut

If Asc(Selection) = 13 Then Selection.MoveEnd 1: Selection.Delete

Selection.Comments.Add Range:=Selection.Range

Selection.Paste

ActiveDocument.TrackRevisions = myTrack

Else

cmntText = attrib2

If addPageNum2 = True Then cmntText = cmntText & \_

"(p. " & pageNum & ") "

If addLineNum2 = True Then cmntText = cmntText & \_

"(line " & lineNum & ") "

Selection.MoveEnd , 1

Selection.Comments.Add Range:=Selection.Range

Selection.TypeText cmntText

Selection.HomeKey Unit:=wdLine

Selection.Fields.Unlink

End If

If keepPaneOpen = False Then ActiveWindow.ActivePane.Close

End Sub

**Sub CommentAdder()**

' Paul Beverley - Version 23.08.22

' Adds a comment off a menu

useCommentPane = False

paneZoom = 240

addItalic = True

addbold = True

' myDefaultSelect = "sentence"

myDefaultSelect = "word"

' useDoubleQuotes = False

useDoubleQuotes = True

addJustPageNum = False

addPageAndLineNum = False

maxLen = 100

cm = "Start\"

cm = cm & ",Dummy\"

cm = cm & "a '[]' is something I’ve /got/ and \*/you’d/\* like, 'yes'?\"

cm = cm & "l '[]' is not in the references list.\"

cm = cm & "L AU: '[]' is not in the references list. (But '[]|', is.)\"

cm = cm & "c '[]' - Not cited in the text.\"

cm = cm & "h '[]' - Have I caught the \*intended\* meaning? /Well?!/\"

cm = cm & "r '[]' - Will the /readers/ know what this means? If so, fine.\"

cm = cm & "R AU: '[]' - Will readers know what '|' refers to? If so, that's fine.\"

cm = cm & "s '[]' - Sorry, but I can't work out the intended meaning here. Is it something like '[]|'?\"

maxLenPrompt = 100

If Selection.Start = Selection.End Then

If InStr(myDefaultSelect, "nten") > 0 Then

Selection.Expand wdSentence

Else

Selection.Expand wdWord

End If

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

pageNum = Selection.Information(wdActiveEndAdjustedPageNumber)

lineNum = Selection.Information(wdFirstCharacterLineNumber)

cm = Replace(cm, " - ", " " & ChrW(8211) & " ")

If useDoubleQuotes = True Then

cm = Replace(cm, " '", " " & ChrW(8220))

cm = Replace(cm, ",'", "," & ChrW(8220))

cm = Replace(cm, "' ", ChrW(8221) & " ")

cm = Replace(cm, "'", ChrW(8221))

cm = Replace(cm, ChrW(8221) & "r", ChrW(8217) & "r")

cm = Replace(cm, ChrW(8221) & "m", ChrW(8217) & "m")

cm = Replace(cm, ChrW(8221) & "t", ChrW(8217) & "t")

cm = Replace(cm, ChrW(8221) & "s", ChrW(8217) & "s")

Else

cm = Replace(cm, " '", " " & ChrW(8216))

cm = Replace(cm, ",'", "," & ChrW(8216))

cm = Replace(cm, "'", ChrW(8217))

End If

cmt = Split(cm, "\")

indx = ""

ns = UBound(cmt) - 1

For i = 1 To ns

If InStr(cmt(i), "Dummy") = 0 Then

indx = indx & Left(cmt(i), 1)

myBit = cmt(i)

If Len(myBit) > maxLenPrompt Then

myBit = Left(myBit, maxLenPrompt)

Do

myBit = Left(myBit, Len(myBit) - 1)

Loop Until Right(myBit, 1) = " "

myBit = myBit & ChrW(8230)

End If

myPrompt = myPrompt & myBit & vbCr

cmt(i) = Mid(cmt(i), 3)

DoEvents

Else

indx = indx & "\_"

End If

Next i

indx = indx

myPrompt = Replace(myPrompt, "[]", "") \_

& vbCr & "Comment?"

'myPrompt = Replace(myPrompt, "/", "")

'myPrompt = Replace(myPrompt, "\*", "")

Do

myCode = InputBox(myPrompt, "CommentAdder")

n = InStr(indx, myCode)

If n = 0 Then Beep

Loop Until n > 0

If myCode = "" Then Exit Sub

If Selection.Start = Selection.End Then Selection.MoveEnd , 1

Selection.Copy

Application.ScreenUpdating = False

On Error GoTo ReportIt

Set myComment = Selection.Comments.Add(Range:=Selection.Range)

myComment.Edit

myFontSize = myComment.Range.Font.Size

myStart = Selection.Start

myPLtext = ""

If addPageAndLineNum = True Then

myPLtext = "(p. " & pageNum & ", line " & lineNum & ") "

Else

If addJustPageNum = True Then myPLtext = myPLtext & "(p. " & \_

pageNum & ") "

End If

Debug.Print myPLtext

Selection.TypeText Text:=cmt(n)

txtPos = InStr(cmt(n), "[]")

txtPos2 = 0

If txtPos > 0 Then txtPos2 = InStr(Mid(cmt(n), txtPos + 2), "[]")

If txtPos2 > 0 Then

txtPos2 = txtPos2 + txtPos + 1

Selection.Start = myStart + txtPos2 - 1

Selection.End = myStart + txtPos2 + 1

Selection.Paste

End If

If txtPos > 0 Then

Selection.Start = myStart + txtPos - 1

Selection.End = myStart + txtPos + 1

Selection.Paste

End If

Application.ScreenUpdating = True

Set rng = Selection.Range.Duplicate

If highlightColour > 0 Or fontColour > 0 Or isUL Then

Selection.Collapse wdCollapseStart

Selection.MoveEnd , 1

End If

Selection.Expand wdParagraph

If Asc(Selection) = 5 Then

Selection.MoveStart , 1

End If

Selection.Collapse wdCollapseStart

Selection.TypeText Text:=myPLtext

Selection.Expand wdParagraph

Selection.Range.Font.Size = myFontSize

Set rng = Selection.Range.Duplicate

' Add italic

If addItalic = True Then

italicPos = InStr(Selection, "/")

Do While italicPos > 0

If italicPos > 0 Then

Selection.MoveStart , italicPos - 1

Selection.End = Selection.Start + 1

Selection.Delete

rng.Select

Selection.MoveStart , italicPos - 1

italicEndPos = InStr(Selection, "/")

If italicPos > 0 Then

Selection.End = Selection.Start + italicEndPos

Selection.Font.Italic = True

Selection.Collapse wdCollapseEnd

Selection.MoveStart , -1

Selection.Delete

End If

End If

rng.Select

italicPos = InStr(Selection, "/")

Loop

End If

' Add bold

If addbold = True Then

rng.Select

boldPos = InStr(Selection, "\*")

Do While boldPos > 0

If boldPos > 0 Then

Selection.MoveStart , boldPos - 1

Selection.End = Selection.Start + 1

Selection.Delete

rng.Select

Selection.MoveStart , boldPos - 1

boldEndPos = InStr(Selection, "\*")

If boldPos > 0 Then

Selection.End = Selection.Start + boldEndPos

Selection.Font.Bold = True

Selection.Collapse wdCollapseEnd

Selection.MoveStart , -1

Selection.Delete

End If

End If

rng.Select

boldPos = InStr(Selection, "\*")

Loop

End If

' Now position the cursor

rng.Select

cursorPos = InStr(Selection, "|")

If cursorPos > 0 Then

Selection.Collapse wdCollapseEnd

i = 0

Do

Selection.MoveLeft , 1

i = i + 1

DoEvents

Loop Until Selection = "|" Or i > maxLen

If i > maxLen Then

MsgBox ("You might need to increase the value of ""maxLen"" in the macro.")

Exit Sub

End If

Selection.MoveEnd , 1

Selection.Delete

Else

Selection.MoveEnd , -1

Selection.Collapse wdCollapseEnd

End If

If useCommentPane = False Then

ActiveWindow.ActivePane.Close

Else

Application.ActiveWindow.View.Zoom.Percentage = paneZoom

End If

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub CommentAddMenu()**

**'** Paul Beverley - Version 26.02.24

' Adds a comment off a menu

useCommentPane = False

paneZoom = 240

myPrefix = "PB: "

myPrefix = ""

addPageNum = False

addLineNum = False

listName = "zzSwitchList"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff\"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff/"

defaultList = myFolder & listName

Set textDoc = ActiveDocument

If Selection.Start = Selection.End Then

Selection.Expand wdSentence

If Right(Selection, 4) = "al. " Or Right(Selection, 5) = "al., " Then

Selection.MoveRight Unit:=wdSentence, Count:=1, Extend:=wdExtend

End If

Do While InStr(" " & vbCr, Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

On Error Resume Next

Set thisDoc = ActiveDocument.ActiveWindow

dirName = ActiveDocument.Path

' Go and look for the list file

gottaList = False

For i = 1 To Application.Windows.Count

If InStr(Application.Windows(i).Document.Name, \_

listName) > 0 Then

Set listDoc = Application.Windows(i).Document

gottaList = True

End If

Next i

If gottaList = False Then

Documents.Open dirName & "\" & listName

If Err.Number = 5174 Then

Err.Clear

Documents.Open defaultList

If Err.Number = 5174 Then

Err.Clear

Documents.Open defaultList & ".docx"

End If

If Err.Number > 0 And Err.Number <> 438 Then GoTo ReportIt

End If

End If

On Error GoTo 0

Set rng = listDoc.Content

rngEnd = rng.End

' Find first comment line

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[[My comments]]"

.Wrap = wdFindContinue

.Forward = True

.MatchWildcards = False

.MatchCase = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If rng.Find.Found = 0 Then

myPrompt = "Your comments text must start with: ""[[My comments]]"""

Beep

myResponse = MsgBox(myPrompt, \_

vbQuestion + vbOKCancel, "CommentAddMenu")

If myResponse <> vbYes Then Exit Sub

End If

' Find first item in list of comments

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

listStart = rng.Start

rng.End = rngEnd

' Construct myFullPrompt

myFullPrompt = ""

allCodes = ""

myLine = rng.Paragraphs(1)

textLeft = rng.Text

Do

sqPos = InStr(myLine, "] ")

If sqPos = 0 And LCase(textLeft) <> UCase(textLeft) Then

rng.Paragraphs(1).Select

MsgBox "Please check the formatting of this line."

Beep

Exit Sub

End If

curlyPos1 = InStr(myLine, "{")

curlyPos2 = InStr(myLine, "}")

If curlyPos1 \* curlyPos2 = 0 Then

rng.Paragraphs(1).Select

MsgBox "This line needs a prompt text inside {} brackets."

Beep

Exit Sub

End If

myCode = Mid(myLine, 2, sqPos - 2)

myPrompt = Mid(myLine, curlyPos1 + 1, curlyPos2 - curlyPos1 - 1)

If InStr(allCodes, "[" & myCode & "]") = 0 Then

myFullPrompt = myFullPrompt & myCode & " = " & myPrompt & vbCr

End If

allCodes = allCodes & Left(myLine, sqPos)

rng.Start = rng.Paragraphs(1).Range.End

myLine = rng.Text

DoEvents

textLeft = rng.Text

Loop Until LCase(textLeft) = UCase(textLeft)

myFullPrompt = Left(myFullPrompt, Len(myFullPrompt) - 1)

myResponse = InputBox(myFullPrompt, "Comment Add Menu")

If myResponse = "" Then

Beep

Exit Sub

End If

rng.Start = listStart

'Now pick up the chosen comment

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & myResponse & "]"

.Wrap = wdFindContinue

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If rng.Find.Found = False Then

Beep

Exit Sub

End If

' Select required comment

rng.Expand wdParagraph

myLine = rng.Text

sqPos = InStr(myLine, "] ")

rng.MoveStart , sqPos + 1

myLine = rng.Text

curlyPos1 = InStr(myLine, "{")

rng.End = rng.Start + curlyPos1 - 2

cursorPos = InStr(rng, "][")

lenCopy = Len(rng)

If lenCopy > 1 Then rng.Copy

' Find page and line number within text

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

myStart = Selection.Start

wasEnd = Selection.End

Set rng = Selection.Range.Duplicate

rng.End = myStart + 1

pageNum = rng.Information(wdActiveEndAdjustedPageNumber)

lineNum = rng.Information(wdFirstCharacterLineNumber)

' Now create the comment

Dim cmt As Comment

Set rngText = Selection.Range.Duplicate

Set cmt = Selection.Comments.Add(Range:=Selection.Range)

If useCommentPane = False Then

ActiveWindow.ActivePane.Close

cmt.Edit

Else

Application.ActiveWindow.View.Zoom.Percentage = paneZoom

End If

myPLtext = ""

If addPageNum = True Then myPLtext = myPLtext & "(p. " & \_

pageNum

If addLineNum = True Then

myPLtext = myPLtext & ", line " & lineNum & ") "

Else

myPLtext = myPLtext & ") "

End If

If Len(myPLtext) > 4 Then myPrefix = myPrefix & myPLtext

If Len(myPrefix) > 0 Then Selection.TypeText myPrefix

For i = 1 To 200

DoEvents

Next

If lenCopy > 1 Then Selection.Paste

For i = 1 To 200

DoEvents

Next

' Replace the <> with the quote

cmt.Range.Select

rngText.Copy

quotePos = InStr(Selection, "<>")

Do While quotePos > 0

Selection.Start = Selection.Start + quotePos - 1

Selection.End = Selection.Start + 2

Selection.Delete

Selection.Paste

cmt.Range.Select

quotePos = InStr(Selection, "<>")

Loop

Application.ScreenUpdating = True

ActiveDocument.TrackRevisions = myTrack

cmt.Range.Select

cursorPos = InStr(Selection, "][")

If cursorPos > 0 Then

myLen = cmt.Range.Characters.Count

Selection.MoveStart , cursorPos - 1

Selection.End = Selection.Start + 2

Selection.Delete

End If

If cursorPos > 0 Then

cmt.Edit

Selection.MoveLeft , (myLen - cursorPos - 1)

Else

If lenCopy > 0 Then cmt.Edit

Selection.TypeText " "

End If

Exit Sub

ReportIt:

If Err.Number = 5174 Then

MsgBox ("Couldn't find file: " & listName)

Else

On Error GoTo 0

End If

End Sub

**Sub CommentAddSimple()**

**'** Paul Beverley - Version 26.02.24

' Adds a new comment with a standard text item added

useCommentPane = True

paneZoom = 200

' myText = "AQ: "

myText = ""

Dim cmt As Comment

Set cmt = Selection.Comments.Add(Range:=Selection.Range)

cmt.Edit

Selection.TypeText Text:=myText

Selection.MoveLeft , 1

Selection.MoveRight , 1

If useCommentPane = False Then

ActiveWindow.ActivePane.Close

Else

Application.ActiveWindow.View.Zoom.Percentage = paneZoom

End If

End Sub

**Sub CommentsPane()**

' Paul Beverley - Version 11.05.21

' Opens and closes the Comments pane

pCentPane = 30

paneZoom = 240

numCmts = ActiveDocument.Comments.Count

nowPane = ActiveDocument.ActiveWindow.View.SplitSpecial

On Error GoTo ReportIt

If numCmts = 0 Then

Beep

MsgBox "No comments in this file."

Exit Sub

End If

If nowPane <> 15 Then

ActiveDocument.ActiveWindow.View.SplitSpecial = wdPaneComments

ActiveWindow.SplitVertical = 100 - pCentPane

Application.ActiveWindow.View.Zoom.Percentage = paneZoom

Else

ActiveDocument.ActiveWindow.View.SplitSpecial = wdPaneNone

End If

CommandBars("Comments").Visible = True

Exit Sub

ReportIt:

errNum = Err.Number

Beep

Debug.Print "Error number = " & errNum

If Err.Number = 4605 Then

MsgBox ("Place the cursor in the text first." \_

& vbCr & vbCr & "Then rerun CommentsPane macro.")

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub CommentsPaneOLD()**

' Paul Beverley - Version 08.11.17

' Open the comments pane

If (ActiveDocument.Comments.Count > 0) Then

ActiveDocument.ActiveWindow.View.SplitSpecial = wdPaneComments

Else

Beep

MsgBox "No comments in this file."

End If

End Sub

**Sub CommentCopier()**

**'** Paul Beverley - Version 15.01.24

' Creates an author query list

auInits = ""

auInits = "AU"

myInits = "PB"

queriesFileOnTop = False

createContextFile = True

showTrackChanges = True

multiSpace = 4

addPageNum = True

addLineNum = False

CR2 = vbCr & vbCr

Set chapterText = ActiveDocument

numComments = ActiveDocument.Comments.Count

If numComments > 0 Then

Set rngCmnts = ActiveDocument.StoryRanges(wdCommentsStory)

End If

Documents.Add

Set queriesFile = ActiveDocument

Set rng = ActiveDocument.Content

For i = 1 To numComments

rng.Text = chapterText.Comments(i).Initial & Trim(Str(i))

rng.Font.Bold = True

rng.InsertAfter Text:=" "

rng.Collapse wdCollapseEnd

rng.FormattedText = chapterText.Comments(i).Range.FormattedText

rng.Collapse wdCollapseEnd

rng.Text = CR2 & "Answer [" & auInits & Trim(Str(i)) & \_

"]: " & CR2 & CR2

rng.Font.Color = wdColorRed

rng.Collapse wdCollapseEnd

Next i

Selection.WholeStory

Selection.Style = wdStyleNormal

' Remove hidden page references

Selection.Fields.Unlink

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Author queries Chapter " & vbCr & vbCr

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

Selection.MoveLeft , 2

' Now create the context file

sp = ""

If createContextFile = True Then

Documents.Add

Set contextFile = ActiveDocument

For i = 1 To multiSpace

sp = sp & vbCr

Next i

chapterText.Activate

Set rng = ActiveDocument.Content

contextFile.Activate

Set rngDummy = ActiveDocument.GoTo(What:=wdGoToBookmark, Name:="\EndOfDoc")

For Each myPar In rng.Paragraphs

If myPar.Range.Comments.Count > 0 Then

Set testChar = myPar.Range.Characters(1)

pageNum = testChar.Information(wdActiveEndPageNumber)

lineNum = testChar.Information(wdFirstCharacterLineNumber)

myPar.Range.Copy

myPLtext = ""

If addPageNum = True Then myPLtext = "(p. " & pageNum

If addLineNum = True Then

myPLtext = myPLtext & ", line " & lineNum & ")"

Else

myPLtext = myPLtext & ") "

End If

If Len(myPLtext) > 4 Then Selection.TypeText myPLtext & vbCr

Selection.Paste

Selection.Collapse wdCollapseEnd

Selection.TypeText sp

End If

Next myPar

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Author queries CONTEXT Chapter " & vbCr & vbCr

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

Selection.MoveLeft , 2

End If

If queriesFileOnTop Then queriesFile.Activate

End Sub

**Sub CopyCommentsAndSave()**

**'** Paul Beverley - Version 10.04.24

' Create a list of all the comments

DocName = ActiveDocument.FullName

NewDocName = Replace(DocName, ".doc", "\_Cmnts.doc")

If ActiveDocument.Comments.Count >= 1 Then

ActiveDocument.StoryRanges(wdCommentsStory).Copy

End If

Documents.Add

Selection.Paste

ActiveDocument.SaveAs FileName:=NewDocName

End Sub

**Sub CommentListNumbered()**

' Paul Beverley - Version 16.01.21

' List all comments in file with index numbers

addAnswerLine = True

deleteTScomments = True

TScode = "T/S:"

If ActiveDocument.Comments.Count = 0 Then

MsgBox "No comments in this file."

Exit Sub

End If

Dim cmnt As Word.Comment

totCmnts = ActiveDocument.Comments.Count

ReDim cmText(totCmnts) As String

ReDim cmInits(totCmnts) As String

CR = vbCr: CR2 = CR & CR

' Collect comments, including formatting

If ActiveDocument.Comments.Count >= 1 Then

ActiveDocument.StoryRanges(wdCommentsStory).Copy

End If

' Collect initials and text of comments

For i = 1 To totCmnts

Set cmnt = ActiveDocument.Comments(i)

cmInits(i) = cmnt.Initial

cmText(i) = cmnt.Range

Next i

Documents.Add

Selection.Paste

Set rng = ActiveDocument.Content

For i = 1 To totCmnts

lenCmnt = Len(cmText(i))

If lenCmnt > 10 Then

thisText = Left(cmText(i), 10)

Else

thisText = cmText(i)

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = thisText

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

rng.MoveEndUntil cset:=Chr(13), Count:=wdForward

If i = 1 Then

rng.InsertBefore Text:="[" & cmInits(i) & Trim(Str(i)) & "]: "

Else

rng.InsertBefore Text:=CR & "[" & cmInits(i) & Trim(Str(i)) & "]: "

End If

rng.Collapse wdCollapseEnd

Next i

Selection.EndKey Unit:=wdStory

Selection.TypeText CR & "["

If addAnswerLine = True Then

Set rng = ActiveDocument.Content

rng.Font.Color = wdColorBlue

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p["

.Replacement.Text = "^pAnswer: ^p^p["

.Replacement.Font.Color = wdColorBlack

.Execute Replace:=wdReplaceAll

End With

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13[\*T/S\*^13"

.Replacement.Text = "^pAnswer: ^p^p["

.Replacement.Font.Color = wdColorBlack

.Execute Replace:=wdReplaceAll

End With

If deleteTScomments = True Then

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = TScode

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myCount = 0

Do While Selection.Find.Found = True

With Selection

.Expand wdParagraph

If addAnswerLine = True Then

.MoveDown Unit:=wdLine, Count:=3, Extend:=wdExtend

Else

.MoveDown Unit:=wdLine, Count:=1, Extend:=wdExtend

End If

.Delete

.Find.Execute

End With

Loop

End If

Set rng = ActiveDocument.Content

rng.Font.Size = ActiveDocument.Styles(wdStyleNormal).Font.Size

Selection.EndKey Unit:=wdStory

Selection.MoveStart , -1

Selection.Delete

' Delete all page number fields

For Each fld In ActiveDocument.Fields

If fld.Type = 33 Then fld.Delete

Next fld

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Author queries " & ChrW(8211) & " Chapter " & vbCr & vbCr

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

ActiveDocument.Paragraphs(1).Range.Font.ColorIndex = wdColorBlack

Selection.MoveLeft , 2

End Sub

**Sub CommentsCollectTabulated()**

' Paul Beverley - Version 21.12.23

' Collects comments into a table

sortOnColumn = 0

splitComment = True

doLandscape = True

maxHeading = "Heading 1"

s = ChrW(160)

s = s & s & s & s & s

myspaces = s & s & s & s & s & s & s & s

' If Word throws up and error 4605 about pasting

' increase the delay value to, say, 1000

myDelay = 500

myScreenOff = True

totCmnts = ActiveDocument.Comments.Count

Selection.HomeKey Unit:=wdStory

Selection.EndKey Unit:=wdStory

If totCmnts = 0 Then

MsgBox "There are no comments in this file!"

Exit Sub

End If

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

ReDim pageNo(totCmnts) As String

ReDim lineNo(totCmnts) As String

For i = 1 To totCmnts

Selection.GoTo What:=wdGoToComment, count:=i

pageNo(i) = Trim(Str(Selection.Range.Information(wdActiveEndAdjustedPageNumber)))

lineNo(i) = Trim(Str(Selection.Range.Information(wdFirstCharacterLineNumber)))

For j = 1 To myDelay

DoEvents

Next j

Next i

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.Copy

Documents.Add

For i = 1 To myDelay

DoEvents

Next i

Selection.Paste

totCmnts = ActiveDocument.Comments.Count

Dim cmnt As Word.Comment

ReDim lineNum(totCmnts) As Integer

ReDim pageNum(totCmnts) As Integer

maxHeadNo = Val(Right(maxHeading, 1))

hdStart = Left(maxHeading, InStr(maxHeading, " ") - 1)

For i = 1 To totCmnts

Selection.GoTo What:=wdGoToComment, count:=i

pn = pageNo(i)

ln = lineNo(i)

Set cmnt = ActiveDocument.Comments(i)

inits = cmnt.Initial

scp = cmnt.Scope

itemNo = Trim(Str(i))

If splitComment = True And InStr(cmnt.Range, "|") = False Then

cmnt.Range.InsertBefore Text:="|"

End If

' Selection.GoTo what:=wdGoToComment, Count:=i

Set rng = ActiveDocument.Range(0, Selection.End)

paraNum = rng.Paragraphs.Count

For j = paraNum - 1 To 1 Step -1

Debug.Print ActiveDocument.Paragraphs(j).Range.Text

myStyle = ActiveDocument.Paragraphs(j).Range.Style

If InStr(myStyle, hdStart) > 0 Then

hdLevel = Val(Right(myStyle, 1))

If hdLevel < maxHeadNo + 1 Then

hd = ActiveDocument.Paragraphs(j).Range.Text

Exit For

End If

End If

DoEvents

' Debug.Print "Reading: ", pageNo(i)

Next j

' BEFORE TEXT HERE <<<<<<<<<<<<<<<<<<<<<<<<

' Full version

' cmnt.Range.InsertBefore Text:=inits & itemNo & "|" & "p." & pn & \_

" l." & ln & "|" & scp & "|"

' Paul's own short version

cmnt.Range.InsertBefore Text:="yNLyp." & pn & "|" & scp

' cmnt.range.InsertBefore Text:="yNLyp." & pn & "|" & scp

' AFTER TEXT HERE <<<<<<<<<<<<<<<<<<<<<<<<<

cmnt.Range.InsertAfter Text:=""

Next i

ActiveDocument.StoryRanges(wdCommentsStory).Copy

' Replace whole text with just the (augmented) comments

Selection.WholeStory

For i = 1 To myDelay

DoEvents

Next i

Selection.Paste

Selection.WholeStory

' Remove hidden page references

Selection.Fields.Unlink

If doLandscape = True Then Selection.PageSetup.Orientation = wdOrientLandscape

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.Text = "^p"

.Replacement.Text = "zCRz"

.Wrap = wdFindContinue

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

For i = 1 To 2

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

' Go and find the first occurrence

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zCRz"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While rng.Find.Found = True

rng2.Start = rng.End

rng2.End = rng.End + 5

If rng2 = "yNLy" Then rng.Delete

rng.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

rng.Find.Execute

Loop

Next i

' Remove the very first new line marker

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "yNLy"

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

' Change other new line markers into new lines

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "yNLy"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Replacement.Highlight = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.EndKey Unit:=wdStory

Selection.MoveStart , -4

If Selection = "zCRz" Then Selection.Delete

Selection.EndKey Unit:=wdStory

Selection.MoveStart , -4

If Selection = "zCRz" Then Selection.Delete

Selection.HomeKey Unit:=wdStory

' HEADING LINE HERE<<<<<<<<<<<<<<<<<<<<<<<<

Selection.TypeText Text:="|Context|Comment/query|Author" & ChrW(160) & \_

"response" & myspaces & vbCr

ActiveDocument.Paragraphs(1).Range.Bold = True

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "|"

.Wrap = wdFindContinue

.Replacement.Text = "^t"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Author queries " & ChrW(8211) & \_

" Chapter " & vbCr & vbCr

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

ActiveDocument.Paragraphs(3).Range.Select

Selection.Collapse wdCollapseEnd

Selection.End = ActiveDocument.Range.End

If sortOnColumn > 0 Then Selection.Sort ExcludeHeader:=False, \_

FieldNumber:="Column " & Trim(Str(sortOnColumn)), \_

SortFieldType:=wdSortFieldAlphanumeric, \_

SortOrder:=wdSortOrderAscending, \_

Separator:=wdSortSeparateByTabs, SortColumn:=False, \_

CaseSensitive:=True

ActiveDocument.Paragraphs(2).Range.Select

Selection.Collapse wdCollapseEnd

Selection.End = ActiveDocument.Range.End

Selection.ConvertToTable Separator:=wdSeparateByTabs

Selection.Tables(1).AutoFitBehavior (wdAutoFitContent)

' Restore the new lines that are inside the comments

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zCRz"

.Replacement.Text = "^p"

.Wrap = wdFindContinue

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Beep

ActiveDocument.TrackRevisions = myTrack

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub CommentContextCopier()**

' Paul Beverley - Version 20.01.18

' Copies paragraphs containing comments into a new file

multiSpace = 4

showTrackChanges = True

myTrack = ActiveDocument.TrackRevisions

If showTrackChanges = True Then \_

ActiveDocument.TrackRevisions = False

Set chapterText = ActiveDocument

For i = 1 To multiSpace

sp = sp & vbCr

Next i

Set rng = ActiveDocument.Content

Documents.Add

Set contextDoc = ActiveDocument

For Each myPara In rng.Paragraphs

If myPara.range.Comments.Count > 0 Then

myPara.range.Copy

Selection.Paste

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=sp

End If

Next myPara

chapterText.Activate

ActiveDocument.TrackRevisions = myTrack

contextDoc.Activate

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Comments context " & \_

ChrW(8211) & " Chapter " & vbCr & vbCr

ActiveDocument.Paragraphs(1).range.Style = "Heading 2"

Selection.MoveLeft , 2

End Sub

**Sub MultiFileComment()**

' Paul Beverley - Version 13.12.17

' List all comments in a set of files

addAnswers = True

myFileTitleSize = 14

CR = vbCr

CR2 = CR & CR

Dim cmnt As word.Comment

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Collect comments from ALL the files in" & \_

" directory:" & dirName & " ?", vbQuestion + vbYesNoCancel, \_

"Multifile Comment Collection")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Collect comments from the files listed here?", \_

vbQuestion + vbYesNoCancel, "Multifile Comment Collection")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

Documents.Add

Set myList = ActiveDocument

For j = 1 To numFiles

thisFile = myFolder & myDelimiter & allMyFiles(j)

Set myDoc = Application.Documents.Open(FileName:=thisFile)

myDocName = myDoc.Name

myFileType = Mid(myDocName, InStr(myDocName, "."))

myDocName = Replace(myDocName, myFileType, "")

' Collect all the comments

totCmnts = ActiveDocument.Comments.Count

If totCmnts > 0 Then

ReDim cmText(totCmnts) As String

For i = 1 To totCmnts

Set cmnt = ActiveDocument.Comments(i)

inits = cmnt.Initial

cmText(i) = cmnt.range

If Left(cmText(i), Len(inits)) <> inits Then \_

cmText(i) = inits & ": " & cmText(i)

If addAnswers = True Then cmText(i) = cmText(i) & CR2 \_

& "Answer: " & CR

cmText(i) = cmText(i) & CR

Next i

cmText(i - 1) = cmText(i - 1) & CR

Else

totCmnts = 1

ReDim cmText(totCmnts) As String

cmText(1) = "zxzx"

End If

' Type the comments into the list

myList.Activate

' First type the filename as a heading

Selection.TypeText Text:=myDocName & vbCr

Selection.MoveUp Unit:=wdLine, Count:=1, Extend:=wdExtend

Selection.Font.Bold = True

Selection.Font.Size = myFileTitleSize

Selection.EndKey Unit:=wdStory

Selection.TypeText CR

' Then type each of the comments

For i = 1 To totCmnts

startHere = Selection.Start

Selection.TypeText Text:="[" & cmText(i) & CR

endHere = Selection.Start

nowLen = Selection.Start - startHere

Selection.End = startHere + InStr(cmText(i), ":")

extraBit = Trim(Str(i)) & "]"

Selection.TypeText Text:=extraBit

Selection.Start = endHere + Len(extraBit)

Selection.End = Selection.Start

Next i

myDoc.Close SaveChanges:=wdDoNotSaveChanges

Next j

' Colour the Answer lines

myList.Activate

If addAnswers = True Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Answer: ^p"

.Replacement.Text = "^&"

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Replacement.Font.Color = wdColorRed

.Execute Replace:=wdReplaceAll

End With

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "1][zxzx"

.Replacement.Text = "No comments^p^p"

.Execute Replace:=wdReplaceAll

End With

' Remove hidden page references

Set rng = ActiveDocument.Content

rng.Fields.Unlink

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub CommentNumbering()**

' Paul Beverley - Version 20.05.19

' Add or remove fixed comment numbers

myInits = "AM"

myTest = ActiveDocument.Comments(1).range

' If the first comment has a [ ] in it...

Set rng = ActiveDocument.StoryRanges(wdCommentsStory)

If InStr(myTest, "[" & myInits) > 0 Then

' Delete a [ ] item from each comment

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[" & myInits & "[0-9]@\]: "

.Wrap = wdFindContinue

.Replacement.Text = ""

.Wrap = wdFindContinue

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

.Text = myInits & "[0-9]@\[: "

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Beep

Else

' Add initials + index number to each comment

For i = 1 To ActiveDocument.Comments.Count

myInits = "[" & ActiveDocument.Comments(i).Initial & Trim(Str(i)) & "]: "

ActiveDocument.Comments(i).range.InsertBefore myInits

Next i

End If

End Sub

**Sub AddCommentMarkersInText()**

**'** Paul Beverley - Version 30.12.11

' Add comment initials and numbers to text

For k = 1 To ActiveDocument.Comments.Count

myInits = "[" & ActiveDocument.Comments(k).Initial & Trim(Str(k)) & "]"

ActiveDocument.Comments(k).Reference.InsertAfter myInits

Set rng = ActiveDocument.Comments(k).Reference

rng.End = rng.End + Len(myInits)

rng.HighlightColorIndex = wdBrightGreen

Next k

End Sub

**Sub CommentInitialReplaceAll()**

' Paul Beverley - Version 14.08.19

' Changes all comment initials

newInitial = "PEB"

For i = 1 To ActiveDocument.Comments.Count

ActiveDocument.Comments(i).Initial = newInitial

Next i

End Sub

**Sub CommentInitialFandR()**

' Paul Beverley - Version 14.08.19

' Finds and replaces comment initials

oldInitial = "SPQR"

newInitial = "PEB"

For i = 1 To ActiveDocument.Comments.Count

If ActiveDocument.Comments(i).Initial = oldInitial Then

ActiveDocument.Comments(i).Initial = newInitial

End If

Next i

End Sub

**Sub DeleteAllComments()**

**'** Paul Beverley - Version 21.02.12

' Delete all comments

numberCmnts = ActiveDocument.Comments.Count

If numberCmnts > 0 Then ActiveDocument.DeleteAllComments

MsgBox ("Comments deleted: " & Str(numberCmnts))

End Sub

**Sub CommentPicker()**

' Paul Beverley - Version 09.11.13

' Copy a comment out of a list of comments

myCode = InputBox("WhichComment number?", "CommentPicker")

For Each pa In ActiveDocument.Paragraphs

myText = pa.Range.Text

If Len(myText) < 4 And Asc(myText) = Asc(myCode) Then

pa.Range.Select

Selection.Collapse wdCollapseEnd

Selection.Paragraphs(1).Range.Select

Selection.MoveEnd , -1

Selection.Copy

Exit Sub

End If

Next

MsgBox ("Can't find comment >> " & myCode & " <<")

End Sub

**Sub CommentPickerInserter()**

' Paul Beverley - Version 14.11.13

' Copy a comment from a list and paste into text

commentDoc = "CommentList"

Set mainDoc = ActiveDocument

Documents(commentDoc).Activate

myCode = InputBox("WhichComment number?", "CommentPicker")

For Each pa In ActiveDocument.Paragraphs

myText = pa.Range.Text

If Len(myText) < 4 And Asc(myText) = Asc(myCode) Then

pa.Range.Select

Selection.Collapse wdCollapseEnd

Selection.Paragraphs(1).Range.Select

Selection.MoveEnd , -1

Selection.Copy

theText = Selection.Text

mainDoc.Activate

Selection.Collapse wdCollapseEnd

theStart = Selection.Start

Selection.Paste

cursorPosn = InStr(theText, "><")

If cursorPosn > 0 Then

Selection.Start = theStart + cursorPosn - 1

Selection.End = theStart + cursorPosn + 1

Selection.Delete

End If

Exit Sub

End If

Next

MsgBox ("Can't find comment >> " & myCode & " <<")

End Sub

**Sub CommentChangeScope()**

' Paul Beverley - Version 04.03.20

' Reduces or extends the scope of a comment

Set rng = Selection.range.Duplicate

Set rng1 = Selection.range.Duplicate

myStart = rng.Start

myEnd = rng.End

rng.Expand wdParagraph

gottaComment = False

For i = 1 To rng.Comments.Count

Set cmt = rng.Comments(i)

cmtStart = cmt.Scope.Start

cmtEnd = cmt.Scope.End

If (myStart > cmtStart And myStart < cmtEnd) \_

Or (myEnd > cmtStart And myEnd < cmtEnd) \_

Or (cmtStart > myStart And cmtStart < myEnd) \_

Or (cmtEnd > myStart And cmtEnd < myEnd) Then

gottaComment = True

Exit For

End If

Next i

If gottaComment = True Then

cmt.Edit

cmt.range.Select

Selection.Copy

cmt.Scope.Select

cmt.Delete

Set cmt = Selection.Comments.Add(range:=rng1)

cmt.Edit

Selection.Paste

cmt.Scope.Select

Else

Beep

End If

End Sub

**Sub CommentNameChanger()**

**'** Paul Beverley - Version 26.02.24

' Changes the name attached to comments

newName = "Referee"

nameToChange = "Paul Beverley"

' To change ALL names, use nameToChange = ""

nameToChange = ""

nameNow = Application.UserName

Application.UserName = newName

Dim newCmt As Comment

For i = 1 To ActiveDocument.Comments.Count

Set cmt = ActiveDocument.Comments(i)

If cmt.Contact = nameToChange Or nameToChange = "" Then

Set rng = cmt.Scope

cmt.Range.Copy

cmt.Delete

Set newCmt = rng.Comments.Add(Range:=rng)

ActiveDocument.Comments(i).Edit

Selection.Paste

DoEvents

End If

Next i

Application.UserName = nameNow

ActiveDocument.ActiveWindow.View.SplitSpecial = wdPaneNone

Beep

End Sub

**Sub CommentTextNormalise()**

' Paul Beverley - Version 21.01.15

' Get rid of "funny effects" in the comment boxes

With ActiveDocument.Styles("Comment Text")

.LanguageID = wdEnglishUK

.AutomaticallyUpdate = False

.BaseStyle = "Normal"

.NextParagraphStyle = "Comment Text"

.Font.Name = "Times New Roman"

.Font.Size = 10

End With

End Sub

**Sub TrackOnOff()**

**'** Paul Beverley - Version 01.01.10

' Toggle track changes on and off

ActiveDocument.TrackRevisions = Not ActiveDocument.TrackRevisions

End Sub

**Sub TrackOnOffAudible()**

**'** Paul Beverley - Version 26.02.11

' Track on/off with audio feedback

' <Ctrl-alt-num-/>

ActiveDocument.TrackRevisions = Not ActiveDocument.TrackRevisions

Beep

myTime = Timer

If ActiveDocument.TrackRevisions = True Then

Do

Loop Until Timer > myTime + 0.2

Beep

End If

End Sub

**Sub ChangeNotTracked()**

**'** Paul Beverley - Version 03.11.11

' Changes (or add) some text, but don't track the change

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If Selection.Start <> Selection.End Then Selection.Delete

newText = InputBox("Change to?")

Selection.TypeText Text:=newText

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub TrackChangeAccept()**

**'** Paul Beverley - Version 23.11.11

' Accepts the track changes on the current line

If Selection.Start = Selection.End Then

Selection.HomeKey Unit:=wdLine

Selection.MoveDown Unit:=wdLine, Count:=1, Extend:=wdExtend

Selection.Range.Revisions.AcceptAll

Selection.Start = Selection.End

Else

Selection.Range.Revisions.AcceptAll

End If

End Sub

**Sub TrackChangeReject()**

**'** Paul Beverley - Version 23.11.11

' Rejects the track changes on the current line

If Selection.Start = Selection.End Then

Selection.HomeKey Unit:=wdLine

Selection.MoveDown Unit:=wdLine, Count:=1, Extend:=wdExtend

Selection.Range.Revisions.RejectAll

Selection.Start = Selection.End

Else

Selection.Range.Revisions.RejectAll

End If

End Sub

**Sub UndoSpecificTrack()**

' Paul Beverley - Version 06.04.16

' Find specific text and undo the track changes on it

myPhrase = "Big Company"

doMatchCase = False

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myPhrase

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While Selection.Find.Found = True

myStart = Selection.Start

myEnd = Selection.End

Selection.MoveLeft , 1

theStart = Selection.Start

Selection.Find.Execute

Selection.MoveRight , 1

Selection.Start = theStart

myCount = myCount + 1

Selection.range.Revisions.RejectAll

Selection.Collapse wdCollapseEnd

Selection.Find.Execute

Loop

MsgBox "Changed: " & myCount

End Sub

**Sub ShowHideAllTracking()**

**'** Paul Beverley - Version 01.01.10

' Toggle showing track changes and comments on and off

ActiveWindow.View.ShowRevisionsAndComments = Not \_

ActiveWindow.View.ShowRevisionsAndComments

End Sub

**Sub ShowHideTracksOnly()**

**'** Paul Beverley - Version 01.01.10

' Toggle showing just track changes on and off

ActiveWindow.View.ShowInsertionsAndDeletions = Not \_

ActiveWindow.View.ShowInsertionsAndDeletions

ActiveWindow.View.ShowFormatChanges = Not \_

ActiveWindow.View.ShowFormatChanges

End Sub

**Sub TrackingShowSwitcher()**

**'** Paul Beverley - Version 23.02.11

' Toggle visibility of track changes

comm = ActiveWindow.View.ShowComments

inDel = ActiveDocument.ActiveWindow.View.ShowInsertionsAndDeletions

If comm = True And inDel = True Then

ActiveDocument.ActiveWindow.View.ShowFormatChanges = False

ActiveDocument.ActiveWindow.View.ShowInsertionsAndDeletions = False

Exit Sub

End If

If inDel = False And comm = True Then

ActiveDocument.ActiveWindow.View.ShowFormatChanges = False

ActiveDocument.ActiveWindow.View.ShowInsertionsAndDeletions = False

ActiveWindow.View.ShowComments = False

Exit Sub

End If

ActiveDocument.ActiveWindow.View.ShowFormatChanges = True

ActiveDocument.ActiveWindow.View.ShowInsertionsAndDeletions = True

ActiveWindow.View.ShowComments = True

End Sub

**Sub TrackingShowSwitcherExtreme()**

**'** Paul Beverley - Version 23.02.11

' Toggle visibility of track changes

If ActiveDocument.ActiveWindow.View.ShowFormatChanges = True Then

ActiveDocument.ActiveWindow.View.ShowFormatChanges = False

Exit Sub

End If

If ActiveDocument.ActiveWindow.View.ShowInsertionsAndDeletions = True Then

ActiveDocument.ActiveWindow.View.ShowInsertionsAndDeletions = False

Exit Sub

End If

If ActiveDocument.ActiveWindow.View.ShowComments = True Then

ActiveDocument.ActiveWindow.View.ShowComments = False

Exit Sub

End If

ActiveDocument.ActiveWindow.View.ShowFormatChanges = True

ActiveDocument.ActiveWindow.View.ShowInsertionsAndDeletions = True

ActiveWindow.View.ShowComments = True

End Sub

**Sub TrackChangeShowHide()**

' Paul Beverley - Version 16.07.15

' Set up track changes to taste

putTCinBubbles = True

With ActiveWindow.View

' Do you want comments showing?

.ShowComments = True

' Do you want insertions and deletions showing?

.ShowInsertionsAndDeletions = True

' Do you want format changes showing?

.ShowFormatChanges = True

' Now switch the track changes on/off

.ShowRevisionsAndComments = Not .ShowRevisionsAndComments

.RevisionsView = wdRevisionsViewFinal

End With

If putTCinBubbles = True Then

ActiveWindow.View.MarkupMode = wdBalloonRevisions

End If

End Sub

**Sub TrackChangeDisplaySelect()**

**'** Paul Beverley - Version 21.05.23

' Cycles through track change display levels

' N.B. You need to put a line at the top of the VBA area:

' Private pbMarkupMode As Integer

' (but without the ' at the beginning of the line, of course)

includeOriginal = True

If pbMarkupMode = 2 And includeOriginal = False Then

pbMarkupMode = 3

End If

Select Case pbMarkupMode

Case 1: stateNext = wdRevisionsMarkupAll

viewNext = wdRevisionsViewFinal

pbMarkupMode = 2

StatusBar = "All"

Case 2: stateNext = wdRevisionsMarkupOriginal

viewNext = wdRevisionsViewOriginal

pbMarkupMode = 3

StatusBar = "Original"

Case 3: stateNext = wdRevisionsMarkupNone

viewNext = wdRevisionsViewFinal

pbMarkupMode = 4

StatusBar = "None"

Case 4: stateNext = wdRevisionsMarkupSimple

viewNext = wdRevisionsViewFinal

StatusBar = "Simple"

pbMarkupMode = 1

Case Else

stateNext = wdRevisionsMarkupAll

viewNext = wdRevisionsViewFinal

pbMarkupMode = 2

StatusBar = "All"

End Select

With ActiveWindow.View.RevisionsFilter

.Markup = stateNext

.View = viewNext

End With

End Sub

**Sub TrackChangeDisplaySwitch()**

**'** Paul Beverley - Version 15.01.24

' Switches track change displays: Simple <=> AllMarkup

Set rng = Selection.Range.Duplicate

stateNow = ActiveWindow.View.RevisionsFilter.Markup

Select Case stateNow

Case wdRevisionsMarkupSimple: stateNext = wdRevisionsMarkupAll

Case wdRevisionsMarkupAll: stateNext = wdRevisionsMarkupSimple

Case Else: stateNext = wdRevisionsMarkupAll

End Select

With ActiveWindow.View.RevisionsFilter

.Markup = stateNext

.View = wdRevisionsViewFinal

End With

rng.Select

ActiveDocument.ActiveWindow.LargeScroll up:=1

rng.Select

ActiveDocument.ActiveWindow.SmallScroll up:=1

End Sub

**Sub VisibleTrackOff()**

**'** Paul Beverley - Version 01.01.10

' Visible reminder that track changes is off

If ActiveDocument.TrackRevisions = True Then

ActiveDocument.TrackRevisions = False

ActiveDocument.Range.Font.Emboss = True

Else

ActiveDocument.Range.Font.Emboss = False

ActiveDocument.TrackRevisions = True

End If

End Sub

**Sub VisibleTrackOff2()**

**'** Paul Beverley - Version 01.01.10

' Visible reminder that track changes is off

If ActiveDocument.TrackRevisions = True Then

ActiveDocument.TrackRevisions = False

ActiveDocument.Range.Font.Underline = wdUnderlineDotted

ActiveDocument.Range.Font.UnderlineColor = wdColorBlue

Else

ActiveDocument.Range.Font.Underline = False

ActiveDocument.TrackRevisions = True

End If

End Sub

**Sub VisibleTrackOff3()**

' Paul Beverley - Version 14.03.17

' A visible trackchange reminder - using a wiggly line!

' based on a macro by Thiers Halliwell

Set myRng = ActiveDocument.range.Borders(wdBorderRight)

Set myRng2 = ActiveDocument.range.Borders(wdBorderLeft)

If ActiveDocument.TrackRevisions = False Then

myRng.LineStyle = wdLineStyleNone

myRng2.LineStyle = wdLineStyleNone

ActiveDocument.TrackRevisions = True

Else

ActiveDocument.TrackRevisions = False

myRng.LineStyle = wdLineStyleSingleWavy

myRng2.LineStyle = wdLineStyleSingleWavy

End If

End Sub

**Sub VisibleTrackOff4()**

' Paul Beverley - Version 27.10.17

' Visible trackchange reminder - yellow background

ActiveDocument.TrackRevisions = Not ActiveDocument.TrackRevisions

If ActiveDocument.TrackRevisions = False Then \_

myColour = 200 Else myColour = 255

ActiveDocument.ActiveWindow.View.DisplayBackgrounds = True

With ActiveDocument.Background.Fill

.ForeColor.RGB = RGB(255, 255, myColour)

.Visible = msoTrue

.Solid

End With

End Sub

**Sub VisibleTrackOff5()**

' Paul Beverley - Version 06.05.17

' Visible trackchange reminder - selectable background colour

r = 216: g = 216: b = 255

ActiveDocument.TrackRevisions = Not ActiveDocument.TrackRevisions

ActiveDocument.ActiveWindow.View.DisplayBackgrounds = True

With ActiveDocument.Background.Fill

If ActiveDocument.TrackRevisions = False Then

.ForeColor.RGB = RGB(r, g, b)

Else

.ForeColor.RGB = RGB(255, 255, 255)

End If

.Visible = msoTrue

.Solid

End With

End Sub

**Sub TrackOnOffVisible()**

**'** Paul Beverley - Version 25.11.24

' Switches tracking on/off with visible background

myColourNumber = 200

markupNow = ActiveWindow.View.RevisionsFilter.Markup

If ActiveDocument.TrackRevisions = False Then GoTo turnTrackOn

Beep

ActiveDocument.TrackRevisions = False

myColour = RGB(255, 255, myColourNumber)

With ActiveDocument.Styles(wdStyleNormal).ParagraphFormat

.Shading.BackgroundPatternColor = myColour

End With

With ActiveDocument.Background.Fill

.ForeColor.RGB = RGB(255, 255, 255)

.Visible = msoTrue

.Solid

End With

With ActiveWindow.View.RevisionsFilter

.Markup = markupNow

.View = wdRevisionsViewFinal

End With

Exit Sub

turnTrackOn:

' Switch tracking back on

ActiveDocument.TrackRevisions = True

myColour = RGB(255, 255, 255)

With ActiveDocument.Styles(wdStyleNormal).ParagraphFormat

.Shading.BackgroundPatternColor = myColour

End With

With ActiveDocument.Background.Fill

.ForeColor.RGB = RGB(255, 255, myColNumber)

.Visible = msoTrue

.Solid

End With

With ActiveWindow.View.RevisionsFilter

.Markup = markupNow

.View = wdRevisionsViewFinal

End With

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

End Sub

**Sub TrackOnOffVisibleMac()**

**'** Paul Beverley - Version 20.02.24

' Switches tracking on/off with visible background

myColNumber = 200

myColour = RGB(255, 255, myColNumber)

Set rng = ActiveDocument.Content

If ActiveDocument.TrackRevisions = False Then

ActiveDocument.Styles(wdStyleNormal).ParagraphFormat.Shading.BackgroundPatternColor \_

= wdColorAutomatic

rng.Font.Shading.BackgroundPatternColor = wdColorAutomatic

ActiveDocument.TrackRevisions = True

Beep

Else

ActiveDocument.TrackRevisions = False

ActiveDocument.Styles(wdStyleNormal).ParagraphFormat.Shading.BackgroundPatternColor \_

= myColour

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

End If

End Sub

**Sub BackgroundColourOnOff()**

' Paul Beverley - Version 25.08.20

' Switches background colour on/off

myColour = RGB(233, 232, 219)

With ActiveDocument.Background.Fill

If .ForeColor.RGB = myColour Then

.ForeColor.RGB = RGB(255, 255, 255)

Else

.ForeColor.RGB = myColour

End If

.Visible = msoTrue

.Solid

End With

End Sub

**Sub** **AcceptTrackedFormatChanges()**

' Paul Beverley - Version 21.06.22

' Accepts just the formatting track changes

' Save the current state of things

filterview = ActiveWindow.View.RevisionsFilter.View

comshow = ActiveWindow.View.ShowComments

inkshow = ActiveWindow.View.ShowInkAnnotations

indelshow = ActiveWindow.View.ShowInsertionsAndDeletions

formshow = ActiveWindow.View.ShowFormatChanges

' Hide all except formatting TCs

With ActiveWindow.View

.ShowComments = False

.ShowInkAnnotations = False

.ShowInsertionsAndDeletions = False

.ShowFormatChanges = True

End With

ActiveDocument.AcceptAllRevisionsShown

' Set things back as they were

With ActiveWindow.View

.ShowComments = comshow

.ShowInkAnnotations = inkshow

.ShowInsertionsAndDeletions = indelshow

.ShowFormatChanges = formshow

.RevisionsFilter.View = filterview

End With

End Sub

**Sub AcceptSpecificTrackChange()**

' Paul Beverley - Version 02.04.11

' Accept all occurrences of one specific track change

' Alt - Ctrl - Shift - f10

On Error GoTo ReportIt

'WordBasic.PreviousChangeOrComment

restart:

thisRev = ActiveDocument.Range(0, \_

Selection.Range.Revisions(1).Range.End).Revisions.Count

Do

i = thisRev

Set myRev = ActiveDocument.Revisions(i)

myRev.Range.Select

myType = myRev.FormatDescription

thisRevString = InputBox(myType, "Accept this change?", Trim(Str(i)))

If thisRevString = "" Then Exit Sub

thisRev = Val(thisRevString)

Loop Until i = thisRev

theEnd = ActiveDocument.Content.End

i = 0

For Each rev In ActiveDocument.Range.Revisions

Set rng = rev.Range

thisType = rev.FormatDescription

If thisType = myType Then

i = i + 1

rng.Revisions.AcceptAll

End If

StatusBar = "Accepting track changes... " \_

& Str(theEnd - rng.End)

Next rev

StatusBar = "Accepted track changes: " & Str(i)

Exit Sub

ReportIt:

' There are no files open at all

If Err.Number = 5941 Then

WordBasic.NextChangeOrComment

Beep

Resume restart

Else

' Display Word's error message

On Error GoTo 0

Resume

End If

End Sub

**Sub AcceptTrackingSelectively()**

' Paul Beverley - Version 10.02.21

' Accepts track changes of specific author/editor

' acceptName = "Microsoft"

acceptName = "Paul Beverley"

For Each rv In ActiveDocument.Revisions

If InStr(rv.Author, acceptName) > 0 Then

rv.Accept

End If

DoEvents

Next rv

Beep

End Sub

**Sub TrackAcceptMoveOn()**

' Paul Beverley - Version 19.02.22

' Accepts change and moves to the next

addHighlight = False

myColour = wdYellow

addColour = True

myFontColour = wdRed

On Error GoTo ReportIt

If Selection.Start <> Selection.End Then

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If addHighlight = True Then \_

Selection.Range.HighlightColorIndex = myColour

If addColour = True Then \_

Selection.Range.Font.ColorIndex = myFontColour

ActiveDocument.TrackRevisions = myTrack

End If

hereNow = Selection.Start

Application.Run MacroName:="AcceptChangesSelected"

i = 0

Do

hereNow = Selection.Start

Application.Run MacroName:="NextChangeOrComment"

inAComment = Selection.Information(wdInCommentPane)

DoEvents

i = i + 1

Loop Until Not inAComment Or i > 10

If Selection.Start < hereNow Or i > 10 Then Beep

Exit Sub

ReportIt:

On Error GoTo 0

Application.Run MacroName:="NextChangeOrComment"

If Selection.Start < hereNow And \_

Selection.Start > 0 Then Beep

End Sub

**Sub TrackSimplifier()**

' Paul Beverley - Version 13.02.20

' Accepts certain types of tracked features

maxChars = 4

myPunctuation = ",;.?!:'""-" & ChrW(8216) & ChrW(8217) \_

& ChrW(8220) & ChrW(8221) & ChrW(8211) & ChrW(8212)

myPrompt = "+1 = Formatting" & vbCr

myPrompt = myPrompt & "+2 = Punctuation" & vbCr

myPrompt = myPrompt & "+4 = Multiple spaces" & vbCr

Do

myInput = InputBox(myPrompt, "TrackSimplifier")

myNumber = Int(Val(myInput))

If myNumber = 0 Then Exit Sub

If myNumber > 7 Then Beep

Loop Until myNumber < 8

doFormat = myNumber Mod 2

myNumber = (myNumber - doFormat) / 2

doPunct = myNumber Mod 2

myNumber = (myNumber - doPunct) / 2

doSpaces = myNumber Mod 2

comshow = ActiveWindow.View.ShowComments

inkshow = ActiveWindow.View.ShowInkAnnotations

indelshow = ActiveWindow.View.ShowInsertionsAndDeletions

formshow = ActiveWindow.View.ShowFormatChanges

If doFormat = 1 Then

' Hide all except formatting TCs

With ActiveWindow.View

.ShowComments = False

.ShowInkAnnotations = False

.ShowInsertionsAndDeletions = False

.ShowFormatChanges = True

End With

ActiveDocument.AcceptAllRevisionsShown

End If

With ActiveWindow.View

.ShowComments = True

.ShowInkAnnotations = True

.ShowInsertionsAndDeletions = True

.ShowFormatChanges = True

End With

If doPunct + doSpaces > 0 Then

For Each rv In ActiveDocument.Revisions

txt = rv.range.Text

myLen = Len(txt)

doAccept = False

If myLen <= maxChars Then

If doPunct = 1 Then

For i = 1 To myLen

If InStr(myPunctuation, Mid(txt, i, 1)) > 0 Then

doAccept = True

Exit For

End If

Next i

DoEvents

End If

If doSpaces = 1 Then

numbSpaces = Len(txt) - Len(Replace(txt, " ", ""))

If numbSpaces > 1 Then doAccept = True

If numbSpaces = 1 And Len(txt) = 1 Then doAccept = True

DoEvents

End If

If doAccept = True Then

rv.range.Revisions.AcceptAll

DoEvents

End If

End If

Next rv

End If

' Set things back as they were

With ActiveWindow.View

.ShowComments = comshow

.ShowInkAnnotations = inkshow

.ShowInsertionsAndDeletions = indelshow

.ShowFormatChanges = formshow

End With

Selection.EndKey Unit:=wdStory

Beep

End Sub

**Sub** **ConsolidateTracking()**

' Paul Beverley - Version 24.09.22

' Turns split tracking into a single change

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

Selection.Cut

Selection.Paste

End Sub

**Sub ConsolidateTrackingAll()**

' Paul Beverley - Version 17.09.22

' Consolidates tracking on all words within selection or all text

ActiveDocument.TrackRevisions = True

If Selection.Start <> Selection.End Then

Set rng = Selection

Else

myResponse = MsgBox("Work on WHOLE text?!", \_

vbQuestion + vbYesNoCancel, "ConsolidateTrackingAll")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

If Selection.Information(wdInFootnote) = True Then \_

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

If Selection.Information(wdInEndnote) = True Then \_

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

End If

For Each wd In rng.Words

If wd.Revisions.Count > 1 Then

wd.Text = wd.Text

End If

Next

End Sub

**Sub CloneWordFile()**

' Paul Beverley - Version 01.09.16

' Create a clean new copy of a document

chkParaStyles = False

chkParaStyles = True

chkFontName = False

chkFontName = True

chkFontSize = False

chkFontSize = True

chkFontColour = False

chkFontColour = True

chkHighlight = False

chkHighlight = True

chkBold = False

chkBold = True

chkItalic = False

chkItalic = True

chkSubscript = False

chkSubscript = True

chkSuperscript = False

chkSuperscript = True

chkSmallCaps = False

chkSmallCaps = True

chkUnderline = False

chkUnderline = True

timeStart = Timer

showTime = True

' The character style feature is verrry slow,

' so it's currently disabled. Ask if you're interested.

charStyle\_1 = ""

charStyle\_1 = "HTML Sample"

chkCharStyles = True

chkCharStyles = False

myStep = 10

Set sceFile = ActiveDocument

'sceFile.range.Revisions.AcceptAll

sceFile.range.Copy

' Convert character style to shadow

If charStyle\_1 > "" Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Style = ActiveDocument.Styles(charStyle\_1)

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Shadow = True

.Execute Replace:=wdReplaceAll

End With

End If

For Each thisDoc In Application.Documents

thisName = thisDoc.Name

If Left(thisName, 3) = "Doc" Then

gottaDoc = True

Exit For

End If

Next thisDoc

If gottaDoc = False Then

Documents.Add

Else

thisDoc.Activate

Selection.WholeStory

Selection.Delete

End If

Set tgtFile = ActiveDocument

tgtFile.range.Paste

tgtFile.range.Style = ActiveDocument.Styles(wdStyleNormal)

tgtFile.range.Font.Reset

tgtFile.range.Font.Color = wdColorAutomatic

tgtFile.range.HighlightColorIndex = wdNoHighlight

For i = 1 To 30

spcs = " " & spcs

Next i

totParas = sceFile.range.Paragraphs.Count

totChars = sceFile.range.Characters.Count

If chkParaStyles = True Then

For i = 1 To totParas - 1

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

' sceFile.Paragraphs(i).range.Select

myDots = " "

myStyle = sce.range.Style

If sce.range.Style <> ActiveDocument.Styles(wdStyleNormal) Then

If ActiveDocument.Styles(myStyle).Type = wdStyleTypeParagraph Then

tgt.Style = sce.range.Style

End If

End If

DoEvents

StatusBar = spcs & "Checking paragraph styles " & totParas - i

Next i

DoEvents

End If

If charStyle\_1 > "" Then

pmpt = spcs & "Checking character styles "

Debug.Print pmpt

For i = 1 To totParas - 1

If sceFile.Paragraphs(i).range.Font.Shadow > 999 Then

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

myDots = " "

For j = 1 To tgt.range.Characters.Count

Set c = sce.range.Characters(j)

tgt.range.Characters(j).Font.Shadow = c.Font.Shadow

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

Else

If sceFile.Paragraphs(i).range.Font.Shadow = True \_

Then tgtFile.Paragraphs(i).range.Font.Shadow = \_

True

End If

StatusBar = pmpt & totParas - i

DoEvents

Next i

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Shadow = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Style = ActiveDocument.Styles(charStyle\_1)

.Replacement.Font.Shadow = False

.Execute Replace:=wdReplaceAll

End With

End If

If chkCharStyles = True Then

For i = 1 To totChars - 1

If sceFile.Characters(i).Style.Type = 2 Then

tgtFile.range.Characters(i).Style = sceFile.Characters(i).Style

DoEvents

End If

chrsLeft = totChars - i

If chrsLeft Mod 10 = 0 Then StatusBar = spcs & "Character styles " & Str(chrsLeft)

Next i

End If

If chkFontName = True And sceFile.Content.Font.Name = "" Then

pmpt = spcs & "Checking font names "

Debug.Print pmpt

For i = 1 To totParas - 1

If sceFile.Paragraphs(i).range.Font.Name = "" Then

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

myDots = " "

For j = 1 To tgt.range.Characters.Count

Set c = sce.range.Characters(j)

tgt.range.Characters(j).Font.Name = c.Font.Name

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

Else

If sceFile.Paragraphs(i).range.Font.Name <> \_

sceFile.Styles(wdStyleNormal).Font.Name Then \_

tgtFile.Paragraphs(i).range.Font.Name = \_

sceFile.Paragraphs(i).range.Font.Name

End If

StatusBar = pmpt & totParas - i

Next i

End If

If chkFontSize = True And sceFile.Content.Font.Size > 999 Then

pmpt = spcs & "Checking font size "

Debug.Print pmpt

For i = 1 To totParas - 1

If sceFile.Paragraphs(i).range.Font.Size > 999 Then

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

myDots = " "

For j = 1 To tgt.range.Characters.Count

Set c = sce.range.Characters(j)

tgt.range.Characters(j).Font.Size = c.Font.Size

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

Else

If sceFile.Paragraphs(i).range.Font.Size <> \_

sceFile.Styles(wdStyleNormal).Font.Size Then \_

tgtFile.Paragraphs(i).range.Font.Size = \_

sceFile.Paragraphs(i).range.Font.Size

End If

StatusBar = pmpt & totParas - i

Next i

End If

If chkBold = True And sceFile.Content.Font.Bold > 999 Then

pmpt = spcs & "Checking bold "

Debug.Print pmpt

For i = 1 To totParas - 1

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

If sce.range.Font.Bold = True Then \_

tgt.range.Font.Bold = True

If sce.range.Font.Bold > 999 Then

myDots = " "

For j = 1 To tgt.range.Characters.Count

tgt.range.Characters(j).Font.Bold = \_

sce.range.Characters(j).Font.Bold

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

End If

StatusBar = pmpt & totParas - i

Next i

End If

If chkItalic = True And sceFile.Content.Font.Italic > 999 Then

pmpt = spcs & "Checking italic "

Debug.Print pmpt

For i = 1 To totParas - 1

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

If sce.range.Font.Italic = True Then \_

tgt.range.Font.Italic = True

If sce.range.Font.Italic > 999 Then

myDots = " "

For j = 1 To tgt.range.Characters.Count

tgt.range.Characters(j).Font.Italic = \_

sce.range.Characters(j).Font.Italic

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

End If

StatusBar = pmpt & totParas - i

Next i

End If

If chkFontColour = True And sceFile.Content.Font.Color > 999 Then

pmpt = spcs & "Checking font colour "

Debug.Print pmpt

For i = 1 To totParas - 1

If sceFile.Paragraphs(i).range.Font.Color > 999 Then

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

myDots = " "

For j = 1 To tgt.range.Characters.Count

Set c = sce.range.Characters(j)

tgt.range.Characters(j).Font.Color = c.Font.Color

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

Else

If sceFile.Paragraphs(i).range.Font.Color <> wdColorAutomatic \_

Then tgtFile.Paragraphs(i).range.Font.Color = \_

sceFile.Paragraphs(i).range.Font.Color

End If

StatusBar = pmpt & totParas - i

Next i

End If

If chkHighlight = True And sceFile.Content.HighlightColorIndex > 999 Then

pmpt = spcs & "Checking highlighting "

Debug.Print pmpt

For i = 1 To totParas - 1

If sceFile.Paragraphs(i).range.HighlightColorIndex > 999 Then

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

myDots = " "

For j = 1 To tgt.range.Characters.Count

Set c = sce.range.Characters(j)

tgt.range.Characters(j).HighlightColorIndex = c.HighlightColorIndex

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

Else

If sceFile.Paragraphs(i).range.HighlightColorIndex <> wdNoHighlight \_

Then tgtFile.Paragraphs(i).range.HighlightColorIndex = \_

sceFile.Paragraphs(i).range.HighlightColorIndex

End If

StatusBar = pmpt & totParas - i

Next i

End If

If chkSuperscript = True And sceFile.Content.Font.Superscript > 999 Then

pmpt = spcs & "Checking superscript "

Debug.Print pmpt

For i = 1 To totParas - 1

If sceFile.Paragraphs(i).range.Font.Superscript > 999 Then

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

myDots = " "

For j = 1 To tgt.range.Characters.Count

Set c = sce.range.Characters(j)

tgt.range.Characters(j).Font.Superscript = c.Font.Superscript

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

Else

If sceFile.Paragraphs(i).range.Font.Superscript = True \_

Then tgtFile.Paragraphs(i).range.Font.Superscript = \_

True

End If

StatusBar = pmpt & totParas - i

Next i

End If

If chkSubscript = True And sceFile.Content.Font.Subscript > 999 Then

pmpt = spcs & "Checking subscript "

Debug.Print pmpt

For i = 1 To totParas - 1

If sceFile.Paragraphs(i).range.Font.Subscript > 999 Then

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

myDots = " "

For j = 1 To tgt.range.Characters.Count

Set c = sce.range.Characters(j)

tgt.range.Characters(j).Font.Subscript = c.Font.Subscript

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

Else

If sceFile.Paragraphs(i).range.Font.Subscript = True \_

Then tgtFile.Paragraphs(i).range.Font.Subscript = \_

True

End If

StatusBar = pmpt & totParas - i

Next i

End If

If chkSmallCaps = True And sceFile.Content.Font.SmallCaps > 999 Then

pmpt = spcs & "Checking small caps "

Debug.Print pmpt

For i = 1 To totParas - 1

If sceFile.Paragraphs(i).range.Font.SmallCaps > 999 Then

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

myDots = " "

For j = 1 To tgt.range.Characters.Count

Set c = sce.range.Characters(j)

tgt.range.Characters(j).Font.SmallCaps = c.Font.SmallCaps

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

Else

If sceFile.Paragraphs(i).range.Font.SmallCaps = True \_

Then tgtFile.Paragraphs(i).range.Font.SmallCaps = \_

True

End If

StatusBar = pmpt & totParas - i

Next i

End If

If chkUnderline = True And sceFile.Content.Font.Underline > 999 Then

pmpt = spcs & "Checking underline "

Debug.Print pmpt

For i = 1 To totParas - 1

If sceFile.Paragraphs(i).range.Font.Underline > 999 Then

Set tgt = tgtFile.Paragraphs(i)

Set sce = sceFile.Paragraphs(i)

myDots = " "

For j = 1 To tgt.range.Characters.Count

Set c = sce.range.Characters(j)

tgt.range.Characters(j).Font.Underline = c.Font.Underline

If j Mod myStep = 0 Then

myDots = myDots & "."

StatusBar = pmpt & totParas - i & myDots

End If

DoEvents

Next j

Else

If sceFile.Paragraphs(i).range.Font.Underline = True \_

Then tgtFile.Paragraphs(i).range.Font.Underline = \_

True

End If

StatusBar = pmpt & totParas - i

Next i

End If

totTime = Timer - timeStart

If showTime = True And totTime > 10 Then

MsgBox ((Int(10 \* totTime / 60) / 10) & \_

" minutes")

End If

StatusBar = spcs & "Finished!!!!!!!"

Beep

End Sub

**Sub CloneWithEquations()**

' Paul Beverley - Version 01.09.16

' Creates a clean copy of a file, including formatted equations

doPrompts = False

doPrompts = True

numEqs = ActiveDocument.OMaths.Count

Set rng = ActiveDocument.Content

rng.Copy

' Create new temporary document

Documents.Add

Set tempDoc = ActiveDocument

Selection.Paste

If doPrompts = True Then MsgBox "Whole text copied. Continue?"

Selection.HomeKey unit:=wdStory

If doPrompts = True Then MsgBox "Sure?"

' Start working on the copy

Selection.EndKey unit:=wdStory

Selection.TypeText Text:=vbCr & vbCr & "----------" & vbCr

' Copy all equations, in order, to the end of the file

Set sc = ActiveDocument.Content

For i = 1 To numEqs

sc.OMaths(i).range.Copy

Selection.EndKey unit:=wdStory

Selection.TypeText Text:=" " & vbCr

Selection.MoveLeft , 2

hereNow = Selection.Start

Selection.Paste

Selection.Start = hereNow

Selection.End = hereNow + 1

If Asc(Selection) = 11 Then Selection.Delete

Next i

' Delete each equation from the text, replacing with a marker

For i = numEqs To 1 Step -1

ActiveDocument.OMaths(i).range.Select

Selection.Delete

Selection.TypeText "zczc" & Trim(Str(1000 + i))

Next i

' Find the equations and temporarily cut them out

Set rng = ActiveDocument.Content

theEnd = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "----------"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

rng.Select

rng.End = theEnd

rng.Select

rng.Cut

' Add markers for all the required formatting elements

' First italic

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Replacement.Text = "pqpqi^&ipqpq"

.MatchWildcards = False

.MatchCase = True

.Execute Replace:=wdReplaceAll

End With

' Bold

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Bold = True

.Replacement.Text = "pqpqb^&bpqpq"

.MatchWildcards = False

.MatchCase = True

.Execute Replace:=wdReplaceAll

End With

' Superscript

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Superscript = True

.Replacement.Text = "pqpqs^&spqpq"

.MatchWildcards = False

.MatchCase = True

.Execute Replace:=wdReplaceAll

End With

' Now put the equations back

Selection.EndKey unit:=wdStory

Selection.Paste

If doPrompts = True Then MsgBox "Equations now copied to text. Continue?"

Selection.EndKey unit:=wdStory

If doPrompts = True Then MsgBox "Sure?"

' Copy the whole of the temporary document

Set sc = ActiveDocument.Content

sc.Copy

' Create new document and paste in TEXT only

Documents.Add

Set newDoc = ActiveDocument

Selection.PasteSpecial DataType:=wdPasteText

If doPrompts = True Then MsgBox "New doc opened. Continue?"

Selection.HomeKey unit:=wdStory

If doPrompts = True Then MsgBox "Sure?"

' Now delete the useless text-only Paul Beverley - Version of the equations

Set rng = ActiveDocument.Content

theEnd = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "----------"

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

rng.End = theEnd

rng.Select

Selection.Delete

' Go back and get the formatted equations

tempDoc.Activate

Selection.HomeKey unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "----------"

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

Selection.End = ActiveDocument.Content.End

Selection.Copy

newDoc.Activate

Selection.EndKey unit:=wdStory

Selection.Paste

If doPrompts = True Then MsgBox "Equations to be copied. Continue?"

Selection.EndKey unit:=wdStory

If doPrompts = True Then MsgBox "Continue?"

' Copy the equations back into their original places

For i = 1 To numEqs

ActiveDocument.OMaths(i).range.Select

Selection.Cut

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc" & Trim(Str(1000 + i))

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

rng.Paste

' rng.Select

' Selection.Paste

Next i

' Restore formatting

' First italic

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "pqpqi(\*)ipqpq"

.Replacement.Text = "\1"

.Replacement.Font.Italic = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Bold

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "pqpqb(\*)bpqpq"

.Replacement.Text = "\1"

.Replacement.Font.Bold = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Superscript

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "pqpqs(\*)spqpq"

.Replacement.Text = "\1"

.Replacement.Font.Superscript = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

If doPrompts = True Then MsgBox "Finished!"

Selection.HomeKey unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "----------"

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

Selection.End = ActiveDocument.Content.End

Selection.Delete

Selection.HomeKey unit:=wdStory

tempDoc.Activate

If doPrompts = False Then ActiveDocument.Close SaveChanges:=False

newDoc.Activate

numEqsNow = ActiveDocument.OMaths.Count

MsgBox "Was: " & numEqs & " Now: " & numEqsNow

End Sub

**Sub PageNumberChecker()**

**'** Paul Beverley - Version 09.02.12

' Check that the text page number = Word's page number

' Is page number at the bottom of the page?

pageNumberAtBottom = False

Application.Browser.Target = wdBrowsePage

pageNow = Selection.Information(wdActiveEndPageNumber)

Do

pageWas = pageNow

Application.Browser.Next

' Have we moved at all from the line where we started

If pageNumberAtBottom = True Then

Selection.MoveStart Unit:=wdLine, Count:=-1

Else

Selection.MoveEnd Unit:=wdLine, Count:=1

End If

' move past the non-numeric characters

Selection.MoveEndUntil cset:="0123456789", Count:=wdBackward

Selection.MoveStartUntil cset:="0123456789", Count:=wdForward

myNumber = Val(Selection)

pageNow = Selection.Information(wdActiveEndPageNumber)

Loop Until pageNow <> myNumber Or pageNow = pageWas

Beep

If pageNow = pageWas Then

' Second beep to show end

myTime = Timer: Do: Loop Until Timer > myTime + 0.2: Beep

End If

End Sub

**Sub ContentsListChecker()**

**'** Paul Beverley - Version 09.02.12

' Confirm the page numbers in the contents list

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Text = "[[["

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

If rng.Find.Found = False Then

' We are in the contents list, so get the page number

Selection.EndKey Unit:=wdLine

Selection.MoveStartWhile cset:="0123456789", Count:=wdBackward

findPage = Val(Selection)

Selection.Collapse wdCollapseEnd

Selection.InsertAfter Text:="[[["

Selection.GoTo What:=wdGoToPage, Count:=findPage

Else

' We must now be back in the contents list

rng.Select

startHere = Selection.End + 1

Selection.MoveDown Unit:=wdScreen, Count:=2

Selection.End = startHere

Selection.EndKey Unit:=wdLine

End If

End Sub

**Sub IndexChecker()**

' Paul Beverley - Version 16.01.21

' Lists all pages on which this word(s) occur(s)

' Either zero or the first page number of the index

indexPageStart = 0

myColour = wdYellow

doWholeWordsOnly = False

If Selection.Start = Selection.End Then Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

myWord = Selection

indexPage = Selection.Information(wdActiveEndAdjustedPageNumber)

Set rng = ActiveDocument.Content

If indexPageStart = 0 Then indexPageStart = \_

rng.Information(wdActiveEndAdjustedPageNumber)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myWord

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.Execute

.IgnoreSpace = True

.IgnorePunct = True

End With

wasNum = 0

myFinds = ""

stopNow = False

Do While rng.Find.Found And stopNow = False

pageNum = rng.Information(wdActiveEndAdjustedPageNumber)

gotOne = True

If doWholeWordsOnly = True Then

rng.MoveEnd wdCharacter, 1

rng.MoveStart wdCharacter, -1

preChar = Left(rng, 1)

postChar = Right(rng, 1)

If UCase(preChar) <> LCase(preChar) Then gotOne = False

If UCase(postChar) <> LCase(postChar) Then gotOne = False

End If

' Got this page already?

If wasNum = pageNum Then gotOne = False

' This is the page in the index where the word occurs!

If pageNum = indexPage Then gotOne = False

' We've arrived at the index, so give up looking

If pageNum >= indexPageStart Then gotOne = False: stopNow = True

' Add it to the list of finds

If gotOne = True Then myFinds = myFinds & ", " & Trim(Str(pageNum))

wasNum = pageNum

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

With Selection.Find

.IgnoreSpace = False

.IgnorePunct = False

End With

myFinds = Mid(myFinds, 3)

myResponse = MsgBox(myFinds & vbCr & vbCr & "OK?", vbQuestion + vbYesNo)

If myResponse = vbYes Then

Selection.Collapse wdCollapseStart

Selection.MoveDown Unit:=wdLine, Count:=1

Else

Selection.EndKey Unit:=wdLine

Selection.InsertAfter Text:=" >>>>>>> " & myFinds

Selection.Range.HighlightColorIndex = myColour

Selection.Range.Font.Bold = True

Selection.Collapse wdCollapseEnd

End If

End Sub

**Sub AlphaHeadersOnIndex()**

' Paul Beverley - Version 10.09.19

' Adds alpha headers to an index

myHead = "`" ' One less than "a"

For Each pa In ActiveDocument.Paragraphs

myNewHead = ""

paText = LCase(pa.range.Text)

If Asc(paText) > Asc(myHead) Then

myNewHead = Left(paText, 1)

pa.range.Select

Selection.InsertBefore Text:=vbCr & UCase(myNewHead) & vbCr

Selection.End = Selection.Start + 2

Selection.range.Font.Bold = True

myHead = myNewHead

End If

Next pa

End Sub

**Sub CitationAlyse()**

**'** Paul Beverley - Version 26.09.24

' Checks citations against references

myFontColour = wdColorBlue

myDateColour = wdColorRed

includeNotes = True

' This next feature is not that reliable

' but try it, if you like, i.e. make it true

' doTrySimplePairing = True

doTrySimplePairing = False

notNames = "In From For Act Jan January Feb February "

notNames = notNames & " Mar March Apr April NotMay June "

notNames = notNames & " July Aug August Sept September Oct Initially "

notNames = notNames & " October Nov November Dec December Finally "

notNames = notNames & " Also As Conversely Correspondingly Consequently "

notNames = notNames & " Equally Furthermore Lastly Moreover However "

notNames = notNames & " Secondly Thirdly Similarly Additionally "

' This is for demo purposes

showTheList = False

' Can't remember why I did this.

' Make it true to see what it does!

addFunnyColours = False

CR = vbCr

CR2 = CR & CR

myScreenOff = True

Dim myCol(4)

myCol(0) = wdYellow

myCol(1) = wdBrightGreen

myCol(2) = wdRed

myCol(3) = wdTurquoise

myCol(4) = wdGray25

' If text is selected, sort it

If Selection.start <> Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

Selection.start = rng.start

Set rng = Selection.Range.Duplicate

If Right(rng.Text, 1) <> vbCr Then

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

Selection.End = rng.End

End If

myResponse = MsgBox("Sort selection by year?", vbQuestion \_

+ vbOK, "CitationAlyse")

If myResponse <> vbOK Then Exit Sub

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

Dim myPara As Paragraph

Set rng = Selection.Range.Duplicate

For Each myPara In rng.Paragraphs

myText = myPara.Range.Text

For i = 2 To Len(myText) - 3

myYearText = Mid(myText, i, 5)

If InStr("abcdefghijkl", Right(myYearText, 1)) = 0 Then \_

myYearText = Left(myYearText, 4)

myYear = Val(myYearText)

If myYear > 1000 And Left(myYearText, 1) <> " " Then

myPara.Range.InsertBefore Text:="]" & myYearText & "["

Exit For

End If

DoEvents

Next i

DoEvents

Next myPara

Selection.Sort

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\][0-9a-h]@\["

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Application.ScreenUpdating = True

Exit Sub

End If

' Main citation listing section

Beep

myResponse = MsgBox("Is the cursor placed in the first item" \_

& CR2 & "of the references list?", vbQuestion \_

+ vbYesNoCancel, "CitationAlyse")

If myResponse <> vbYes Then Exit Sub

On Error GoTo ReportIt

Application.ScreenUpdating = False

' copy references right to the end

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

startRefs = Selection.start

Set rngOld = ActiveDocument.Content

rngOld.start = startRefs

rngOld.Copy

rngOld.Collapse wdCollapseStart

rngOld.start = 0

' create new file of the text (not refs) + notes

Set mainDoc = ActiveDocument

Documents.Add

Selection.Text = rngOld.Text

' Copy footnotes and endnotes, text only

gotFoots = (mainDoc.Footnotes.Count > 0)

gotEnds = (mainDoc.Endnotes.Count > 0)

DoEvents

If gotFoots = True And includeNotes = True Then

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr

Selection.Text = mainDoc.StoryRanges(wdFootnotesStory).Text

End If

If gotEnds = True And includeNotes = True Then

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr

Selection.Text = mainDoc.StoryRanges(wdEndnotesStory).Text

End If

DoEvents

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ",([0-9]{4})"

.Wrap = wdFindStop

.Replacement.Text = ", \1"

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-z] \([A-Z]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.MoveStart wdWord, -1

rng.MoveEnd wdCharacter, -1

' rng.Select

rng.Text = LCase(rng.Text)

rng.MoveStart wdWord, -1

' rng.Select

myMidWord = Left(rng, 3)

If myMidWord = "und" Or myMidWord = "and" Then

rng.MoveStart wdWord, -1

rng.Text = LCase(rng.Text)

End If

' rng.Select

rng.start = endNow + 2

rng.End = endNow + 2

rng.Find.Execute

DoEvents

Loop

' Find and replace to create easy-to-search text

myList = "#ü|uu#û|qcq#~<([A-Z]{2}).|\1# in press| 2999#/| #" & \_

"~<([A-Z]).|\1#(| #)| #:| #;| #, | #" & \_

"~<([A-Z])> ([A-Z])|\1ü\2#~<([A-Z]{2})> ([A-Z])|\1ü\2#" & \_

"#.|ü#(|ü#)|ü#[|ü#]|ü#" & \_

" et alii|üetüal# et al|üetüal# & |üandü# und |üandü#-|û#" & \_

"~<de la |DeüLaü#~<De la |DeüLaü#~<De La |DeüLaü#" & \_

" e |üeü# y |üyü#~<van |Vanü#~<de |Deü#" & \_

" and |üandü#~<von |Vonü#~<van |Vanü#~<de |Deü#" & \_

"~<El |Elü#~<del |Delü#~<Del |Delü#~<la |Laü#~<La |Laü#" & \_

"~<le |Leü#~<di |Diü#~<dos |Dosü#~<la |Laü#~<le |Leü#" & \_

"~<der |Derü#üvon |üVonü#üvan |üVanü#üde |üDeü#üder |üDerü#" & \_

"üdi |üDiü#üdos |üDosü#üla |üLaü#üle |üLeü# ü| #ü | #" & \_

"~([0-9])ü|\1 #~ü([0-9])| \1# | #"

FandR = Split(myList, "#")

Set rng = ActiveDocument.Content

For i = 1 To UBound(FandR)

myFR = FandR(i)

If Left(myFR, 1) <> ChrW(124) And \_

InStr(myFR, ChrW(124)) > 0 Then

If Left(myFR, 1) = "~" Then

myWild = True

myFR = Mid(myFR, 2)

Else

myWild = False

End If

barpos = InStr(myFR, ChrW(124))

myFind = Left(myFR, barpos - 1)

myRep = Mid(myFR, barpos + 1)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = myRep

.Replacement.Highlight = True

.MatchWildcards = myWild

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

Next i

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[12][0-9]{3}[!0-9]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

' Find & Do to find all years

Do While rng.Find.Found = True

myEnd = rng.End

myYear = Replace(rng, "ü", "")

myYear = Replace(myYear, ",", "")

myYear = Replace(myYear, ";", "")

myYear = Trim(myYear)

rng.Collapse wdCollapseStart

rng.MoveStart wdWord, -1

myInit = Left(rng, 1)

Debug.Print rng

If (UCase(myInit) <> myInit Or UCase(myInit) \_

= LCase(myInit)) And Left(rng, 3) <> "and" Then GoTo getNext

If Left(rng, 3) = "and" Then rng.start = rng.start + 4

wd1 = Trim(rng)

rng.Collapse wdCollapseStart

rng.MoveStart wdWord, -1

wd2 = Trim(rng)

myInit = Left(wd2, 1)

gotTwo = (UCase(myInit) = myInit) And \_

(UCase(myInit) <> LCase(myInit))

' Find surname one and add to citeList1

citeList1 = citeList1 & wd1 & " " & myYear & CR

' Find surname two and add to citeList2

If gotTwo Then citeList2 = citeList2 & wd2 & \_

" " & wd1 & " " & myYear & CR

' Check if another date follows

rng.start = myEnd

rng.End = myEnd

Do

rng.MoveEnd wdWord, 1

If Len(rng) < 4 Then

rng.Collapse wdCollapseEnd

rng.MoveEnd wdWord, 1

End If

myYearNumber = Val(rng)

If Val(rng) > 1000 Then

myYear = Trim(rng)

citeList1 = citeList1 & wd1 & " " & myYear & CR

If gotTwo Then citeList2 = citeList2 & wd2 & \_

" " & wd1 & " " & myYear & CR

End If

rng.MoveStart wdWord, 1

myEnd = rng.start

DoEvents

Loop Until myYearNumber < 1000

getNext:

rng.start = myEnd

rng.End = myEnd

rng.Find.Execute

Loop

Set rng = ActiveDocument.Content

rng.Delete

Selection.TypeText Text:=citeList1

myEnd = ActiveDocument.Content.End

Selection.TypeText Text:=citeList2

Set rng = ActiveDocument.Content

rng.start = myEnd

' rng.Font.Italic = True

Set rng = ActiveDocument.Content

rng.Font.Color = myFontColour

' Change by F&R û to -, ü to space, uu to ü, qcq to û

myList = "#ü| #uu|ü#û|-#qcq|û# 2999| in press#"

FandR = Split(myList, "#")

Set rng = ActiveDocument.Content

For i = 1 To UBound(FandR)

myFR = FandR(i)

If Left(myFR, 1) <> ChrW(124) And \_

InStr(myFR, ChrW(124)) > 0 Then

barpos = InStr(myFR, ChrW(124))

myFind = Left(myFR, barpos - 1)

myRep = Mid(myFR, barpos + 1)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = myRep

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

Next i

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

' Remove duplicates

numPars = ActiveDocument.Paragraphs.Count

For j = numPars To 2 Step -1

Set rng1 = ActiveDocument.Paragraphs(j).Range

Set rng2 = ActiveDocument.Paragraphs(j - 1).Range

If rng1 = rng2 Then rng1.Delete

DoEvents

Next j

' Highlight multiple part citations

numPars = ActiveDocument.Paragraphs.Count

Set rng = ActiveDocument.Content

allText = rng

myStrike = False

If addFunnyColours = True Then

For i = 1 To numPars

myText = ActiveDocument.Paragraphs(i).Range.Text

numTimes = Len(Replace(allText, myText, myText & "!")) - Len(allText)

If numTimes > 1 And Len(myText) > 2 Then

addCol = Int(5 \* Rnd())

baseCol = Int(5 \* Rnd())

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Replace(myText, vbCr, "")

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = False

.Execute

DoEvents

End With

Do While rng.Find.Found

rng.HighlightColorIndex = myCol(baseCol)

For j = 1 To rng.Characters.Count

If (j Mod 5) < 2 Then

rng.Characters(j).HighlightColorIndex = myCol(addCol)

End If

Next j

DoEvents

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

End If

DoEvents

Next i

End If

Options.DefaultHighlightColorIndex = oldColour

If showTheList = True Then

Application.ScreenUpdating = True

For i = 1 To 20

DoEvents

Next i

MsgBox "Here's the list of citations"

End If

If myScreenOff = True \_

Then Application.ScreenUpdating = False

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr

wasEnd = Selection.End

Selection.Paste

Set rng = ActiveDocument.Content

rng.start = wasEnd

rng.Fields.Unlink

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p "

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Add author name to "ditto" type references

myPrevAuthor = ""

For i = 1 To rng.Paragraphs.Count

Set myRef = rng.Paragraphs(i).Range

myWord = Trim(myRef.Words(1))

If Len(myRef.Text) > 1 And Len(myWord) = 1 Then

myRef.Select

myWord = myRef.Words(2)

End If

mySample = Left(myRef.Text, 2)

If Len(myRef.Text) > 1 And LCase(mySample) = \_

UCase(mySample) Then

myRef.Words(1) = myPrevAuthor & " "

Else

myPrevAuthor = myWord

End If

DoEvents

Next i

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

DoEvents

Set rng = ActiveDocument.Content

rng.InsertAfter Text:=CR2

Set rng = ActiveDocument.Content

rng.ParagraphFormat.LineSpacingRule = wdLineSpaceSingle

rng.ParagraphFormat.SpaceBefore = 12

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[12][0-9a-j]{3,4}>"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "^&"

.Replacement.Font.Color = myDateColour

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Citation Query List"

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

' Remove blank lines at the top

Set para2 = ActiveDocument.Paragraphs(2)

Do While Len(para2.Range.Text) < 3

para2.Range.Delete

Loop

notNames = notNames & " "

For i = ActiveDocument.Paragraphs.Count To 2 Step -1

Set rng = ActiveDocument.Paragraphs(i).Range

If InStr(notNames, rng.Words(1).Text) > 0 Then \_

rng.Words(1).Delete

Next i

' Add year at beginning of line

Set rng = ActiveDocument.Content

For Each myPara In rng.Paragraphs

myText = myPara.Range.Text

For i = 2 To Len(myText) - 3

myYearText = Mid(myText, i, 5)

If InStr("abcdefghijkl", Right(myYearText, 1)) = 0 Then \_

myYearText = Left(myYearText, 4)

myYear = Val(myYearText)

If myYear > 1000 And Left(myYearText, 1) <> " " Then

myPara.Range.InsertBefore Text:="]" & myYearText & "["

Exit For

End If

DoEvents

Next i

DoEvents

Next myPara

' Sort

If rng.Words(1) = "Citation " Then

rng.start = rng.Paragraphs(1).Range.End

End If

rng.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

DoEvents

If rng.End = ActiveDocument.Content.End Then \_

rng.InsertAfter Text:=vbCr

numPars = rng.Paragraphs.Count

myStart = 5

If numPars < myStart Then myStart = numPars

For i = myStart To 1 Step -1

Set rng2 = rng.Paragraphs(i).Range

If Len(rng2.Text) < 2 Then rng2.Delete

Next i

' Add blank line at year change

Set rng = ActiveDocument.Content

wasYear = ""

For i = rng.Paragraphs.Count To 1 Step -1

myYear = Left(rng.Paragraphs(i).Range.Text, 5)

If myYear <> wasYear Then \_

rng.Paragraphs(i).Range.InsertAfter Text:=vbCr

wasYear = myYear

DoEvents

Next i

' Remove the year markers

myBar = "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"

myBar = "^p" & myBar & myBar & "^p"

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p^p"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = myBar

.Replacement.Highlight = False

.Replacement.Font.Color = wdColorAutomatic

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\][0-9a-h]@\["

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

ActiveDocument.Paragraphs(2).Range.Text = ""

Beep

Application.ScreenUpdating = True

nmlFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "~"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Name = nmlFont

.Replacement.Font.Size = 12

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

If doTrySimplePairing = False Then Exit Sub

' Attempt simple pairing

Set rngAll = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = rngAll.FormattedText

Set rngAll = ActiveDocument.Content

rng.Font.Underline = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "."

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^p"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Color = wdBlack

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[\(\)]"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = wdColorBlue

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

myName = Trim(rng.Text)

nameStart = rng.start

rng.Collapse wdCollapseEnd

rng.MoveEnd wdWord, 1

myYear = rng.Text

rng.start = nameStart

rng.Font.Underline = True

rngAll.Collapse wdCollapseEnd

With rngAll.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myName & "[ ,a-zA-Z\-" & ChrW(8217) & "]{1,}" & myYear

.Wrap = wdFindContinue

.Font.Underline = False

.Replacement.Text = ""

.MatchWildcards = True

.Execute

If rngAll.Text = rng.Text Then

rngAll.Font.Underline = True

.Execute

End If

rngAll.Expand wdParagraph

If rngAll.Words.Count < 12 Then

.Execute

End If

End With

If rngAll.Find.Found Then

rng.Font.Color = wdColorGray25

' You have to apply a highlight, then remove it because

' VBA refuses to remove the highlight on a single space!

rng.HighlightColorIndex = wdYellow

rng.HighlightColorIndex = wdNoHighlight

rngAll.Select

rngAll.Expand wdParagraph

rngAll.Font.Color = wdColorGray25

rngAll.HighlightColorIndex = wdNoHighlight

rngAll.Collapse wdCollapseEnd

End If

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = wdColorBlue

.Font.Underline = False

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

DoEvents

myCount = myCount + 1

If myCount Mod 20 = 0 Then rng.Select

Loop

Beep

Set rng = ActiveDocument.Content

rng.Font.Underline = False

Selection.HomeKey Unit:=wdStory

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub ShortTitleUnderline()**

' Paul Beverley - Version 17.03.22

' Underlines the current author name(s)

Selection.Expand wdWord

Set rng = Selection.Range.Duplicate

Do

rng.Move wdWord, 1

rng.MoveEnd , 1

Debug.Print rng

Loop Until UCase(rng) = LCase(rng) And rng <> "-"

rng.MoveEnd , -1

rng.Start = Selection.Start

rng.Select

Selection.Range.Font.Underline = True

End Sub

**Sub ShortTitleAlyse()**

' Paul Beverley - Version 17.03.22

' Analyses Short titles to check references against citations

myFontColour = wdColorBlue

myDateColour = wdColorRed

Set rng = ActiveDocument.Content

noteStart = InStr(rng, vbCr & vbCr & vbCr)

rng.Start = noteStart - 1

If rng.End - noteStart < 100 Then noteStart = 0

If noteStart = 0 Then

myResponse = MsgBox("Please separate footnotes" & \_

" and references with two blank lines.", \_

vbOK, "ShortTitleAlyse")

Exit Sub

End If

' Make sure each note has a space at the beginning

For Each par In rng.Paragraphs

myLine = par.Range.Text

If Asc(myLine) <> 32 And Len(myLine) > 2 Then \_

par.Range.InsertBefore Text:=" "

Next par

' Copy into a new blank file

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

' Colour the notes

Set rng = ActiveDocument.Content

noteStart = InStr(rng, vbCr & vbCr & vbCr)

rng.Start = noteStart - 1

rng.Font.Color = myFontColour

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Font.Underline = True

.Replacement.Text = "^pzczc"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Years in red

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]{4}>"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Color = wdColorRed

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "See "

.Replacement.Text = "^pzczc "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^13[^2 ]"

.Replacement.Text = "^pzczc "

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

' Drop each author name onto a new line

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Replacement.Text = "^p^&"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

rng.Sort

Do

i = i + 1

DoEvents

Loop Until rng.Paragraphs(i).Range.Words.Count > 2

rng.Paragraphs(i).Range.Select

Selection.Start = 0

Selection.Delete

Set rng = ActiveDocument.Content

zcPos = InStr(rng, "zczc")

rng.Start = zcPos - 1

rng.Delete

Set rng = ActiveDocument.Content

With rng.ParagraphFormat

.LeftIndent = CentimetersToPoints(1)

.SpaceBeforeAuto = False

.SpaceAfterAuto = False

.FirstLineIndent = CentimetersToPoints(-1)

.SpaceAfter = 12

.LineUnitAfter = 0

End With

myBar = "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"

myBar = myBar & myBar & "^p"

Set rng = ActiveDocument.Content

numParas = rng.Paragraphs.Count

ReDim auNames(numParas) As String

For i = 1 To numParas

auNames(i) = rng.Paragraphs(i).Range.Words(1)

Next i

gotArun = False

nameWas = rng.Words(1)

runStart = 0

For i = 1 To numParas - 2

name1 = auNames(i)

name2 = auNames(i + 1)

name3 = auNames(i + 2)

If name1 <> name2 And name2 = name3 Then

Set rng = ActiveDocument.Paragraphs(i + 1).Range

rng.InsertBefore Text:="zczc"

End If

If name1 = name2 And name2 <> name3 Then

Set rng = ActiveDocument.Paragraphs(i + 2).Range

rng.InsertBefore Text:="zczc"

End If

DoEvents

Next i

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczczczc"

.Wrap = wdFindContinue

.Replacement.Text = myBar

.Replacement.Font.Underline = False

.Replacement.Font.Color = wdColorAutomatic

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "zczc"

.Execute Replace:=wdReplaceAll

End With

Beep

End Sub

**Sub ParagraphShrink()**

' Paul Beverley - Version 07.01.22

' Shrinks the current paragraph (for use with CitationAlyse)

mySize = 6

' myHighlight = wdYellow

myHighlight = wdNoHighlight

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Font.Size = mySize

If myHighlight > 0 Then rng.HighlightColorIndex = myHighlight

rng.Collapse wdCollapseEnd

rng.Select

End Sub

**Sub CiteFileLink()**

**'** Paul Beverley - Version 18.04.12

' Link text and references files for CiteCheck

Set textDoc = ActiveDocument

Application.GoBack

Set refsDoc = ActiveDocument

' We are now in the references

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "textDocName" Then varExists = True

If v.Name = "refsDocName" Then v.Delete

Next v

If varExists = False Then

ActiveDocument.Variables.Add "textDocName", textDoc.Name

Else

ActiveDocument.Variables("textDocName") = textDoc.Name

End If

' Now go to the text file

textDoc.Activate

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "refsDocName" Then varExists = True

If v.Name = "textDocName" Then v.Delete

Next v

If varExists = False Then

ActiveDocument.Variables.Add "refsDocName", refsDoc.Name

Else

ActiveDocument.Variables("refsDocName") = refsDoc.Name

End If

myResponse = MsgBox("Text = " & textDoc.Name & " References = " & refsDoc.Name, \_

vbQuestion + vbYesNoCancel)

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then

myResponse = MsgBox("Text = " & refsDoc.Name & " References = " & textDoc.Name, \_

vbQuestion + vbYesNoCancel)

If myResponse = vbCancel Then Exit Sub

If myResponse = vbYes Then

textDoc.Activate

' We are now in the references

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "textDocName" Then varExists = True

If v.Name = "refsDocName" Then v.Delete

Next v

If varExists = False Then

ActiveDocument.Variables.Add "textDocName", refsDoc.Name

Else

ActiveDocument.Variables("textDocName") = refsDoc.Name

End If

refsDoc.Activate

' We are now in the text

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "refsDocName" Then varExists = True

If v.Name = "textDocName" Then v.Delete

Next v

If varExists = False Then

ActiveDocument.Variables.Add "refsDocName", textDoc.Name

Else

ActiveDocument.Variables("refsDocName") = textDoc.Name

End If

End If

End If

' Prepare the Find to look for dates

With Selection.Find

.ClearFormatting

.Highlight = False

.Replacement.ClearFormatting

.Text = "[0-9]{4}[a-k\);:,]"

.Replacement.Text = ""

.Forward = True

.Wrap = False

.MatchWildcards = True

.Execute

End With

End Sub

**Sub CiteFileLink2()**

' Paul Beverley - Version 25.06.12

' Link text and references files for CiteCheck

notTheseFiles = "zzSwitchList,zzFReditList,AllTextWord"

gottaRefs = False

gottaText = False

Do Until gottaRefs = True And gottaText = True

For Each myWnd In Application.Windows

myWnd.Document.Activate

thisFile = Replace(myWnd.Document.Name, ".docx", "")

thisFile = Replace(myWnd.Document.Name, ".doc", "")

If InStr(notTheseFiles, thisFile) = 0 Then

thisOne = InputBox("Text or Refs?", "CiteFileLink")

If Len(thisOne) > 0 Then

If Asc(LCase(thisOne)) = Asc("r") Then

Set refsDoc = myWnd.Document

gottaRefs = True

End If

If Asc(LCase(thisOne)) = Asc("t") Then

Set textDoc = myWnd.Document

gottaText = True

End If

End If

End If

If gottaRefs = True And gottaText = True Then Exit For

Next myWnd

Loop

refsDoc.Activate

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "textDocName" Then varExists = True

If v.Name = "refsDocName" Then v.Delete

Next v

If varExists = False Then

ActiveDocument.Variables.Add "textDocName", textDoc.Name

Else

ActiveDocument.Variables("textDocName") = textDoc.Name

End If

textDoc.Activate

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "refsDocName" Then varExists = True

If v.Name = "textDocName" Then v.Delete

Next v

If varExists = False Then

ActiveDocument.Variables.Add "refsDocName", refsDoc.Name

Else

ActiveDocument.Variables("refsDocName") = refsDoc.Name

End If

' Prepare the Find to look for dates

With Selection.Find

.ClearFormatting

.Highlight = False

.Replacement.ClearFormatting

.Text = "[0-9]{4}[a-k\);:,]"

.Replacement.Text = ""

.Forward = True

.Wrap = False

.MatchWildcards = True

.Execute

End With

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub CiteFindName()**

**'** Paul Beverley - Version 21.10.11

' Jump back from date to name

beepIFand = True

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z]"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = False

.MatchWildcards = True

.Execute

End With

backHere = Selection.Start

Selection.MoveLeft Unit:=wdCharacter, Count:=3

Selection.Words(1).Select

If Selection = "and " Then

Selection.MoveLeft Unit:=wdCharacter, Count:=3

Selection.Words(1).Select

Selection.End = Selection.Start

If beepIFand = True Then Beep

Else

Selection.Start = backHere

Selection.Words(1).Select

Selection.End = Selection.Start

End If

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = oldFind

.Forward = True

.Highlight = False

.Replacement.Text = oldReplace

.MatchWildcards = True

.Wrap = wdFindContinue

End With

End Sub

**Sub CiteCheck()**

' Paul Beverley - Version 09.12.16

' Checks reference citations

moveOnAfterOK = True

citeCheckBackwards = False

wholeName = False

mainColour = wdYellow

myWarningColour = wdRed

myCitationColour = wdGray25

' If you want a beep warning for a multi-author citation...

addaBeepMulti = False

myCite = "myTempBookmark"

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

oldWild = Selection.Find.MatchWildcards

oldColour = Options.DefaultHighlightColorIndex

isRefs = Selection.range.Font.Underline

' Refs doc procedure

If isRefs Then

If Selection.Start = Selection.End Then Selection.Expand wdWord

myName = Trim(Selection)

' remember whether the selection is highlighted

thisColour = Selection.range.HighlightColorIndex

isHighlight = (thisColour = myWarningColour) Or thisColour > 255

' check for a hyphenated name

Selection.MoveRight unit:=wdWord, Count:=2, Extend:=wdExtend

If Selection.range.Words(2) = "-" Then myName = Trim(Selection)

Selection.Collapse wdCollapseStart

' if the name (beginning of the line) is highlighted,

' unhighlight all identical author names

If isHighlight Then

Selection.Expand wdParagraph

Selection.MoveLeft , 2

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13" & myName & "??"

.Font.Underline = True

.Replacement.Text = ""

.Replacement.Highlight = False

.Wrap = wdFindContinue

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

' go back to the text

If ActiveDocument.Bookmarks.Exists(myCite) Then

ActiveDocument.Bookmarks(myCite).Select

Else

Beep

End

If moveOnAfterOK = True Then

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Highlight = False

.Replacement.ClearFormatting

.Text = "[0-9]{4}[a-k\);:,]"

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindStop

.MatchWildcards = True

.Execute

End With

Selection.MoveLeft unit:=wdWord, Count:=2

Selection.Expand wdWord

Exit Sub

End If

GoTo restoreFandR

End If

' Text doc procedure

ActiveDocument.Bookmarks.Add Name:="myTempMark"

' OR, if you're in the text, select the current word

Selection.Expand wdWord

Selection.MoveEndWhile cset:=ChrW(8217) & " '", Count:=wdBackward

myName = Trim(Selection)

' remember where the name starts and ends

Set rngName = Selection.range.Duplicate

'nameStart = Selection.Start

'nameEnd = Selection.End

Selection.MoveRight unit:=wdWord, Count:=2, Extend:=wdExtend

Selection.MoveEndWhile cset:=ChrW(8217) & " '", Count:=wdBackward

If Selection.range.Words(2) = "-" Then myName = Trim(Selection)

' find the next date

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Text = "[01][0-9]{3}"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Set rngDate = Selection.range.Duplicate

' check if it's '2005a', '2005b' etc

Selection.Collapse wdCollapseEnd

Selection.MoveEnd , 1

mySuffix = Selection

If Asc(mySuffix) < 97 Or Asc(mySuffix) > 122 Then

Selection.MoveEnd , -1

End If

'Selection.Start = dateStart

'myDate = Selection

'dateEnd = Selection.End

' Record whole citation

Selection.Start = rngNameStart

wholeCitation = Selection

isMulti = False

If InStr(Selection, " and ") > 0 Or InStr(Selection, " and ") > 0 \_

Or InStr(Selection, " et al") > 0 Then isMulti = True

Selection.Collapse wdCollapseStart

' Find the beginning of the references

Set rng = ActiveDocument.Content

With Selection.Find

.ClearFormatting

.Text = ""

.Replacement.Text = ""

.Font.Underline = True

.Forward = True

.MatchWildcards = False

.Execute

End With

' Selection.MoveLeft Unit:=wdCharacter, Count:=1

' find the name

If citeCheckBackwards = True Then

crBit = ""

cr13Bit = ""

Else

crBit = "^p"

cr13Bit = "^13"

End If

Selection.HomeKey unit:=wdStory

With Selection.Find

.ClearFormatting

.Text = crBit & myName

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindStop

.MatchWildcards = False

.Execute

End With

' if no such reference exists then

' try to find a name starting with the same

' letters, and then exit

If Selection.Find.Found = False Then

For i = 1 To Len(myName) - 1

Beep

myTime = Timer: Do: Loop Until Timer > myTime + 0.05

myName = Left(myName, Len(myName) - 1)

With Selection.Find

.ClearFormatting

.Text = crBit & myName

.Execute

End With

If Selection.Find.Found = True Then GoTo restoreFandR

Next

GoTo restoreFandR

End If

' Changes highlight colour

Options.DefaultHighlightColorIndex = mainColour

Do

' if you've found one, is it the right date?

Selection.MoveStart , 1

Selection.MoveDown unit:=wdParagraph, Count:=1, Extend:=wdExtend

wholeRef = Selection

myStart = Selection.Start

myEnd = Selection.End

With Selection.Find

.ClearFormatting

.Text = myDate

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

gotOne = Selection.Find.Found

Selection.Find.MatchWildcards = False

Selection.Start = Selection.End

If Asc(Selection) > 96 And Asc(Selection) < 123 Then gotOne = False

If gotOne = True Then

dateEnd = Selection.End

' Check the current highlight colour a few characters to the right

Selection.Start = Selection.End - 1

hiColour = Selection.range.HighlightColorIndex

Selection.Start = myStart

Selection.End = myStart

' Move selection to top of the screen

Selection.MoveDown unit:=wdScreen, Count:=2

Selection.End = myStart - 1

Selection.MoveRight unit:=wdCharacter, Count:=1

' Ask if OK?

Selection.End = dateEnd + 1

Selection.range.HighlightColorIndex = mainColour

If isMulti = True Then

If addaBeepMulti = True Then Beep

Selection.range.HighlightColorIndex = myWarningColour

End If

Selection.End = Selection.Start

myResponse = MsgBox("OK?", vbQuestion + vbYesNo)

If myResponse = vbNo Then

Selection.MoveDown unit:=wdParagraph, Count:=1, Extend:=wdExtend

Selection.range.HighlightColorIndex = hiColour

Selection.End = Selection.Start

GoTo restoreFandR

Else

' Unhighlight reference

Selection.MoveDown unit:=wdParagraph, Count:=1, Extend:=wdExtend

Selection.range.HighlightColorIndex = wdNoHighlight

Selection.End = Selection.Start

' Go back to text and grey all identical citations

lookingFor = textDoc

Windows(textDoc).Activate

If Err.Number = 5941 Then

Err.Clear

textDoc = Replace(textDoc, ".", " [Compatibility Mode].")

Windows(textDoc).Activate

End If

Options.DefaultHighlightColorIndex = myCitationColour

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Selection.Start = Selection.End

Set rng = ActiveDocument.range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = wholeCitation

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Wrap = wdFindContinue

.Execute Replace:=wdReplaceAll

End With

ActiveDocument.TrackRevisions = nowTrack

Options.DefaultHighlightColorIndex = oldColour

If moveOnAfterOK = True Then

With Selection.Find

.ClearFormatting

.Highlight = False

.Replacement.ClearFormatting

.Text = "[0-9]{4}[a-k\);:,]"

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindStop

.MatchWildcards = True

.Execute

End With

Exit Sub

End If

End If

GoTo restoreFandR

End If

' but if not the right date, look for next name

Selection.Start = Selection.End

Selection.MoveLeft unit:=wdCharacter, Count:=1

With Selection.Find

.ClearFormatting

.Text = crBit & myName

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

nameStart = Selection.Start + 1

' keep looking until no such date can be found

Loop Until Selection.Find.Found = False

' if no such date for that author, highlight all

' like names, but first choose your favourite

' highlighting colour

Selection.HomeKey unit:=wdStory

Beep

' find the first such name

Options.DefaultHighlightColorIndex = myWarningColour

'Set rng = ActiveDocument.Range

With Selection.Find

.ClearFormatting

If wholeName = True Then

.Text = cr13Bit & myName & "[!a-z]"

Else

.Text = cr13Bit & myName

End If

.Replacement.Text = ""

.Replacement.Highlight = True

.Forward = True

.MatchWildcards = True

.Execute

End With

' highlight them all

Selection.Find.Execute Replace:=wdReplaceAll

Selection.MoveStart , 1

Selection.End = Selection.Start

For i = 1 To 3

Beep

myTime = Timer: Do: Loop Until Timer > myTime + 0.1

Next

restoreFandR:

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = oldWild

End With

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub MultiFileCopier()**

' Paul Beverley - Version 13.12.17

' Save a duplicate set of files with a postfix

toFolderName = "aaMainText"

' toFolderName = "WorkingText"

myPostFix = "\_PB\_01"

' myPostFix = ""

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Copy ALL the files in this folder:" & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "Multifile Copier")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Copy just the files listed here?", \_

vbQuestion + vbYesNoCancel, "Multifile Copier")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

' Check if new folder is ready

baseFolder = myFolder

Do

baseFolder = Left(baseFolder, Len(baseFolder) - 1)

Loop Until Right(baseFolder, 1) = myDelimiter

Set fs = CreateObject("Scripting.FileSystemObject")

If fs.FolderExists(baseFolder & toFolderName) = False Then

MsgBox "Please create folder: " & toFolderName

Exit Sub

End If

' Load files and save copies

For i = 1 To numFiles

thisName = allMyFiles(i)

newName = toFolderName & myDelimiter & Replace(thisName, \_

".", myPostFix & ".")

thisFile = myFolder & myDelimiter & thisName

Set myDoc = Application.Documents.Open(FileName:=thisFile)

StatusBar = allMyFiles(i)

DoEvents

myNewFile = baseFolder & newName

ActiveDocument.SaveAs FileName:=myNewFile

myDoc.Close SaveChanges:=wdDoNotSaveChanges

Next

Selection.HomeKey Unit:=wdStory

' Dummy copy to clear clipboard

Set rng = ActiveDocument.Content

rng.End = rng.Start + 1

rng.Copy

Beep

End Sub

**Sub SaveAsWithIndex()**

**'** Paul Beverley - Version 30.06.23

' Saves the current file, adding a postfix

switchTC\_ON = True

myPostfix = "\_PB\_01"

myName = ActiveDocument.FullName

dotPos = InStr(myName, ".do")

fType = Mid(myName, dotPos)

myNm = Replace(myName, fType, "")

myPostfix = myPostfix & ".doc"

num = Val(Right(myNm, 2))

If InStr(myNm, Left(myPostfix, 3)) = 0 Then

myResponse = MsgBox("Save the current file, adding a postfix?", \_

vbQuestion + vbYesNo, "SaveAsWithIndex")

If myResponse = vbNo Then Exit Sub

myName = Replace(myName, ".doc", myPostfix)

If switchTC\_ON Then ActiveDocument.TrackRevisions = True

Else

newNum = Trim(Str(num + 1))

If Len(newNum) = 1 Then newNum = "0" & newNum

myName = Left(myNm, Len(myNm) - 2) & newNum & fType

myResponse = MsgBox("Save the current file, with number " \_

& newNum & " ?", vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

End If

TestStr = ""

TestStr = Dir(myName)

If TestStr > "" Then

Beep

myResponse = MsgBox("Filename exists! Overwrite?", \_

vbQuestion + vbYesNo, "SaveAsWithIndex")

If myResponse = vbNo Then Exit Sub

End If

ActiveDocument.SaveAs fileName:=myName

End Sub

**Sub OpenmySize()**

' Paul Beverley - Version 02.11.21

' Opens the window to a particular size, position and magnification

myZoom = 170

myZoom = 200

extraLeft = 800

If ActiveDocument.ActiveWindow.WindowState = wdWindowStateMaximize Then

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Move Left:=1440 - extraLeft, Top:=2

Application.Resize Width:=1440 + extraLeft, Height:=800

ActiveWindow.ActivePane.View.Zoom.Percentage = myZoom

Else

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Move Left:=1440, Top:=2

Application.Resize Width:=1440, Height:=800

ActiveDocument.ActiveWindow.WindowState = wdWindowStateMaximize

ActiveWindow.ActivePane.View.Zoom.Percentage = myZoom

End If

With ActiveWindow.View.RevisionsFilter

.Markup = wdRevisionsMarkupAll

.View = wdRevisionsViewFinal

End With

End Sub

**Sub** **MultiFileAcceptTrackChanges()**

' Paul Beverley - Version 21.10.21

' Accept track changes in multiple files

alsoRemoveComments = False

If alsoRemoveComments = True Then

Beep

myResponse = MsgBox("Also remove comments?!", vbQuestion \_

+ vbYesNoCancel, "MultiFileAcceptTrackChanges")

If myResponse = vbCancel Then Exit Sub

If myResponse = vbNo Then alsoRemoveComments = False

End If

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Accept track changes on ALL the files in this folder:" & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "Multifile FRedit")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Accept track changes on just the files listed here?", \_

vbQuestion + vbYesNoCancel, "Multifile FRedit")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

Beep

myResponse = MsgBox("Are you SURE you want to accept track changes on all" & Str(numFiles) \_

& " files?" & vbCr & vbCr & "There's no going back, you know!", \_

vbQuestion + vbYesNoCancel, "WARNING! WARNING! WARNING!")

If myResponse <> vbYes Then Exit Sub

For i = 1 To numFiles

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set myDoc = Application.Documents.Open(FileName:=thisFile)

StatusBar = allMyFiles(i)

DoEvents

ActiveDocument.AcceptAllRevisions

numberCmnts = ActiveDocument.Comments.Count

If numberCmnts > 0 And alsoRemoveComments = True \_

Then ActiveDocument.DeleteAllComments

myDoc.Save

ActiveWindow.Close

Next

Selection.HomeKey Unit:=wdStory

MsgBox ("Tracks accepted on " + Str(numFiles) + " files")

End Sub

**Sub AddWordToStyleList()**

' Paul Beverley - Version 26.05.12

' Adds the selected text to the style list file

listName = "StyleSheet"

listName2 = "WordList"

withFormatting = True

myFont = "Calibri"

If Selection.Start = Selection.End Then

Selection.Words(1).Select

End If

myText = Selection

If withFormatting = True Then Selection.Copy

Set mainDoc = ActiveDocument

' Look for the list file

gotList = False

For i = 1 To Application.Windows.Count

If InStr(Application.Windows(i).Document.Name, listName) > 0 Or \_

InStr(Application.Windows(i).Document.Name, listName2) > 0 Then

Set listDoc = Application.Windows(i).Document

gotList = True

End If

Next

' Go to list and type word(s)

If gotList = True Then

listDoc.Activate

If withFormatting = True Then

Selection.Paste

Else

Selection.TypeText Text:=myText

Selection.Font.Name = myFont

End If

Selection.TypeText Text:=vbCrLf

mainDoc.Activate

Else

MsgBox "Sorry, can't find the style list."

Exit Sub

End If

End Sub

**Sub MultiFilePDF()**

' Paul Beverley - Version 13.12.17

' Save a folder full of files as PDFs

showTrackChanges = True

pdfFilesInFolder = True

pdfFolderName = "PDF"

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

If pdfFilesInFolder = True Then

mySubFolder = "\" & pdfFolderName

Else

mySubFolder = ""

End If

myResponse = MsgBox("Save PDFs of ALL the files in this folder: " & mySubFolder & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "Multifile PDF Writer")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

If pdfFilesInFolder = True Then

myResponse = MsgBox("Save PDFs of just the files listed here, in this folder: " & \_

mySubFolder & myFolder & " ?", \_

vbQuestion + vbYesNoCancel, "Multifile PDF Writer")

If myResponse <> vbYes Then Exit Sub

End If

For i = 1 To numFiles

thisName = allMyFiles(i)

myFileType = Mid(thisName, InStr(thisName, "."))

justName = Replace(thisName, myFileType, "")

thisFile = myFolder & myDelimiter & thisName

Set myDoc = Application.Documents.Open(FileName:=thisFile)

StatusBar = allMyFiles(i)

If pdfFilesInFolder = True Then

justName = pdfFolderName & "\" & justName

End If

DoEvents

If showTrackChanges = True Then

ActiveDocument.ExportAsFixedFormat OutputFileName:=justName, \_

ExportFormat:=wdExportFormatPDF, Item:=wdExportDocumentWithMarkup, \_

OpenAfterExport:=False

Else

ActiveDocument.ExportAsFixedFormat OutputFileName:=justName, \_

ExportFormat:=wdExportFormatPDF, OpenAfterExport:=False

End If

myDoc.Close SaveChanges:=wdDoNotSaveChanges

Next

Selection.HomeKey Unit:=wdStory

' Dummy copy to clear clipboard

Set rng = ActiveDocument.Content

rng.End = rng.Start + 1

rng.Copy

MsgBox (numFiles & " files saved as pdf")

End Sub

**Sub MultiFileShowHiddenText()**

' Paul Beverley - Version 25.08.20

' Unhides hidden text in multiple files

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Accept track changes on ALL the files in this folder:" & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "Multifile FRedit")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Accept track changes on just the files listed here?", \_

vbQuestion + vbYesNoCancel, "Multifile FRedit")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

Beep

myResponse = MsgBox("Are you SURE you want to show hidden text on all" & Str(numFiles) \_

& " files?" & vbCr & vbCr & "There's no going back, you know!", \_

vbQuestion + vbYesNoCancel, "WARNING! WARNING! WARNING!")

If myResponse <> vbYes Then Exit Sub

For i = 1 To numFiles

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set myDoc = Application.Documents.Open(FileName:=thisFile)

StatusBar = allMyFiles(i)

DoEvents

Set rng = ActiveDocument.Content

rng.Font.Hidden = False

myDoc.Save

ActiveWindow.Close

Next

Selection.HomeKey Unit:=wdStory

MsgBox ("Tracks accepted on " + Str(numFiles) + " files")

End Sub

**Sub DoNowt()**

**'** Paul Beverley - Version 01.02.10

' Do nothing at all!

Beep

End Sub

**Sub OvertypeBeep()**

**'** Paul Beverley - Version 01.01.10

' Sound warning beep on overtype

Options.Overtype = Not Options.Overtype

beep

If Options.Overtype = True Then

Do

Loop Until Timer > myTime + 0.2

beep

End If

End Sub

**Sub OvertypeBeep2()**

**'** Paul Beverley - Version 01.01.10

' Sound warning beep on overtype + visual

Options.Overtype = Not Options.Overtype

beep

myTime = Timer

If Options.Overtype = True Then

StatusBar = String(100, "#")

Do

Loop Until Timer > myTime + 0.2

beep

End If

End Sub

**Sub** **ReverseList()**

' Paul Beverley - Version 05.09.20

' Reverses the order of items in a list

If Selection.Start <> Selection.End Then

Set rng = Selection.Range.Duplicate

Else

Beep

myResponse = MsgBox("Work on the whole file?", \_

vbQuestion + vbYesNoCancel, "ReverseList")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

End If

i = 999

For Each myPara In rng.Paragraphs

myPara.Range.InsertBefore Text:=Trim(Str(i)) & " "

i = i - 1

Next myPara

rng.Sort SortOrder:=wdSortOrderAscending

For Each myPara In rng.Paragraphs

myPara.Range.Words(1).Delete

Next myPara

End Sub

**Sub PictureShow()**

**'** Paul Beverley - Version 31.08.11

' Switch pictures on and off

ActiveWindow.View.ShowPicturePlaceHolders = \_

Not (ActiveWindow.View.ShowPicturePlaceHolders)

End Sub

**Sub HideShowPictures()**

**'** Paul Beverley - Version 31.08.11

' Switch pictures on and off (Was called PictureShow)

ActiveWindow.View.ShowPicturePlaceHolders = \_

Not (ActiveWindow.View.ShowPicturePlaceHolders)

End Sub

**Sub SpecialCharList()**

**'** Paul Beverley - Version 28.02.10

' Creates a list of the Unicode characters in the document

Dim myFileSystem, myFileList, myFile, myFileType As String

Dim myDoc As Document

myChars = ""

mySymbols = ""

myResponse = MsgBox("Greek character collector" & vbCrLf & \_

"Multiple files?", vbQuestion + vbYesNo)

If myResponse = vbNo Then myFile = "": GoTo oneFile

myFolder = ActiveDocument.Path

ActiveDocument.Close SaveChanges:=False

Set myFileSystem = CreateObject("Scripting.FileSystemObject")

Set myFileList = myFileSystem.GetFolder(myFolder).Files

FilesTotal = 0

For Each myFile In myFileList

myFileType = Right(myFile, 4)

If (myFileType = ".doc" Or myFileType = "docx") And \_

Left(myFile, 1) <> "~" Then

StatusBar = myFile

Set myDoc = Application.Documents.Open(FileName:=myFile.Path, \_

ReadOnly:=True)

oneFile:

StatusBar = " Counting ..."

For Each myChr In ActiveDocument.Range.Characters

If AscW(myChr) >= 255 Then

If InStr(myChars, myChr) = 0 Then myChars = myChars & myChr & vbCrLf

End If

If myChr.Font.Name = "Symbol" Then

If InStr(mySymbols, myChr) = 0 Then mySymbols = mySymbols & myChr & vbCrLf

End If

Next myChr

If myResponse = vbNo Then GoTo theEnd

myDoc.Close SaveChanges:=wdDoNotSaveChanges

End If

Next myFile

theEnd:

Documents.Add

Selection.TypeText Text:=myChars & vbCrLf

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCrLf & vbCrLf \_

& "Symbol fonts" & vbCrLf & vbCrLf

firstEnd = Selection.End

Selection.TypeText Text:=mySymbols & vbCrLf

Selection.Start = firstEnd

Selection.Sort SortOrder:=wdSortOrderAscending

Selection.Font.Name = "Symbol"

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Unicode fonts" & vbCrLf & vbCrLf

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13{1,}"

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

StatusBar = ""

Selection.HomeKey Unit:=wdStory

End Sub

**Sub KeystrokeLister()**

**'** Paul Beverley - Version 12.12.24

' Creates a list of custom key allocations

Dim myKeys As String

Dim KeyCat As String

Dim Cmnd As String

Dim kb As KeyBinding

myKeys = "KeyString" & vbTab & "Category" \_

& vbTab & "Command" & vbCrLf

Documents.Add

Selection.InsertAfter myKeys

Selection.Collapse wdCollapseEnd

For Each kb In KeyBindings

Select Case kb.KeyCategory

Case 0: KeyCat = "Disable"

Case 1: KeyCat = "Command"

Case 2: KeyCat = "Macro"

Case 3: KeyCat = "Font"

Case 4: KeyCat = "AutoText"

Case 5: KeyCat = "Style"

Case 6: KeyCat = "Symbol"

Case 7: KeyCat = "Prefix"

End Select

If KeyCat <> "Disable" Then

Cmnd = Replace(kb.Command, "Normal.NewMacros.", "Nml.")

myKeys = kb.KeyString & vbTab & KeyCat \_

& vbTab & Cmnd & vbCr

End If

Selection.InsertAfter myKeys

Selection.Collapse wdCollapseEnd

Next kb

ActiveDocument.Paragraphs(1).Range.Font.Bold = True

Set rng = ActiveDocument.Content

Set tbl = rng.ConvertToTable(Separator:=wdSeparateByTabs)

tbl.Style = "Table Grid"

tbl.Columns.AutoFit

tbl.Sort ExcludeHeader:=True, FieldNumber:="Column 3"

' Now create a second copy of the list, but sorted by command

Selection.HomeKey Unit:=wdStory

Selection.InsertBreak Type:=wdPageBreak

Set rng = ActiveDocument.Content

rng.MoveStart , 1

Set rng2 = ActiveDocument.Content

rng2.Collapse wdCollapseStart

rng2.FormattedText = rng.FormattedText

ActiveDocument.Tables(1).Sort ExcludeHeader:=True, \_

FieldNumber:="Column 1"

Selection.HomeKey Unit:=wdStory

End Sub

**Sub MacrosAllRestore()**

' Paul Beverley - Version 18.11.19

' Adds keybindings from a list of macro names

' Keybindings start

' Keybindings end

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Sub Macros" & "AllRestore()"

.MatchCase = False

.MatchWildcards = False

.Execute

End With

If Selection.Find.Found = False Then

MsgBox "Can't find Sub Macros" & "AllRestore" & vbCr & vbCr \_

& "Is this a macro backup file?"

Exit Sub

End If

Selection.Collapse wdCollapseEnd

With Selection.Find

.Text = "Keybind" & "ings start"

.Execute

End With

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

bindersStart = Selection.Start

With Selection.Find

.Text = "Keybind" & "ings end"

.Execute

End With

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Selection.MoveEnd , -1

Selection.Start = bindersStart

numKeys = 0

Set myList = Selection.range.Duplicate

For Each myPara In myList.Paragraphs

myLine = myPara.range.Text

myLine = Replace(myLine, vbCr, "")

If Len(myLine) > 1 And InStr(myLine, ":") > 0 Then

colonPos = InStr(myLine, ":")

myMacroName = Trim(Mid(myLine, 2, colonPos - 2))

ks = Mid(myLine, colonPos + 3)

kCode = 0

If InStr(ks, "Ctrl+") > 0 Then

kCode = kCode + 512

ks = Replace(ks, "Ctrl+", "")

End If

If InStr(ks, "Alt+") > 0 Then

kCode = kCode + 1024

ks = Replace(ks, "Alt+", "")

End If

If InStr(ks, "Shift+") > 0 Then

kCode = kCode + 256

ks = Replace(ks, "Shift+", "")

End If

' Ordinary capital letters

If ks Like "[A-Z]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' Ordinary numbers

If ks Like "[0-9]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' F keys

If Left(ks, 1) = "F" And Len(ks) > 1 Then

aCode = 111 + Val(Replace(ks, "F", ""))

ks = ""

End If

Select Case ks

Case "!": aCode = wdKey1: kCode = kCode + 256 ' Shifted number 1

Case """": aCode = wdKey2: kCode = kCode + 256 ' Shifted number 2

Case ChrW(163): aCode = wdKey3: kCode = kCode + 256 ' Shifted number 3

Case "$": aCode = wdKey4: kCode = kCode + 256 ' Shifted number 4

Case "%": aCode = wdKey5: kCode = kCode + 256 ' Shifted number 5

Case "^": aCode = wdKey6: kCode = kCode + 256 ' Shifted number 6

Case "&": aCode = wdKey7: kCode = kCode + 256 ' Shifted number 7

Case "\*": aCode = wdKey8: kCode = kCode + 256 ' Shifted number 8

Case "(": aCode = wdKey9: kCode = kCode + 256 ' Shifted number 9

Case ")": aCode = wdKey0: kCode = kCode + 256 ' Shifted number 0

Case "'": aCode = 192

Case "-": aCode = wdKeyHyphen ' The - key

Case "#": aCode = 222

Case ",": aCode = wdKeyComma ' The , key

Case ".": aCode = wdKeyPeriod ' The key

Case "/": aCode = wdKeySlash

Case ":": aCode = wdKeySemiColon: kCode = kCode + 256

Case ";": aCode = wdKeySemiColon ' The ; key

Case "?": aCode = wdKeySlash: kCode = kCode + 256

Case "@": aCode = 192: kCode = kCode + 256

Case "[": aCode = wdKeyOpenSquareBrace ' The ] key

Case "\": aCode = wdKeyBackSlash ' The \ key

Case "]": aCode = wdKeyCloseSquareBrace ' The ] key

Case "\_": aCode = wdKeyHyphen ' Does it need? kCode = kCode + 256

Case "`": aCode = 223

Case "{": aCode = wdKeyOpenSquareBrace: kCode = kCode + 256

Case "}": aCode = wdKeyCloseSquareBrace: kCode = kCode + 256

Case "~": aCode = 222: kCode = kCode + 256

Case "+": aCode = wdKeyEquals ' kCode = kCode + 256

Case "<": aCode = wdKeyComma: kCode = kCode + 256

Case "=": aCode = wdKeyEquals ' The = key

Case ">": aCode = wdKeyPeriod: kCode = kCode + 256

Case "Backspace": aCode = wdKeyBackspace ' The Backspace key

Case "Clear (Num 5)": aCode = wdKeyNumeric5: kCode = kCode + 256

Case "Delete": aCode = wdKeyDelete ' The Delete key

Case "Down": aCode = 40 ' The Down Cursor key

Case "End": aCode = 35 ' The EndD key

Case "Home": aCode = wdKeyHome ' The HOME key

Case "Insert": aCode = wdKeyInsert ' The Insert key

Case "Left": aCode = 37 ' The Left Cursor key

Case "Num -": aCode = wdKeyNumericSubtract ' The - key on the keypad

Case "Num \*": aCode = wdKeyNumericMultiply ' The \* key on the keypad

Case "Num /": aCode = wdKeyNumericDivide ' The / key on the keypad

Case "Num +": aCode = wdKeyNumericAdd ' The + key on the keypad

Case "Num 0": aCode = wdKeyNumeric0 ' The 0 key on the keypad

Case "Num 1": aCode = wdKeyNumeric1 ' The 1 key on the keypad

Case "Num 2": aCode = wdKeyNumeric2 ' The 2 key on the keypad

Case "Num 3": aCode = wdKeyNumeric3 ' The 3 key on the keypad

Case "Num 4": aCode = wdKeyNumeric4 ' The 4 key on the keypad

Case "Num 5": aCode = wdKeyNumeric5 ' The 5 key on the keypad

Case "Num 6": aCode = wdKeyNumeric6 ' The 6 key on the keypad

Case "Num 7": aCode = wdKeyNumeric7 ' The 7 key on the keypad

Case "Num 8": aCode = wdKeyNumeric8 ' The 8 key on the keypad

Case "Num 9": aCode = wdKeyNumeric9 ' The 9 key on the keypad

Case "Page Down": aCode = 34 ' The Page Down key

Case "Page Up": aCode = 33 ' The Page Up key

Case "Return": aCode = wdKeyReturn ' The Enter or Return key

Case "Right": aCode = 39 ' The Right Cursor key

Case "Space": aCode = wdKeySpacebar ' The Spacebar key

Case "Up": aCode = 38 ' The Up Cursor key

End Select

kCode = kCode + aCode

hexCode = Replace(Hex(kCode), "FFFF", "")

KeyBindings.Add KeyCode:=kCode, \_

KeyCategory:=wdKeyCategoryMacro, Command:=myMacroName

numKS = numKS + 1

End If

Next myPara

MsgBox ("Restored: " & numKS & " macro key assignments")

End Sub

**Sub MacrosAllBackup()**

' Paul Beverley - Version 18.11.19

' Creates a list of all macro keystrokes

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Sub Macros" & "AllRestore()"

.MatchCase = False

.MatchWildcards = False

.Execute

End With

CR = vbCr

CR2 = CR & CR

If Selection.Find.Found = False Then

MsgBox "Can't find Sub Macros" & "AllRestore" & CR & CR \_

& "Please copy your macros into this file."

Exit Sub

End If

Selection.Collapse wdCollapseEnd

With Selection.Find

.Text = "Keybind" & "ings start"

.Execute

End With

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

bindersStart = Selection.Start

With Selection.Find

.Text = "Keybind" & "ings end"

.Execute

End With

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Selection.MoveEnd , -1

Selection.Start = bindersStart

Selection.Delete

Selection.TypeText Text:=CR

numKeys = 0

For Each kb In KeyBindings

If kb.KeyCategory = 2 Then ' It's a Macro

cmd = kb.Command

If Left(cmd, 6) = "Normal" Then

cmd = Replace(cmd, "Normal.NewMacros.", "")

Selection.TypeText Text:="' " & cmd & ": " & kb.KeyString & CR

numKeys = numKeys + 1

End If

End If

Next kb

Selection.Start = bindersStart

Selection.Sort SortOrder:=wdSortOrderAscending

DoEvents

myTot = ActiveDocument.range.End

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "End" & " Sub"

.Replacement.Text = "^&!"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

numMacros = ActiveDocument.range.End - myTot

If numMacros > 0 Then WordBasic.EditUndo

Beep

MsgBox "Macro key assignments saved: " & numKeys & CR

Selection.HomeKey Unit:=wdStory

m = Month(Now)

mn = Trim(Str(m))

If m < 10 Then mn = "0" & mn

d = Day(Now)

dt = Trim(Str(d))

If d < 10 Then dt = "0" & dt

myPrompt = "' Macro backup " & Year(Date) & " " & mn \_

& " " & dt & CR2

myPrompt = myPrompt & "' Macros saved: " & numMacros & CR

myPrompt = myPrompt & "' Keybindings saved: " & numKeys & CR2

Selection.TypeText Text:=myPrompt

End Sub

**Sub MacrosAllRestoreMac()**

' Paul Beverley - Version 03.02.18

' Adds keybindings from a list of macro names

' Keybindings start

' Keybindings end

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Sub Macros" & "AllRestore()"

.MatchCase = False

.MatchWildcards = False

.Execute

End With

If Selection.Find.Found = False Then

MsgBox "Can't find Sub MacrosAllRestore" & vbCr & vbCr \_

& "Is this a macro backup file?"

Exit Sub

End If

Selection.Collapse wdCollapseEnd

With Selection.Find

.Text = "Keybind" & "ings start"

.Execute

End With

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

bindersStart = Selection.Start

With Selection.Find

.Text = "Keybind" & "ings end"

.Execute

End With

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Selection.MoveEnd , -1

Selection.Start = bindersStart

numKeys = 0

Set myList = Selection.range.Duplicate

For Each myPara In myList.Paragraphs

myLine = myPara.range.Text

myLine = Replace(myLine, vbCr, "")

If Len(myLine) > 1 And InStr(myLine, ":") > 0 Then

colonPos = InStr(myLine, ":")

myMacroName = Trim(Mid(myLine, 2, colonPos - 2))

ks = Mid(myLine, colonPos + 3)

kCode = 0

If InStr(ks, "Command+") > 0 Then

kCode = kCode + 256

ks = Replace(ks, "Command+", "")

End If

If InStr(ks, "Control+") > 0 Then

kCode = kCode + 4096

ks = Replace(ks, "Control+", "")

End If

If InStr(ks, "Option+") > 0 Then

kCode = kCode + 2048

ks = Replace(ks, "Option+", "")

End If

If InStr(ks, "Shift+") > 0 Then

kCode = kCode + 512

ks = Replace(ks, "Shift+", "")

End If

' Ordinary capital letters

If ks Like "[A-Z]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' Ordinary numbers

If ks Like "[0-9]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' F keys

If Left(ks, 1) = "F" And Len(ks) > 1 Then

aCode = 111 + Val(Replace(ks, "F", ""))

ks = ""

End If

Select Case ks

Case "!": aCode = wdKey1: kCode = kCode + 256 ' Shifted number 1

Case """": aCode = wdKey2: kCode = kCode + 256 ' Shifted number 2

Case ChrW(163): aCode = wdKey3: kCode = kCode + 256 ' Shifted number 3

Case "$": aCode = wdKey4: kCode = kCode + 256 ' Shifted number 4

Case "%": aCode = wdKey5: kCode = kCode + 256 ' Shifted number 5

Case "^": aCode = wdKey6: kCode = kCode + 256 ' Shifted number 6

Case "&": aCode = wdKey7: kCode = kCode + 256 ' Shifted number 7

Case "\*": aCode = wdKey8: kCode = kCode + 256 ' Shifted number 8

Case "(": aCode = wdKey9: kCode = kCode + 256 ' Shifted number 9

Case ")": aCode = wdKey0: kCode = kCode + 256 ' Shifted number 0

Case "'": aCode = 192

Case "-": aCode = wdKeyHyphen ' The - key

Case "#": aCode = 222

Case ",": aCode = wdKeyComma ' The , key

Case ".": aCode = wdKeyPeriod ' The key

Case "/": aCode = wdKeySlash

Case ":": aCode = wdKeySemiColon: kCode = kCode + 256

Case ";": aCode = wdKeySemiColon ' The ; key

Case "?": aCode = wdKeySlash: kCode = kCode + 256

Case "@": aCode = 192: kCode = kCode + 256

Case "[": aCode = wdKeyOpenSquareBrace ' The ] key

Case "\": aCode = wdKeyBackSlash ' The \ key

Case "]": aCode = wdKeyCloseSquareBrace ' The ] key

Case "\_": aCode = wdKeyHyphen ' Does it need? kCode = kCode + 256

Case "`": aCode = 223

Case "{": aCode = wdKeyOpenSquareBrace: kCode = kCode + 256

Case "}": aCode = wdKeyCloseSquareBrace: kCode = kCode + 256

Case "~": aCode = 222: kCode = kCode + 256

Case "+": aCode = wdKeyEquals ' kCode = kCode + 256

Case "<": aCode = wdKeyComma: kCode = kCode + 256

Case "=": aCode = wdKeyEquals ' The = key

Case ">": aCode = wdKeyPeriod: kCode = kCode + 256

Case "Backspace": aCode = wdKeyBackspace ' The Backspace key

Case "Clear (Num 5)": aCode = wdKeyNumeric5: kCode = kCode + 256

Case "Delete": aCode = wdKeyDelete ' The Delete key

Case "Down": aCode = 40 ' The Down Cursor key

Case "End": aCode = 35 ' The EndD key

Case "Home": aCode = wdKeyHome ' The HOME key

Case "Insert": aCode = wdKeyInsert ' The Insert key

Case "Left": aCode = 37 ' The Left Cursor key

Case "Num -": aCode = wdKeyNumericSubtract ' The - key on the keypad

Case "Num \*": aCode = wdKeyNumericMultiply ' The \* key on the keypad

Case "Num /": aCode = wdKeyNumericDivide ' The / key on the keypad

Case "Num +": aCode = wdKeyNumericAdd ' The + key on the keypad

Case "Num 0": aCode = wdKeyNumeric0 ' The 0 key on the keypad

Case "Num 1": aCode = wdKeyNumeric1 ' The 1 key on the keypad

Case "Num 2": aCode = wdKeyNumeric2 ' The 2 key on the keypad

Case "Num 3": aCode = wdKeyNumeric3 ' The 3 key on the keypad

Case "Num 4": aCode = wdKeyNumeric4 ' The 4 key on the keypad

Case "Num 5": aCode = wdKeyNumeric5 ' The 5 key on the keypad

Case "Num 6": aCode = wdKeyNumeric6 ' The 6 key on the keypad

Case "Num 7": aCode = wdKeyNumeric7 ' The 7 key on the keypad

Case "Num 8": aCode = wdKeyNumeric8 ' The 8 key on the keypad

Case "Num 9": aCode = wdKeyNumeric9 ' The 9 key on the keypad

Case "Page Down": aCode = 34 ' The Page Down key

Case "Page Up": aCode = 33 ' The Page Up key

Case "Return": aCode = wdKeyReturn ' The Enter or Return key

Case "Right": aCode = 39 ' The Right Cursor key

Case "Space": aCode = wdKeySpacebar ' The Spacebar key

Case "Up": aCode = 38 ' The Up Cursor key

End Select

kCode = kCode + aCode

hexCode = Replace(Hex(kCode), "FFFF", "")

KeyBindings.Add KeyCode:=kCode, \_

KeyCategory:=wdKeyCategoryMacro, Command:=myMacroName

numKS = numKS + 1

End If

Next myPara

MsgBox ("Restored: " & numKS & " macro key assignments")

End Sub

**Sub KeystrokesMacroSave()**

' Paul Beverley - Version 13.07.17

' Creates a list of all macro keystrokes

Documents.Add

For Each kb In KeyBindings

If kb.KeyCategory = 2 Then ' It's a Macro

cmd = kb.Command

If Left(cmd, 6) = "Normal" Then

Selection.TypeText Text:=cmd & vbTab & kb.KeyString & vbCr

End If

End If

Next kb

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

Selection.HomeKey Unit:=wdStory

n = ActiveDocument.Paragraphs.Count

Beep

Selection.TypeText Text:="All macro key assignments" & \_

vbCr & "Saved: " & n - 1 & vbCr

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Normal.NewMacros."

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Color = wdColorGray25

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub KeystrokesMacroRestore()**

' Paul Beverley - Version 18.07.17

' Applies keystrokes from a list of macros and keystrokes

numKS = 0

For Each myPara In ActiveDocument.Paragraphs

myLine = myPara.range.Text

myLine = Replace(myLine, vbCr, "")

If Len(myLine) > 1 And InStr(myLine, ".") > 0 Then

tabPos = InStr(myLine, vbTab)

myMacroName = Left(myLine, tabPos - 1)

ks = Mid(myLine, tabPos + 1)

kCode = 0

If InStr(ks, "Ctrl+") > 0 Then

kCode = kCode + 512

ks = Replace(ks, "Ctrl+", "")

End If

If InStr(ks, "Alt+") > 0 Then

kCode = kCode + 1024

ks = Replace(ks, "Alt+", "")

End If

If InStr(ks, "Shift+") > 0 Then

kCode = kCode + 256

ks = Replace(ks, "Shift+", "")

End If

' Ordinary capital letters

If ks Like "[A-Z]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' Ordinary numbers

If ks Like "[0-9]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' F keys

If Left(ks, 1) = "F" And Len(ks) > 1 Then

aCode = 111 + Val(Replace(ks, "F", ""))

ks = ""

End If

Select Case ks

Case "!": aCode = wdKey1: kCode = kCode + 256 ' Shifted number 1

Case """": aCode = wdKey2: kCode = kCode + 256 ' Shifted number 2

Case ChrW(163): aCode = wdKey3: kCode = kCode + 256 ' Shifted number 3

Case "$": aCode = wdKey4: kCode = kCode + 256 ' Shifted number 4

Case "%": aCode = wdKey5: kCode = kCode + 256 ' Shifted number 5

Case "^": aCode = wdKey6: kCode = kCode + 256 ' Shifted number 6

Case "&": aCode = wdKey7: kCode = kCode + 256 ' Shifted number 7

Case "\*": aCode = wdKey8: kCode = kCode + 256 ' Shifted number 8

Case "(": aCode = wdKey9: kCode = kCode + 256 ' Shifted number 9

Case ")": aCode = wdKey0: kCode = kCode + 256 ' Shifted number 0

Case "'": aCode = 192

Case "-": aCode = wdKeyHyphen ' The - key

Case "#": aCode = 222

Case ",": aCode = wdKeyComma ' The , key

Case ".": aCode = wdKeyPeriod ' The key

Case "/": aCode = wdKeySlash

Case ":": aCode = wdKeySemiColon: kCode = kCode + 256

Case ";": aCode = wdKeySemiColon ' The ; key

Case "?": aCode = wdKeySlash: kCode = kCode + 256

Case "@": aCode = 192: kCode = kCode + 256

Case "[": aCode = wdKeyOpenSquareBrace ' The ] key

Case "\": aCode = wdKeyBackSlash ' The \ key

Case "]": aCode = wdKeyCloseSquareBrace ' The ] key

Case "\_": aCode = wdKeyHyphen ' Does it need? kCode = kCode + 256

Case "`": aCode = 223

Case "{": aCode = wdKeyOpenSquareBrace: kCode = kCode + 256

Case "}": aCode = wdKeyCloseSquareBrace: kCode = kCode + 256

Case "~": aCode = 222: kCode = kCode + 256

Case "+": aCode = wdKeyEquals ' kCode = kCode + 256

Case "<": aCode = wdKeyComma: kCode = kCode + 256

Case "=": aCode = wdKeyEquals ' The = key

Case ">": aCode = wdKeyPeriod: kCode = kCode + 256

Case "Backspace": aCode = wdKeyBackspace ' The Backspace key

Case "Clear (Num 5)": aCode = wdKeyNumeric5: kCode = kCode + 256

Case "Delete": aCode = wdKeyDelete ' The Delete key

Case "Down": aCode = 40 ' The Down Cursor key

Case "End": aCode = 35 ' The EndD key

Case "Home": aCode = wdKeyHome ' The HOME key

Case "Insert": aCode = wdKeyInsert ' The Insert key

Case "Left": aCode = 37 ' The Left Cursor key

Case "Num -": aCode = wdKeyNumericSubtract ' The - key on the keypad

Case "Num \*": aCode = wdKeyNumericMultiply ' The \* key on the keypad

Case "Num /": aCode = wdKeyNumericDivide ' The / key on the keypad

Case "Num +": aCode = wdKeyNumericAdd ' The + key on the keypad

Case "Num 0": aCode = wdKeyNumeric0 ' The 0 key on the keypad

Case "Num 1": aCode = wdKeyNumeric1 ' The 1 key on the keypad

Case "Num 2": aCode = wdKeyNumeric2 ' The 2 key on the keypad

Case "Num 3": aCode = wdKeyNumeric3 ' The 3 key on the keypad

Case "Num 4": aCode = wdKeyNumeric4 ' The 4 key on the keypad

Case "Num 5": aCode = wdKeyNumeric5 ' The 5 key on the keypad

Case "Num 6": aCode = wdKeyNumeric6 ' The 6 key on the keypad

Case "Num 7": aCode = wdKeyNumeric7 ' The 7 key on the keypad

Case "Num 8": aCode = wdKeyNumeric8 ' The 8 key on the keypad

Case "Num 9": aCode = wdKeyNumeric9 ' The 9 key on the keypad

Case "Page Down": aCode = 34 ' The Page Down key

Case "Page Up": aCode = 33 ' The Page Up key

Case "Return": aCode = wdKeyReturn ' The Enter or Return key

Case "Right": aCode = 39 ' The Right Cursor key

Case "Space": aCode = wdKeySpacebar ' The Spacebar key

Case "Up": aCode = 38 ' The Up Cursor key

End Select

kCode = kCode + aCode

hexCode = Replace(Hex(kCode), "FFFF", "")

KeyBindings.Add KeyCode:=kCode, \_

KeyCategory:=wdKeyCategoryMacro, Command:=myMacroName

numKS = numKS + 1

End If

Next myPara

MsgBox ("Restored: " & numKS & " macro key assignments")

End Sub

**Sub KeystrokesSaveAll()**

' Paul Beverley - Version 19.07.17

' Creates a list of all user-defined keystrokes

Documents.Add

allKeys = ""

For Each kb In KeyBindings

myCatNum = kb.KeyCategory

If myCatNum > 0 And myCatNum <> wdKeyCategoryPrefix Then

Select Case myCatNum

Case wdKeyCategorySymbol: myCat = "Symbol"

Case wdKeyCategoryFont: myCat = "Font"

Case wdKeyCategoryStyle: myCat = "Style"

Case wdKeyCategoryCommand: myCat = "Command"

Case wdKeyCategoryMacro: myCat = "Macro"

Case Else: myCat = "Unknown!"

End Select

cmd = kb.Command

allKeys = allKeys & myCat & vbTab & cmd & vbTab & kb.KeyString & vbCr

End If

Next kb

Selection.TypeText Text:=allKeys

' Wait for repagination

' Set rng = ActiveDocument.GoTo(what:=wdGoToBookmark, Name:="\EndOfDoc")

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

' Wait for repagination

' Set rng = ActiveDocument.GoTo(what:=wdGoToBookmark, Name:="\EndOfDoc")

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Normal.NewMacros."

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Color = wdColorGray25

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "MathTypeCommands.UILib.MTCommand\_"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Color = wdColorGray25

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

n = ActiveDocument.Paragraphs.Count

Beep

Selection.TypeText Text:="All key assignments" & \_

vbCr & "Saved: " & n - 3 & vbCr

End Sub

**Sub KeystrokesRestoreAll()**

' Paul Beverley - Version 20.07.17

' Creates keybindings from a list

Beep

myResponse = MsgBox("KeystrokesRestoreAll:" & vbCr & vbCr & \_

"Please use this macro with great caution; it could" \_

& vbCr & "seriously mess up your keystroke allocations," \_

& vbCr & "if you're not careful." & vbCr & vbCr & "Continue?", \_

vbQuestion + vbYesNoCancel, "KeystrokesRestoreAll")

If myResponse <> vbYes Then Exit Sub

On Error GoTo ReportIt

numKS = 0

nowPara = ActiveDocument.range(0, Selection.Paragraphs(1).range.End).Paragraphs.Count

If nowPara <> 1 Then

myResponse = MsgBox("Start from top of list?", vbQuestion + vbYesNo)

If myResponse = vbYes Then nowPara = 1

End If

For i = nowPara To ActiveDocument.Paragraphs.Count

Set myPara = ActiveDocument.Paragraphs(i).range.Duplicate

myLine = myPara.Text

If Left(myLine, 1) = "#" Then GoTo theEnd

myLine = Replace(myLine, vbCr, "")

tabPos = InStr(myLine, vbTab)

If Len(myLine) > 1 And tabPos > 0 Then

' First find the category

myCategory = Left(myLine, tabPos - 1)

myLine = Mid(myLine, tabPos + 1)

' Find the command and keystroke

tabPos = InStr(myLine, vbTab)

myItem = Left(myLine, tabPos - 1)

ks = Mid(myLine, tabPos + 1)

kCode = 0

If InStr(ks, "Ctrl+") > 0 Then

kCode = kCode + 512

ks = Replace(ks, "Ctrl+", "")

End If

If InStr(ks, "Alt+") > 0 Then

kCode = kCode + 1024

ks = Replace(ks, "Alt+", "")

End If

If InStr(ks, "Shift+") > 0 Then

kCode = kCode + 256

ks = Replace(ks, "Shift+", "")

End If

' Ordinary capital letters

If ks Like "[A-Z]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' Ordinary numbers

If ks Like "[0-9]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' F keys

If Left(ks, 1) = "F" And Len(ks) > 1 Then

aCode = 111 + Val(Replace(ks, "F", ""))

ks = ""

End If

Select Case ks

Case "!": aCode = wdKey1: kCode = kCode + 256 ' Shifted number 1

Case """": aCode = wdKey2: kCode = kCode + 256 ' Shifted number 2

Case ChrW(163): aCode = wdKey3: kCode = kCode + 256 ' Shifted number 3

Case "$": aCode = wdKey4: kCode = kCode + 256 ' Shifted number 4

Case "%": aCode = wdKey5: kCode = kCode + 256 ' Shifted number 5

Case "^": aCode = wdKey6: kCode = kCode + 256 ' Shifted number 6

Case "&": aCode = wdKey7: kCode = kCode + 256 ' Shifted number 7

Case "\*": aCode = wdKey8: kCode = kCode + 256 ' Shifted number 8

Case "(": aCode = wdKey9: kCode = kCode + 256 ' Shifted number 9

Case ")": aCode = wdKey0: kCode = kCode + 256 ' Shifted number 0

Case "'": aCode = 192

Case "-": aCode = wdKeyHyphen ' The - key

Case "#": aCode = 222

Case ",": aCode = wdKeyComma ' The , key

Case ".": aCode = wdKeyPeriod ' The key

Case "/": aCode = wdKeySlash

Case ":": aCode = wdKeySemiColon: kCode = kCode + 256

Case ";": aCode = wdKeySemiColon ' The ; key

Case "?": aCode = wdKeySlash: kCode = kCode + 256

Case "@": aCode = 192: kCode = kCode + 256

Case "[": aCode = wdKeyOpenSquareBrace ' The ] key

Case "\": aCode = wdKeyBackSlash ' The \ key

Case "]": aCode = wdKeyCloseSquareBrace ' The ] key

Case "\_": aCode = wdKeyHyphen ' Does it need? kCode = kCode + 256

Case "`": aCode = 223

Case "{": aCode = wdKeyOpenSquareBrace: kCode = kCode + 256

Case "}": aCode = wdKeyCloseSquareBrace: kCode = kCode + 256

Case "~": aCode = 222: kCode = kCode + 256

Case "+": aCode = wdKeyEquals ' kCode = kCode + 256

Case "<": aCode = wdKeyComma: kCode = kCode + 256

Case "=": aCode = wdKeyEquals ' The = key

Case ">": aCode = wdKeyPeriod: kCode = kCode + 256

Case "Backspace": aCode = wdKeyBackspace ' The Backspace key

Case "Clear (Num 5)": aCode = wdKeyNumeric5: kCode = kCode + 256

Case "Delete": aCode = wdKeyDelete ' The Delete key

Case "Down": aCode = 40 ' The Down Cursor key

Case "End": aCode = 35 ' The EndD key

Case "Home": aCode = wdKeyHome ' The HOME key

Case "Insert": aCode = wdKeyInsert ' The Insert key

Case "Left": aCode = 37 ' The Left Cursor key

Case "Num -": aCode = wdKeyNumericSubtract ' The - key on the keypad

Case "Num \*": aCode = wdKeyNumericMultiply ' The \* key on the keypad

Case "Num /": aCode = wdKeyNumericDivide ' The / key on the keypad

Case "Num +": aCode = wdKeyNumericAdd ' The + key on the keypad

Case "Num 0": aCode = wdKeyNumeric0 ' The 0 key on the keypad

Case "Num 1": aCode = wdKeyNumeric1 ' The 1 key on the keypad

Case "Num 2": aCode = wdKeyNumeric2 ' The 2 key on the keypad

Case "Num 3": aCode = wdKeyNumeric3 ' The 3 key on the keypad

Case "Num 4": aCode = wdKeyNumeric4 ' The 4 key on the keypad

Case "Num 5": aCode = wdKeyNumeric5 ' The 5 key on the keypad

Case "Num 6": aCode = wdKeyNumeric6 ' The 6 key on the keypad

Case "Num 7": aCode = wdKeyNumeric7 ' The 7 key on the keypad

Case "Num 8": aCode = wdKeyNumeric8 ' The 8 key on the keypad

Case "Num 9": aCode = wdKeyNumeric9 ' The 9 key on the keypad

Case "Page Down": aCode = 34 ' The Page Down key

Case "Page Up": aCode = 33 ' The Page Up key

Case "Return": aCode = wdKeyReturn ' The Enter or Return key

Case "Right": aCode = 39 ' The Right Cursor key

Case "Space": aCode = wdKeySpacebar ' The Spacebar key

Case "Up": aCode = 38 ' The Up Cursor key

End Select

Select Case myCategory

Case "Symbol": myCat = wdKeyCategorySymbol

Case "Font": myCat = wdKeyCategoryFont

Case "Style": myCat = wdKeyCategoryStyle

Case "Command": myCat = wdKeyCategoryCommand

Case "Macro": myCat = wdKeyCategoryMacro

Case Else: myCat = 0

End Select

If myCat = 0 Then

myPara.Select

Selection.Collapse wdCollapseStart

Beep

MsgBox ("Not valid keybinding, I don't think.")

Exit Sub

End If

kCode = kCode + aCode

KeyBindings.Add KeyCode:=kCode, \_

KeyCategory:=myCat, Command:=myItem

numKS = numKS + 1

End If

Next i

theEnd:

myPara.Select

Selection.Collapse wdCollapseStart

Beep

MsgBox ("Restored: " & numKS & " key assignments")

ReportIt:

If Err.Number = 5346 Then

myPara.Select

Selection.Collapse wdCollapseStart

Beep

MsgBox ("Couldn't find " & myCategory & ": " & myItem)

End If

End Sub

**Sub CustomKeys()**

**'** Paul Beverley - Version 03.02.10

' Open the Customize Keyboard dialogue box

With Dialogs(wdDialogToolsCustomizeKeyboard)

.Category = 2

.Show

End With

End Sub

**Sub MacroVersionChecker()**

' Paul Beverley - Version 18.11.19

' Checks Paul Beverley - Version dates of all your macros against MacroList

Set myListDoc = ActiveDocument

listIfUpToDate = True

Dim namePosn As Long

gotList = False

For Each myDoc In Documents

thisName = myDoc.Name

If InStr(thisName, "MacroList") > 0 Then

Set listRng = myDoc.Content

gotList = True

fullList = LCase(listRng.Text)

End If

Next myDoc

If gotList = False Then

Beep

myResponse = MsgBox("Download MacroList file from Paul's website?", \_

vbQuestion + vbYesNoCancel, "MacroVersionChecker")

If myResponse <> vbYes Then Exit Sub

Documents.Open FileName:="http://www.archivepub.co.uk/documents/MacroList"

Set listRng = ActiveDocument.Content

fullList = LCase(listRng.Text)

End If

myListDoc.Activate

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Sub [a-zA-Z]{1,}\(\)"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

tabs2 = vbTab & vbTab

myOutput = "1111zzzzZBlankLine" & tabs2 & vbCr & "2222zzzzZBlankLine" & \_

tabs2 & vbCr & "3333zzzzZBlankLine" & vbCr

Do While rng.Find.Found = True

myCount = myCount + 1

EndNow = rng.End

rng.Start = rng.Start + 4

rng.End = rng.End - 2

mcName = rng.Text

rng.Collapse wdCollapseEnd

rng.End = rng.End + 80

versPosn = InStr(rng.Text, "' Paul Beverley - Version ")

If versPosn > 0 Then

rng.Start = rng.Start + versPosn + 9

rng.End = rng.Start + 8

mcDate = rng.Text

Else

mcDate = "(Not Paul's?)"

End If

namePosn = InStr(fullList, LCase(mcName) & vbCr)

If namePosn > 0 Then

paulDate = Mid(fullList, namePosn + Len(mcName) + 2, 8)

Else

paulDate = "Not listed?"

End If

myNewItem = mcName & vbTab & mcDate & vbTab

If paulDate = "Not listed?" Then myNewItem = "3333" & myNewItem \_

& paulDate

If mcDate <> paulDate And (LCase(paulDate) = UCase(paulDate)) \_

Then myNewItem = "1111" & myNewItem & " (" \_

& paulDate & ")"

If mcDate = "(Not Paul's?)" Then myNewItem = "2222" & myNewItem

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

If mcDate <> paulDate Or listIfUpToDate = True Then \_

myOutput = myOutput & myNewItem & vbCr

If myCount Mod 25 = 0 Then Debug.Print myNewItem & \_

" " & Str(myCount)

Loop

Documents.Add

Selection.TypeText Text:=myOutput

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending

With rng.Find

.Text = "[123]{4}"

.Font.Italic = False

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.Text = "zzzzZBlankLine"

.Font.Italic = False

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Macros list"

startTable = Selection.End + 1

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

Selection.Start = startTable

Selection.End = ActiveDocument.range.End

Selection.ConvertToTable Separator:=wdSeparateByTabs

Set tb = ActiveDocument.Tables(1)

tb.AutoFitBehavior (wdAutoFitContent)

tb.Borders(wdBorderTop).LineStyle = wdLineStyleNone

tb.Borders(wdBorderLeft).LineStyle = wdLineStyleNone

tb.Borders(wdBorderBottom).LineStyle = wdLineStyleNone

tb.Borders(wdBorderRight).LineStyle = wdLineStyleNone

tb.Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

tb.Borders(wdBorderVertical).LineStyle = wdLineStyleNone

Selection.HomeKey Unit:=wdStory

Beep

MsgBox "Found: " & myCount & " macros"

End Sub

**Sub MacroUpdater()**

' Paul Beverley - Version 16.01.21

' Updates to the current macro text, preserving the keystroke

Selection.Expand wdParagraph

macroStart = Selection.Start

If Left(Selection, 3) <> "Sub" Then

myResponse = MsgBox("Please place the cursor in the 'Sub' line of the macro", \_

vbQuestion + vbOKCancel, "MacroUpdater")

Exit Sub

End If

Do While InStr("(", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.MoveEnd , -1

MacroName = Mid(Selection, 5)

myKeyString = ""

For Each kb In KeyBindings

If kb.KeyCategory = 2 Then ' It's a Macro

cmd = kb.Command

gottit = (InStr(cmd, "." & MacroName) > 0)

If Left(cmd, 6) = "Normal" And gottit And \_

Right(cmd, Len(MacroName)) = MacroName Then

myKeyString = kb.KeyString

Exit For

End If

End If

Next kb

Dim v As Variable

varsExist = False

For Each v In ActiveDocument.Variables

If v.Name = "macroName" Then varsExist = True: Exit For

Next v

If varsExist = False Then

ActiveDocument.Variables.Add "macroName", MacroName

ActiveDocument.Variables.Add "myKeyString", myKeyString

End If

Do

myText = InputBox("1: Copy this macro" & vbCr & "2: Restore keystroke", "MacroUpdater")

myNumber = Val(myText)

If myNumber = 0 Then Beep: Exit Sub

Loop Until myNumber = 1 Or myNumber = 2

If myNumber = 1 Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^pEnd S" & "ub^p"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Selection.Start = macroStart

Selection.Copy

Selection.MoveLeft , 1

Selection.MoveRight , 1

Selection.Expand wdParagraph

ActiveDocument.Variables("macroName") = MacroName

ActiveDocument.Variables("myKeyString") = myKeyString

Beep

MsgBox "New version of macro copied and ready to paste into VBA."

Exit Sub

End If

MacroName = ActiveDocument.Variables("macroName")

myKeyString = ActiveDocument.Variables("myKeyString")

myResponse = MsgBox("Set keystroke to: " & myKeyString & "?", \_

vbQuestion + vbYesNoCancel, "MacroUpdater")

If myResponse = vbYes Then

ks = myKeyString

kCode = 0

If InStr(ks, "Ctrl+") > 0 Then

kCode = kCode + 512

ks = Replace(ks, "Ctrl+", "")

End If

If InStr(ks, "Alt+") > 0 Then

kCode = kCode + 1024

ks = Replace(ks, "Alt+", "")

End If

If InStr(ks, "Shift+") > 0 Then

kCode = kCode + 256

ks = Replace(ks, "Shift+", "")

End If

' Ordinary capital letters

If ks Like "[A-Z]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' Ordinary numbers

If ks Like "[0-9]" Then

aCode = Asc(UCase(ks))

ks = ""

End If

' F keys

If Left(ks, 1) = "F" And Len(ks) > 1 Then

aCode = 111 + Val(Replace(ks, "F", ""))

ks = ""

End If

Select Case ks

Case "!": aCode = wdKey1: kCode = kCode + 256 ' Shifted number 1

Case """": aCode = wdKey2: kCode = kCode + 256 ' Shifted number 2

Case ChrW(163): aCode = wdKey3: kCode = kCode + 256 ' Shifted number 3

Case "$": aCode = wdKey4: kCode = kCode + 256 ' Shifted number 4

Case "%": aCode = wdKey5: kCode = kCode + 256 ' Shifted number 5

Case "^": aCode = wdKey6: kCode = kCode + 256 ' Shifted number 6

Case "&": aCode = wdKey7: kCode = kCode + 256 ' Shifted number 7

Case "\*": aCode = wdKey8: kCode = kCode + 256 ' Shifted number 8

Case "(": aCode = wdKey9: kCode = kCode + 256 ' Shifted number 9

Case ")": aCode = wdKey0: kCode = kCode + 256 ' Shifted number 0

Case "'": aCode = 192

Case "-": aCode = wdKeyHyphen ' The - key

Case "#": aCode = 222

Case ",": aCode = wdKeyComma ' The , key

Case ".": aCode = wdKeyPeriod ' The key

Case "/": aCode = wdKeySlash

Case ":": aCode = wdKeySemiColon: kCode = kCode + 256

Case ";": aCode = wdKeySemiColon ' The ; key

Case "?": aCode = wdKeySlash: kCode = kCode + 256

Case "@": aCode = 192: kCode = kCode + 256

Case "[": aCode = wdKeyOpenSquareBrace ' The ] key

Case "\": aCode = wdKeyBackSlash ' The \ key

Case "]": aCode = wdKeyCloseSquareBrace ' The ] key

Case "\_": aCode = wdKeyHyphen ' Does it need? kCode = kCode + 256

Case "`": aCode = 223

Case "{": aCode = wdKeyOpenSquareBrace: kCode = kCode + 256

Case "}": aCode = wdKeyCloseSquareBrace: kCode = kCode + 256

Case "~": aCode = 222: kCode = kCode + 256

Case "+": aCode = wdKeyEquals ' kCode = kCode + 256

Case "<": aCode = wdKeyComma: kCode = kCode + 256

Case "=": aCode = wdKeyEquals ' The = key

Case ">": aCode = wdKeyPeriod: kCode = kCode + 256

Case "Backspace": aCode = wdKeyBackspace ' The Backspace key

Case "Clear (Num 5)": aCode = wdKeyNumeric5: kCode = kCode + 256

Case "Delete": aCode = wdKeyDelete ' The Delete key

Case "Down": aCode = 40 ' The Down Cursor key

Case "End": aCode = 35 ' The EndD key

Case "Home": aCode = wdKeyHome ' The HOME key

Case "Insert": aCode = wdKeyInsert ' The Insert key

Case "Left": aCode = 37 ' The Left Cursor key

Case "Num -": aCode = wdKeyNumericSubtract ' The - key on the keypad

Case "Num \*": aCode = wdKeyNumericMultiply ' The \* key on the keypad

Case "Num /": aCode = wdKeyNumericDivide ' The / key on the keypad

Case "Num +": aCode = wdKeyNumericAdd ' The + key on the keypad

Case "Num 0": aCode = wdKeyNumeric0 ' The 0 key on the keypad

Case "Num 1": aCode = wdKeyNumeric1 ' The 1 key on the keypad

Case "Num 2": aCode = wdKeyNumeric2 ' The 2 key on the keypad

Case "Num 3": aCode = wdKeyNumeric3 ' The 3 key on the keypad

Case "Num 4": aCode = wdKeyNumeric4 ' The 4 key on the keypad

Case "Num 5": aCode = wdKeyNumeric5 ' The 5 key on the keypad

Case "Num 6": aCode = wdKeyNumeric6 ' The 6 key on the keypad

Case "Num 7": aCode = wdKeyNumeric7 ' The 7 key on the keypad

Case "Num 8": aCode = wdKeyNumeric8 ' The 8 key on the keypad

Case "Num 9": aCode = wdKeyNumeric9 ' The 9 key on the keypad

Case "Page Down": aCode = 34 ' The Page Down key

Case "Page Up": aCode = 33 ' The Page Up key

Case "Return": aCode = wdKeyReturn ' The Enter or Return key

Case "Right": aCode = 39 ' The Right Cursor key

Case "Space": aCode = wdKeySpacebar ' The Spacebar key

Case "Up": aCode = 38 ' The Up Cursor key

End Select

kCode = kCode + aCode

hexCode = Replace(Hex(kCode), "FFFF", "")

KeyBindings.Add KeyCode:=kCode, \_

KeyCategory:=wdKeyCategoryMacro, Command:=MacroName

End If

Beep

End Sub

**Sub ToCupdate()**

' Paul Beverley - Version 16.12.13

' Update (and customise) the table of contents

' Ctrl-Alt-F3

ActiveDocument.TablesOfContents(1).Update

Options.DefaultHighlightColorIndex = wdYellow

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

' Now selectively delete any lines in the ToC

' that you don't want.

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Text = "Macros for Editors by Paul Beverley, LCGI"

.Style = "Hyperlink"

.Replacement.Text = ""

.Execute Replace:=wdReplaceOne

End With

Selection.TypeBackspace

' This deletes some lines I don't want, from here...

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Stylesheet"

.Style = "Hyperlink"

.Replacement.Text = ""

.Execute

End With

myStart = Selection.Start

' ...to here

Selection.Start = Selection.End

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Notes to publisher"

.Style = "Hyperlink"

.Replacement.Text = ""

.Execute

End With

Selection.HomeKey Unit:=wdStory

' Extend the selection back to the first line

' that needs deleting.

Selection.Start = myStart

Selection.TypeBackspace

Selection.TypeBackspace

Selection.HomeKey Unit:=wdStory

With Selection.Find

.Text = "Textual Analysis"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdLine

With Selection.Find

.Text = "Pre-editing Tools"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdLine

With Selection.Find

.Text = "Editing: Text Change"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdLine

With Selection.Find

.Text = "Editing: Information"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdLine

With Selection.Find

.Text = "Editing: Highlighting"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdLine

With Selection.Find

.Text = "Editing: Navigation"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdLine

With Selection.Find

.Text = "Editing: Comment Handling"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdLine

With Selection.Find

.Text = "Editing: Track Changes"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdLine

With Selection.Find

.Text = "Other Tools"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdLine

With Selection.Find

.Text = "The Macros"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdLine

With Selection.Find

.Text = "Changes Log"

.Style = "Hyperlink"

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = oldFind

.Replacement.Text = oldReplace

End With

End Sub

**Sub MacroIndent()**

**'** Paul Beverley - Version 01.01.10

' Automatically indent a macro listing

' Indent step size:

sizeStep = 2

' For indenting the wrapped-over long lines:

bigStep = 5

' Do you want the whole Sub to End Sub to be indented?

subIndent = False

' Remove all existing indents

Set rng = ActiveDocument.Range

For i = 1 To 2

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "^13 {1,}([! ])"

.Wrap = wdFindContinue

.Replacement.Text = "^p\1"

.Execute Replace:=wdReplaceAll

End With

Next i

' Add new indents

myIndent = 0

maxLine = ActiveDocument.Paragraphs.Count

For i = 1 To maxLine

Set rng = ActiveDocument.Paragraphs(i).Range

myLine = rng

myLine = Left(myLine, Len(myLine) - 1)

jumpNow = 0

jumpNext = 0

If myLine = "Do" Then jumpNext = 1

If Left(myLine, 8) = "Do While" Then jumpNext = 1

If subIndent = True And Left(myLine, 4) = "Sub " Then jumpNext = 1

If Left(myLine, 8) = "Function" Then jumpNext = 1

If Left(myLine, 5) = "While" Then jumpNext = 1

If Left(myLine, 4) = "Else" Then jumpNext = 1: jumpNow = -1

If Left(myLine, 4) = "With" Then jumpNext = 1

If Right(myLine, 4) = "Then" Then jumpNext = 1

If Left(myLine, 3) = "For" Then jumpNext = 1

If Left(myLine, 11) = "Select Case" Then jumpNext = 1

If Left(myLine, 4) = "End " Then jumpNow = -1

If Left(myLine, 4) = "Wend" Then jumpNow = -1

If subIndent = False And Left(myLine, 7) = "End" & " Sub" Then jumpNow = 0

If Left(myLine, 4) = "Loop" Then jumpNow = -1

If Left(myLine, 4) = "Next" Then jumpNow = -1

wasIndenting = indenting

indenting = (Right(myLine, 1) = "\_")

extrabit = 0

If wasIndenting = False And indenting = True Then extrabit = bigStep

If wasIndenting = True And indenting = False Then extrabit = -bigStep

myIndent = myIndent + sizeStep \* jumpNow

If myIndent > 0 Then rng.InsertBefore String(myIndent, " ")

myIndent = myIndent + sizeStep \* jumpNext + extrabit

StatusBar = "Lines to go: " & maxLine - i

If myIndent < 0 Then

rng.Select

MsgBox ("Structure error?")

Exit Sub

End If

Next i

StatusBar = ""

Set rng = ActiveDocument.Range

' Empty the blank lines of spaces

For i = 1 To 8

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = " ^p"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

Next i

End Sub

**Sub WikiSwitch()**

**'** Paul Beverley - Version 01.01.10

' Convert text to and from Wiki format

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "=" & "=^p"

.Execute

End With

If rng.Find.Found = True Then

Call WikiToStyles

Else

Call WikiToText

End If

Selection.HomeKey Unit:=wdStory

End Sub

**Sub WikiToText()**

**'** Paul Beverley - Version 01.01.10

' Convert from text to Wiki format

Dim Eq2, Eq3, Tk2, Tk3, pr, npr As String

Eq2 = "=" & "=": Eq3 = Eq2 + "="

Tk2 = "'" & "'": Tk3 = Tk2 + "'"

pr = "pre": npr = "</" & pr & ">"

pr = "<" & pr & ">"

' Convert Heading 2 to <=><=><=>Title<=><=><=>

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "(\*)^13"

.Style = "Heading 2"

.Replacement.Text = Eq3 & "\1" & Eq3 & "^p"

.Replacement.Style = wdStyleNormal

.Execute Replace:=wdReplaceAll

End With

' Convert Heading 1 to <=><=>Title<=><=>

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "(\*)^13"

.Format = True

.Style = "Heading 1"

.Replacement.Text = Eq2 & "\1" & Eq2 & "^p"

.Replacement.Style = wdStyleNormal

.Execute Replace:=wdReplaceAll

End With

' Convert Bold to <'><'><'>word<'><'><'>

more = True

Do

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Bold = True

.Execute

End With

more = rng.Find.Found

If more = True Then

rng.Font.Bold = False

rng.InsertBefore Tk3

rng.InsertAfter Tk3

End If

Loop Until more = False

' Convert Italic to <'><'>word<'><'>

more = True

Do

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Execute

End With

more = rng.Find.Found

If more = True Then

rng.Font.Italic = False

rng.InsertBefore Tk2

rng.InsertAfter Tk2

End If

Loop Until more = False

' Convert HTML sample to [pre]word[/pre]

gogo = True

Do

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Style = "HTML Sample"

.Execute

End With

myStart = rng.Start

myEnd = rng.End

gogo = rng.Find.Found

If gogo = True Then

Set rng = ActiveDocument.Range

rng.Start = myEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Style = "Default Paragraph Font"

.Execute

End With

myEnd = rng.Start

rng.Start = myStart

rng.End = myEnd

rng.Style = "Default Paragraph Font"

rng.InsertBefore pr & vbCrLf

rng.InsertAfter npr & vbCrLf

End If

Loop Until gogo = False

' Remove blank lines above an npr

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = "^p^p" & npr

.Replacement.Text = "^p" & npr & "^p"

.Execute Replace:=wdReplaceAll

End With

' Add second blank lines after an npr before a heading

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = npr & "^p^p"

.Replacement.Text = npr & "^p^p^p=="

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub WikiToStyles()**

**'** Paul Beverley - Version 01.01.10

' Convert from Wiki format to text

Dim Eq2, Eq3 As String

Eq2 = "=" & "=": Eq3 = Eq2 + "="

Tk2 = "'" & "'": Tk3 = Tk2 + "'"

pr = "pre": npr = "</" & pr & ">"

pr = "<" & pr & ">"

' Convert <=><=><=>Title<=><=><=> to Heading 2

Dim rng As Range

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = Eq3 & "(\*)" & Eq3 & "^13"

.Replacement.Text = "\1^p"

.Replacement.Style = "Heading 2"

.Execute Replace:=wdReplaceAll

End With

' Convert <=><=>Title<=><=> to Heading 1

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = Eq2 & "(\*)" & Eq2 & "^13"

.Replacement.Text = "\1^p"

.Replacement.Style = "Heading 1"

.Execute Replace:=wdReplaceAll

End With

' Convert <'><'><'>word<'><'><'> to Bold

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = Tk3 & "(\*)" & Tk3

.Replacement.Text = "\1"

.Replacement.Font.Bold = True

.Execute Replace:=wdReplaceAll

End With

' Convert <'><'>word<'><'> to Italic

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = Tk2 & "(\*)" & Tk2

.Replacement.Text = "\1"

.Replacement.Font.Italic = True

.Execute Replace:=wdReplaceAll

End With

' Convert [pre]section[/pre] to HTML sample

Do

Set rng = ActiveDocument.Range

fileEnd = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = "^p" & pr & "^p"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceOne

End With

codeStart = rng.End

If rng.End <> fileEnd Then

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = "^p" & npr & "^p"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceOne

End With

codeEnd = rng.End

rng.Start = codeStart

rng.Style = "HTML Sample"

End If

Loop Until rng.End = fileEnd

End Sub

**Sub ForumTextFormat()**

**'** Paul Beverley - Version 21.11.21

' Converts text from Word to Forum and vice versa

' Are there any existing format codes?

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

' Anything inside [], starting with /

.Text = "\[/\*\]"

.Replacement.Text = ""

.MatchWildcards = True

.Execute

End With

If rng.Find.Found = True Then

' If so, restore Word formatting and remove them

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[b\](\*)\[/b\]"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

.Replacement.Font.Bold = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[i\](\*)\[/i\]"

.Replacement.Text = "\1"

.Replacement.Font.Italic = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[u\](\*)\[/u\]"

.Replacement.Text = "\1"

.Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[sub\](\*)\[/sub\]"

.Replacement.Text = "\1"

.Replacement.Font.Subscript = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[sup\](\*)\[/sup\]"

.Replacement.Text = "\1"

.Replacement.Font.Superscript = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[s\](\*)\[/s\]"

.Replacement.Font.StrikeThrough = True

.Replacement.Text = "\1"

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[color=#FF0000\](\*)\[/color\]"

.Replacement.Font.Color = wdColorRed

.Replacement.Text = "\1"

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[color=#0000CC\](\*)\[/color\]"

.Replacement.Font.Color = wdColorBlue

.Replacement.Text = "\1"

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[code] [/code]"

.Replacement.Text = "^t"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[code\](\*)\[/code\]"

.Replacement.Style = "HTML Sample"

.Replacement.Text = "\1"

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "&lt;"

.Replacement.Text = "<"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "&gt;"

.Replacement.Text = ">"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Else

' Otherwise ADD the forum commands

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Bold = True

.Replacement.Text = "[b]^&[/b]"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Replacement.Text = "[i]^&[/i]"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Subscript = True

.Replacement.Text = "[sub]^&[/sub]"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Superscript = True

.Replacement.Text = "[sup]^&[/sup]"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = True

.Replacement.Text = "[s]^&[/s]"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Replacement.Text = "[u]^&[/u]"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Style = "HTML Sample"

.Replacement.Text = "[code]^&[/code]"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = wdColorBlue

.Replacement.Text = "[color=#0000CC]^&[/color]"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Color = wdColorRed

.Replacement.Text = "[color=#FF0000]^&[/color]"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

normalFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

.ClearFormatting

.Replacement.ClearFormatting

.Text = "'"

.Font.Name = normalFont

.Replacement.Text = "'"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = """"

.Font.Name = normalFont

.Replacement.Text = """"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = " - "

.Replacement.Text = " ^= "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8216) & "quote" & ChrW(8217)

.Replacement.Text = "^39quote^39"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^t"

.Replacement.Text = "[code] [/code]"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<"

.Replacement.Text = "&lt;"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ">"

.Replacement.Text = "&gt;"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "&lt;div"

.Replacement.Text = "<div"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "&lt;/div&gt;"

.Replacement.Text = "</div>"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

Options.AutoFormatAsYouTypeReplaceQuotes = False

.ClearFormatting

.Replacement.ClearFormatting

.Text = "'&gt;"

.Replacement.Text = "'>"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Options.AutoFormatAsYouTypeReplaceQuotes = True

Set rng = ActiveDocument.Content

rng.Copy

End If

End Sub

**Sub WhatsAppTextFormat()**

' Paul Beverley - Version 11.11.20

' Converts text from Word to WhatsApp formatting

nmlFont = ActiveDocument.Styles(wdStyleNormal).Font.Name

' Are there any existing format codes?

Set rng = ActiveDocument.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[\_\\*`~]"

.Replacement.Text = ""

.MatchWildcards = True

.Execute

End With

If rng.Find.Found = True Then

' If so, remove them

rng.Find.Execute Replace:=wdReplaceAll

Else

' Otherwise ADD the WhatsApp formatting commands

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Format = True

.Font.Bold = True

.Replacement.Text = "\*^&\*"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Font.Italic = True

.Replacement.Text = "\_^&\_"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Font.StrikeThrough = True

.Replacement.Text = "~^&~"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Font.Name = "Courier New"

.Replacement.Text = "```^&```"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Set rng = ActiveDocument.Content

rng.Copy

End If

End Sub

**Sub IndexElide()**

**'** Paul Beverley - Version 23.12.10

' Add elision to an index

For Each para In ActiveDocument.Paragraphs

Set rng = para.Range

thisNum = 0: firstNum = 0: topNum = 0: prevNum = 0

' Read thisNumber

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "[0-9]{1,}"

.Replacement.Text = ""

.Execute

End With

If rng.Find.Found = True Then

thisText = rng

thisNum = Val(thisText)

thisNumStart = rng.Start

thisNumEnd = rng.End

rng.Start = rng.End

End If

If thisNum = 0 Then GoTo nextPara

gotJustOne:

' Got first number in a possible run

firstNum = thisNum

firstNEnd = thisNumEnd

onARun:

' Come here top look for the next number

topNum = thisNum: prevNum = thisNum

topNumStart = thisNumStart

' Read next number

' but first find where the line ends

Set rng2 = para.Range

lineEnd = rng2.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "[0-9]{1,}"

.Replacement.Text = ""

.Execute

End With

' If we've found a number and it's on the current line

If rng.Find.Found = True And rng.Start < lineEnd Then

thisText = rng

thisNum = Val(thisText)

thisNumStart = rng.Start

thisNumEnd = rng.End

rng.Start = rng.End

End If

If thisNum = prevNum Then

' If no more numbers, and this is only

' a single number then go to next para

If topNum = firstNum Then GoTo nextPara

Else

' If the run is continuing

If thisNum = prevNum + 1 Then GoTo onARun

End If

' If we're at the beginning of a new run ...

If firstNum = topNum Then GoTo gotJustOne

' ...type the hyphen in the previous run

rng.Start = firstNEnd

rng.End = topNumStart

chopLength = topNumStart - firstNEnd - 1

rng.Select

Selection.Delete

Selection.TypeText Text:= chr(150)

thisNumStart = thisNumStart - chopLength

thisNumEnd = thisNumEnd - chopLength

rng.Start = thisNumEnd

rng.End = thisNumEnd

prevNum = 0

If topNum = thisNum Then

' If the end of line has been reached

GoTo nextPara

Else

' If not, then we've got a first number

' so go and find a second one.

GoTo gotJustOne

End If

nextPara:

Next para

Selection.HomeKey Unit:=wdStory

End Sub

**Sub BasicIndexer()**

**'** Paul Beverley - Version 27.12.10

' Does basic indexing

textFile = "ReadyForIndexing"

listFile = "Keywords\_Plus"

pageMarker = "Page Proof page "

searchDelimiter = ","

listDelimiter = ", "

repeatNumbers = True

Application.Windows(listFile).Activate

For Each para In ActiveDocument.Paragraphs

Set rng = para.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = True

.Text = "[" & searchDelimiter & "]"

.Replacement.Text = ""

.Execute

End With

rng.End = rng.Start

rng.Start = para.Range.Start

headWord = rng

StatusBar = ">>>>>>>>>>>> " & headWord

' Various dashes and apostrophes to "any character"

headWord = Replace(headWord, "-", "^?")

headWord = Replace(headWord, ChrW(8211), "^?")

headWord = Replace(headWord, ChrW(8212), "^?")

headWord = Replace(headWord, "'", "^?")

headWord = Replace(headWord, ChrW(8217), "^?")

If Len(headWord) > 3 Then

Application.Windows(textFile).Activate

Set rng2 = ActiveDocument.Range

foundPages = ""

Do

previousPage = ""

' Find each occurrence of headword

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.MatchCase = False

.MatchWholeWord = True

.Text = headWord

.Replacement.Highlight = True

.Replacement.Text = ""

.Execute Replace:=wdReplaceOne

End With

comeBackHere = rng2.End

rng2.Start = rng2.End

If rng2.Find.Found Then

' Find current page number

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = pageMarker

.Replacement.Text = ""

.Execute

End With

If rng2.Find.Found = True Then

rng2.Start = rng2.End

rng2.MoveEnd wdCharacter, 6

textBit = rng2

pageNum = Left(textBit, InStr(textBit, " ") - 1)

If repeatNumbers = True Then

foundPages = foundPages & pageNum & listDelimiter

Else

If previousNumber <> pageNum Then

foundPages = foundPages & pageNum & listDelimiter

previousNumber = pageNum

End If

End If

rng2.Start = comeBackHere

rng2.End = comeBackHere

Else

comeBackHere = 0

End If

Else

comeBackHere = 0

End If

Loop Until comeBackHere = 0

Application.Windows(listFile).Activate

rng.Start = para.Range.End - 1

rng.InsertAfter Text:=": " & foundPages

Application.Windows(textFile).Activate

End If

Next para

' Remove trailing commas from list

Application.Windows(listFile).Activate

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = ", ^p"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

End Sub

**Sub BasicIndexer2()**

' Paul Beverley - Version 22.10.12

' Also does basic indexing

repeatNumbers = False

listDelimiter = ", "

addaTab = True

myResponse = MsgBox("Is the cursor on the first line of the word list?", \_

vbQuestion + vbYesNo)

If myResponse = vbNo Then Exit Sub

paraStart = ActiveDocument.Range(0, Selection.Paragraphs(1).Range.End).Paragraphs.Count

Selection.HomeKey Unit:=wdLine

listStart = Selection.Start

For paraNum = paraStart To ActiveDocument.Paragraphs.Count

Set rng = ActiveDocument.Paragraphs(paraNum).Range

rng.Select

Selection.Range.HighlightColorIndex = wdGray25

Selection.MoveEnd , -1

headWord = Selection

' Various dashes and apostrophes to "any character"

headWord = Replace(headWord, "-", "^?")

headWord = Replace(headWord, ChrW(8211), "^?")

headWord = Replace(headWord, ChrW(8212), "^?")

headWord = Replace(headWord, "'", "^?")

headWord = Replace(headWord, ChrW(8217), "^?")

If Len(headWord) > 3 Then

Set rng = ActiveDocument.Content

foundPages = ""

Do

previousPage = ""

' Find each occurrence of headword

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.MatchCase = False

.Highlight = False

.Text = headWord

.Replacement.Highlight = True

.Execute Replace:=wdReplaceOne

End With

rng.Start = rng.End

If rng.Find.Found Then

' Find current page number

Application.Browser.Target = wdBrowsePage

pageNum = Trim(Str(rng.Information(wdActiveEndAdjustedPageNumber)))

If repeatNumbers = True Then

foundPages = foundPages & pageNum & listDelimiter

Else

If previousNumber <> pageNum Then

foundPages = foundPages & pageNum & listDelimiter

previousNumber = pageNum

End If

End If

End If

Loop Until rng.Find.Found = False

If addaTab = True Then foundPages = vbTab & foundPages

Selection.InsertAfter Text:=foundPages

End If

Next paraNum

' Remove trailing commas from list

Set rng = ActiveDocument.Content

rng.Start = listStart

rng.HighlightColorIndex = 0

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.MatchWildcards = False

.Text = ", ^p"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

rng.Select

Selection.Collapse wdCollapseEnd

End Sub

**Sub NumberParasAuto()**

**'** Paul Beverley - Version 21.05.11

' Adds hierarchical section numbering

addChapterNum = False

chapWord = ""

chapNum = 1

' Make sure that page breaks are in Normal style

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^m"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Replacement.Style = ActiveDocument.Styles(wdStyleNormal)

End With

h = "Heading "

levelWas = 1

chapNum = chapNum - 1

myNum = Trim(Str(chapNum))

For Each para In ActiveDocument.Paragraphs

para.Range.Select

head = Selection.Style

If head = "Heading 1" Then

head = ""

If Left(para.Range, Len(chapWord)) = chapWord Then

levelWas = 1

chapNum = chapNum + 1

myNum = Trim(Str(chapNum))

If addChapterNum = True Then para.Range.InsertBefore \_

Text:=myNum & vbTab

End If

End If

If InStr(head, h) > 0 Then

levelNow = Val(Replace(head, h, ""))

If levelNow = levelWas Then

endNum = Val(Right(myNum, 2))

If endNum < 1 Then endNum = Val(Right(myNum, 1))

newEndNum = endNum + 1

myNum = Replace(myNum & "!", Trim(Str(endNum)) & "!", Trim(Str(newEndNum)))

Else

If levelNow > levelWas Then

For i = 1 To levelNow - levelWas

myNum = myNum + ".1"

Next i

Else

For i = 1 To levelWas - levelNow

myNumPlus = myNum & "!"

temp = myNumPlus

temp = Right(temp, Len(temp) - InStr(temp, ".") - 1)

If Asc(temp) <> Asc(".") Then

temp = Right(temp, Len(temp) - 1)

End If

Next i

myNum = Replace(myNumPlus, temp, "")

endNum = Val(Right(myNum, 2))

If endNum < 1 Then endNum = Val(Right(myNum, 1))

newEndNum = endNum + 1

myNum = Replace(myNum & "!", Trim(Str(endNum)) & \_

"!", Trim(Str(newEndNum)))

End If

End If

levelWas = levelNow

para.Range.InsertBefore Text:=myNum & vbTab

End If

Next para

End Sub

**Sub** **NumberParasTagged()**

' Paul Beverley - Version 18.06.20

' Adds numbering to first-level headings tagged with <a>

myTag = "<a>"

titleLine = ActiveDocument.Paragraphs(1)

anglePos = InStr(titleLine, ">")

chapNumText = Mid(titleLine, anglePos + 1)

chapNum = Val(chapNumText)

secNum = 0

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myTag

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.Collapse wdCollapseEnd

rng.Select

secNum = secNum + 1

Selection.TypeText Text:=Trim(Str(chapNum)) & "." & \_

Trim(Str(secNum)) & vbTab

rng.Start = endNow

rng.Find.Execute

Loop

End Sub

**Sub TypeSectionNumber()**

' Paul Beverley - Version 29.03.17

' Adds a section number to the current heading

doSentenceCase = False

doTitleCase = True

chopThese = ".,: "

chapNum = "1"

varExists = False

For Each v In ActiveDocument.Variables

If v.Name = "secNumber" Then varExists = True: Exit For

Next v

If varExists = False Then ActiveDocument.Variables.Add "secNumber", chapNum & ".0"

secNumText = ActiveDocument.Variables("secNumber")

Selection.Expand wdParagraph

tabPos = InStr(Selection, vbTab)

If tabPos > 0 Then

secNumText = Left(Selection, tabPos - 1)

ActiveDocument.Variables("secNumber") = secNumText

Beep

Selection.Collapse wdCollapseEnd

Exit Sub

End If

prevNumText = secNumText

myIndex = Right(secNumText, 2)

myRoot = Left(secNumText, Len(secNumText) - 2)

If Left(myIndex, 1) = " " Then

myIndex = Right(secNumText, 1)

myRoot = Left(secNumText, Len(secNumText) - 1)

End If

If Left(myIndex, 1) = "." Then

myIndex = Right(secNumText, 1)

myRoot = Left(secNumText, Len(secNumText) - 1)

End If

secNumText = myRoot & Trim(Str(Val(myIndex) + 1))

Selection.Expand wdParagraph

Selection.MoveEnd , -1

Selection.Start = Selection.End - 1

If InStr(chopThese, Selection) > 0 Then Selection.Delete

Selection.Expand wdParagraph

Selection.MoveEnd , -1

Selection.Start = Selection.End - 1

If InStr(chopThese, Selection) > 0 Then Selection.Delete

Selection.Expand wdParagraph

Selection.MoveEnd , -1

Selection.Start = Selection.End - 1

If InStr(chopThese, Selection) > 0 Then Selection.Delete

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

myText = InputBox("Number?", "TypeSectionNumber", secNumText)

If myText = "" Then Exit Sub

If myText = "-" Then

myText = prevNumText & ".1"

End If

If myText = "+" Then

n = Split(secNumText, ".")

Debug.Print secNumText

myLevel = UBound(n)

myText = Trim(Str(n(0)))

If myLevel = 1 Then

MsgBox "Use ""-"" to go to a lower level heading."

Exit Sub

End If

For i = 1 To myLevel - 1

If i = myLevel - 1 Then

myText = myText & "." & Trim(Str(n(i) + 1))

Else

myText = myText & "." & Trim(Str(n(i)))

End If

Next i

End If

If myText = "++" Then

n = Split(secNumText, ".")

Debug.Print secNumText

myLevel = UBound(n)

myText = Trim(Str(n(0)))

If myLevel = 1 Then

MsgBox "Use ""-"" to go to a lower level heading."

Exit Sub

End If

For i = 1 To myLevel - 2

If i = myLevel - 2 Then

myText = myText & "." & Trim(Str(n(i) + 1))

Else

myText = myText & "." & Trim(Str(n(i)))

End If

Next i

End If

If myText = "+++" Then

n = Split(secNumText, ".")

Debug.Print secNumText

myLevel = UBound(n)

myText = Trim(Str(n(0)))

If myLevel = 1 Then

MsgBox "Use ""-"" to go to a lower level heading."

Exit Sub

End If

For i = 1 To myLevel - 3

If i = myLevel - 3 Then

myText = myText & "." & Trim(Str(n(i) + 1))

Else

myText = myText & "." & Trim(Str(n(i)))

End If

Next i

End If

Selection.TypeText Text:=myText & vbTab

ActiveDocument.Variables("secNumber") = myText

If doTitleCase = True Then Call TitleHeadingCapper

If doSentenceCase = True Then Call HeadingSentenceCase

End Sub

**Sub FindNextBigText()**

' Paul Beverley - Version 29.03.17

' Searches down for a bigger than Normal font

myBaseStyle = ActiveDocument.Styles(wdStyleNormal)

normalSize = ActiveDocument.Styles(myBaseStyle).Font.Size

myEnd = ActiveDocument.Content.End

Do

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

DoEvents

If Selection.End > myEnd - 2 Or \_

Selection.range.Information(wdInEndnote) Then

Beep

Exit Sub

End If

If Selection.range.Information(wdWithInTable) Then

Do

Selection.MoveDown , 1

DoEvents

Loop Until Not (Selection.range.Information(wdWithInTable))

End If

If Selection.range.Information(wdInFootnote) Then

Do

Selection.MoveDown , 1

DoEvents

Loop Until Not (Selection.range.Information(wdInFootnote))

End If

Selection.MoveEnd , 1

Loop Until Selection.range.Font.Size > normalSize

Selection.MoveUp , 1

Selection.MoveDown , 1

End Sub

**Sub SuitToText()**

**'** Paul Beverley - Version 21.07.11

' Convert playing card suit symbolks to text

For Each fld In ActiveDocument.Fields

myText = fld.Code.Text

codePos = InStr(myText, "SYMBOL ") + 7

myCode = Mid(myText, codePos, 3)

Select Case Val(myCode)

Case 167: mySuit = "cx"

Case 168: mySuit = "dx"

Case 169: mySuit = "hx"

Case 170: mySuit = "sx"

Case Else: mySuit = "??????"

End Select

fld.Select

Selection.TypeText Text:=mySuit

Next

End Sub

**Sub MarkFinalOnOff()**

**'** Paul Beverley - Version 14.05.11

' Switch "Mark as final" facility on/off

ActiveDocument.Final = Not (ActiveDocument.Final)

End Sub

**Sub URLlinker()**

**'** Paul Beverley - Version 30.07.24

' Finds all URLs and emails in the text and links them

Set rng = Selection.Range.Duplicate

With rng.Find

.Text = "^$.^$"

.Font.Underline = False

.Replacement.Text = ""

.Wrap = wdFindStop

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

Do While InStr(vbCr & " ", Right(rng.Text, 1)) = 0

rng.MoveEnd , 1

DoEvents

Loop

rng.MoveEnd , -1

If rng.Characters.Last = "." Then rng.MoveEnd , -1

If rng.Characters.Last = "," Then rng.MoveEnd , -1

Do While InStr(vbCr & " ", Left(rng.Text, 1)) = 0

rng.MoveStart , -1

If rng = rngWas Then Exit Do

rngWas = rng.Text

DoEvents

Loop

Debug.Print rng.start

rng.MoveStart , 1

myAddress = rng.Text

Debug.Print myAddress

If InStr(rng, "@") Or InStr(rng, "/") Or InStr(rng, "www") \_

Or InStr(rng, "http") Then

If InStr(rng, "http") > 0 Then

rng.Text = Replace(rng.Text, "https://", "")

rng.Text = Replace(rng.Text, "http://", "")

End If

If InStr(rng, "@") > 0 Then

emailText = rng.Text

Set myLink = ActiveDocument.Hyperlinks.Add(Anchor:=rng, \_

Address:="mailto:" & myAddress, TextToDisplay:=rng.Text)

Else

Set myLink = ActiveDocument.Hyperlinks.Add(Anchor:=rng, \_

Address:=myAddress, TextToDisplay:=rng.Text)

End If

rng.start = myLink.Range.End

rng.Select

rng.Collapse wdCollapseStart

End If

rng.MoveStart , 10

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Beep

End Sub

**Sub URLshrinker()**

**'** Paul Beverley - Version 15.07.23

' Reduces the extent of a URL link to just the selected text

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

' Find the beginning and end of the existing link

Do

rng.MoveStart , -1

DoEvents

Debug.Print rng.Font.Underline

Loop Until rng.Font.Underline > 1

Do

rng.MoveEnd , 1

DoEvents

Debug.Print rng.Font.Underline

Loop Until rng.Font.Underline > 1

myCode = rng.Fields(1).Code

myCode = Replace(myCode, "HYPERLINK", "")

myCode = Replace(myCode, """", "")

rng.Fields(1).Unlink

Set newLink = ActiveDocument.Hyperlinks.Add(Anchor:=Selection.Range, \_

Address:=myCode)

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub URLunlinker()**

' Paul Beverley - Version 28.02.20

' Unlinks all the URLs in the selection or the whole file

' foreText = " ("

' afterText = ") "

foreText = " "

afterText = " "

' remove1 = "http://"

' remove2 = "**[Error! Hyperlink reference not valid.](https://\")**

remove1 = ""

remove2 = ""

If Selection.Start <> Selection.End Then

Set rng = Selection.range.Duplicate

Else

Set rng = ActiveDocument.Content

End If

For i = rng.Fields.Count To 1 Step -1

Set fld = rng.Fields(i)

myType = fld.Type

If myType = 88 Then

fld.Select

myURLtext = fld.Code

myDisplayText = fld.Result

myURLtext = Replace(myURLtext, "HYPERLINK """, "")

qtPos = InStr(myURLtext, """")

myURLtext = Trim(Left(myURLtext, qtPos - 1))

myURLtext = Replace(myURLtext, remove1, "")

myURLtext = Replace(myURLtext, remove2, "")

myDisplayText = Replace(myDisplayText, remove1, "")

myDisplayText = Replace(myDisplayText, remove2, "")

Debug.Print myDisplayText

Debug.Print myURLtext & vbCr

myStart = Selection.Start

myEnd = myStart + Len(myDisplayText)

fld.Delete

If myURLtext <> myDisplayText Then

Selection.TypeText Text:=myDisplayText & foreText \_

& myURLtext & afterText

Else

Selection.TypeText Text:=myDisplayText

End If

Selection.Start = myStart

Selection.End = myEnd

If makeTextBold = True Then Selection.range.Bold = True

Selection.range.Font.ColorIndex = wdAuto

Selection.range.Underline = False

End If

Next i

End Sub

**Sub EmailLinker()**

' Paul Beverley - Version 18.10.19

' Finds all email addresses in the text and links them

charsInEmails = "[%./:a-zA-Z0-9\_\-]"

myFind = "<" & charsInEmails & "{3,}\@" & charsInEmails & "{3,}"

Set rng = ActiveDocument.Content

With rng.Find

.Text = myFind

.Replacement.Text = ""

.Wrap = wdFindStop

.Forward = True

.MatchWildcards = True

.Execute

End With

rng.Select

Do While rng.Find.Found = True

rng.Select

Selection.Collapse wdCollapseStart

stNow = Selection.Start

Selection.MoveLeft , 1

stBefore = Selection.Start

If stNow = stBefore + 1 Then

myAddress = rng.Text

rng.Select

Set myLink = ActiveDocument.Hyperlinks.Add(Anchor:=rng, \_

Address:=myAddress, TextToDisplay:=rng.Text)

rng.Start = myLink.range.End

rng.End = ActiveDocument.Content.End

Else

rng.Start = rng.End

End If

rng.Find.Execute

DoEvents

Loop

Beep

End Sub

**Sub ShowFormatting()**

' Paul Beverley - Version 10.05.17

' Display all formatting or just paragraph marks

tabsToo = True

sa = ActiveWindow.View.ShowAll

sp = ActiveWindow.View.ShowParagraphs

If Not (sa) And Not (sp) Then ActiveWindow.View.ShowAll = True

If sa And Not (sp) Then

ActiveWindow.View.ShowAll = False

ActiveWindow.View.ShowParagraphs = True

If tabsToo = True Then ActiveWindow.View.ShowTabs = True

End If

If sp Then

ActiveWindow.View.ShowParagraphs = False

ActiveWindow.View.ShowTabs = False

End If

End Sub

**Sub ShowFormattingMenu()**

' Paul Beverley - Version 01.02.18

' Displays (or not) various formatting markers, and the highlighting

myFavouriteOption\_1 = 7

myFavouriteKey\_1 = "/"

myFavouriteOption\_2 = 1

myFavouriteKey\_2 = "#"

Do

menu = menu & "1 = show paragraphs|"

menu = menu & "2 = show spaces|"

menu = menu & "4 = show tabs|"

menu = menu & "8 = hide the highlighting|"

menu = menu & "16 = show hyphens|"

menu = menu & "32 = show bookmarks|"

menu = Replace(menu, "|", vbCr)

myResponse = InputBox(menu, "ShowFormattingMenu")

Select Case myResponse

Case myFavouriteKey\_1:

myCode = myFavouriteOption\_1

gotValid = True

Case myFavouriteKey\_2:

myCode = myFavouriteOption\_2

gotValid = True

Case Else:

myCode = Val(myResponse)

gotValid = (myCode > -1 And myCode < 64)

If Not gotValid Then

menu = "Type a number between 0 and 63:||"

End If

End Select

Loop Until gotValid

i = myCode

If i > 63 Then i = 63

sp = i Mod 2

i = Int(i / 2)

ss = i Mod 2

i = Int(i / 2)

st = i Mod 2

i = Int(i / 2)

noh = i Mod 2

i = Int(i / 2)

shy = i Mod 2

sb = Int(i / 2)

ActiveWindow.View.ShowSpaces = (ss = 1)

ActiveWindow.View.ShowHighlight = (noh = 0)

ActiveWindow.View.ShowHyphens = (shy = 1)

ActiveWindow.View.ShowTabs = (st = 1)

ActiveWindow.View.ShowParagraphs = (sp = 1)

ActiveWindow.View.ShowBookmarks = (sb = 1)

End Sub

**Sub HighlightNotThisSize()**

' Paul Beverley - Version 03.10.18

' Highlights all text NOT the same size as the current text

myColour = wdBrightGreen

If Selection.Start = Selection.End Then

myResponse = MsgBox("Highlight the whole file?!", vbQuestion \_

+ vbYesNoCancel, "HighlightNotThisSize")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

Else

Set rng = Selection.range.Duplicate

End If

rng.Font.DoubleStrikeThrough = True

Selection.Collapse wdCollapseStart

mySize = Selection.range.Font.Size

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Size = mySize

.Replacement.Text = ""

.Replacement.Font.DoubleStrikeThrough = False

.Wrap = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.DoubleStrikeThrough = True

.Wrap = False

.Replacement.Text = ""

.Replacement.Font.DoubleStrikeThrough = False

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub RibbonMinMax()**

' Paul Beverley - Version 18.12.14

' Toggle between maximize and minimize of the ribbon

ActiveWindow.ToggleRibbon

End Sub

**Sub MacroSelect()**

' Paul Beverley - Version 12.01.23

' Selects the current macro text

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

Selection.Start = Selection.End

Selection.MoveUp , 3

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^pEnd S" & "ub"

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

endMacro = Selection.End + 2

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^pSub "

.Forward = False

.Replacement.Text = ""

.Execute

End With

Selection.MoveStart , 1

Selection.End = endMacro

With Selection.Find

.Text = oldFind

.Forward = True

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

End Sub

**Sub Enigmatizer()**

' Paul Beverley - Version 25.10.19

' Obfuscates/anonymises the current file

shuffleNumbers = True

notTheseStyles = "Heading 1,Heading 2"

notTheseStyles = ""

vowels = "aeiou"

cons = "bcdfghjklmnpqrstvwstn"

numCons = Len(cons)

nums = "0123456789"

ActiveDocument.TrackRevisions = False

tot = ActiveDocument.Paragraphs.Count

For Each myPara In ActiveDocument.Paragraphs

parStyle = myPara.range.Style

If InStr(notTheseStyles, parStyle) = 0 And Len(myPara.range.Text) > 3 Then

numChars = myPara.range.Characters.Count

For i = 2 To numChars - 1

cho = myPara.range.Characters(i)

ch = LCase(cho)

If UCase(cho) <> LCase(cho) Then

If InStr(vowels, LCase(ch)) > 0 Then

ch = cho

Else

ch = Mid(cons, Int(numCons \* Rnd()) + 1, 1)

End If

isUC = (UCase(cho) = cho)

If isUC Then ch = UCase(ch)

myPara.range.Characters(i).Text = ch

Else

If shuffleNumbers = True And InStr(nums, ch) > 0 Then

ch = Mid(nums, Int(10 \* Rnd()) + 1, 1)

myPara.range.Characters(i).Text = ch

End If

End If

DoEvents

Next i

End If

p = p + 1

l = tot - p

Next myPara

Beep

End Sub

**Sub TweetCheck()**

' Paul Beverley - Version 10.05.18

' Highlights paragraphs with too many characters

maxChars = 280

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If Selection.End > Selection.Start Then

Set rng = ActiveDocument.Content

gotOne = False

For Each myPara In ActiveDocument.Paragraphs

If myPara.range.End > Selection.End Then

' Allowing one character for the newline...

If myPara.range.Characters.Count > maxChars + 1 Then

If gotOne = False Then

myPara.range.Select

Beep

Selection.Collapse wdCollapseStart

End If

rng.Start = myPara.range.Start

rng.End = rng.Start + maxChars

rng.HighlightColorIndex = wdGray25

gotOne = True

End If

End If

Next myPara

If gotOne = False Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Selection.Collapse wdCollapseStart

End If

Else

Selection.Expand wdParagraph

If Selection.range.Characters.Count > maxChars + 1 Then

Selection.End = Selection.Start + maxChars

Selection.range.HighlightColorIndex = wdGray25

Else

Beep

End If

Selection.Collapse wdCollapseEnd

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub AutoCorrectItemsList()**

' Paul Beverley - Version 03.01.17

' Lists all current autocorrect items

Documents.Add

For i = 1 To Application.AutoCorrect.Entries.Count

myReplaceThis = Application.AutoCorrect.Entries(i).Name

myWithThis = Application.AutoCorrect.Entries(i).Value

Selection.TypeText Text:=myReplaceThis & vbTab & myWithThis & vbCr

Next i

End Sub

**Sub AutoCorrectItemsDeleteAdd()**

' Paul Beverley - Version 03.01.17

' Optionally deletes all items, then adds new items

myResponse = MsgBox("Delete all existing autocorrect list?", vbQuestion \_

+ vbYesNoCancel, "Autocorrect Item Replacer")

If myResponse = vbYes Then

totItems = Application.AutoCorrect.Entries.Count

For i = totItems To 1 Step -1

Application.AutoCorrect.Entries(i).Delete

Next i

End If

myResponse = MsgBox("Add in these autocorrect items?", vbQuestion \_

+ vbYesNoCancel, "Autocorrect Item Replacer")

If myResponse = vbCancel Or myResponse = vbNo Then Exit Sub

myItemCount = Application.AutoCorrect.Entries.Count

myCount = 0

For Each myPara In ActiveDocument.Paragraphs

myText = Replace(myPara.range.Text, vbCr, "")

tabPos = InStr(myText, vbTab)

If tabPos > 0 Then

myReplaceThis = Left(myText, tabPos - 1)

myWithThis = Mid(myText, tabPos + 1, (Len(myText) - 1))

If Len(myReplaceThis) > 0 And Len(myWithThis) > 0 Then \_

Application.AutoCorrect.Entries.Add myReplaceThis, myWithThis

myCount = myCount + 1

End If

Next

MsgBox ("A total of " & myCount & " entries have been added," & vbCr \_

& "making a total of " & Application.AutoCorrect.Entries.Count)

End Sub

**Sub BoldFirstOccurrence()**

' Paul Beverley - Version 13.09.19

' Emboldens the first occurrence of words in a list

' myColour = wdBlue

myColour = 0

myHighlight = wdYellow

' myHighlight = 0

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Set rng = ActiveDocument.range(Selection.Start, ActiveDocument.Content.End)

myFirst = ""

Dim myText As String

For Each para In rng.Paragraphs

myText = Replace(para.range.Text, vbCr, "")

If myFirst = "" Then myFirst = myText

Set rng2 = ActiveDocument.Content

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText

.Forward = True

.MatchCase = False

.Replacement.Text = ""

.Replacement.Font.Bold = True

.Execute Replace:=wdReplaceOne

End With

If myColour > 0 Then rng2.Font.ColorIndex = myColour

If myHighlight > 0 Then rng2.HighlightColorIndex = myHighlight

Next para

Beep

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFirst

.Forward = True

.MatchCase = False

.Execute

End With

rng.Select

Selection.Collapse wdCollapseStart

MsgBox "Finished"

End Sub

**Sub MacroLauncher()**

' Paul Beverley - Version 12.05.22

' Offers a list of macros to launch (was called "MacroMenu")

myList1 = "a=AccentAlyse, d=DocAlyse, h=HyphenAlyse"

myList2 = "w=WordPairAlyse, p=ProperNounAlyse, i=IZISCount"

myList3 = "z=IStoIZ, s=IZtoIS, l=SpellingErrorLister"

myList4 = "c=CopyTextSimple, m=MultiFileText, f=FRedit"

myList5 = "e=SpellingErrorHighlighter, u=UKUSCount"

' myList5 = "e=SpellingErrorHighlighter, u=UKUSCount, g=MegAlyse"

myList = "," & myList1 & "," & myList2 & "," & myList3 & "," \_

& myList4 & "," & myList5

myList = Replace(myList, ",,", ",")

mcr = Split(myList, ",")

myPrompt = ""

myCodes = ""

numItems = UBound(mcr)

For i = 1 To numItems

mcr(i) = Trim(mcr(i))

myCode = Left(mcr(i), 1)

myCodes = myCodes & myCode

mName = Mid(mcr(i), 3)

myPrompt = myPrompt & Trim(Str(i)) & " " & ChrW(8211) & " "

myPrompt = myPrompt & mName & " (" & myCode & ")" & vbCr

Next i

Do

myResponse = UCase(InputBox(myPrompt, "MacroLauncher"))

myNum = Val(myResponse)

If myResponse = "" Then Exit Sub

If myNum = 0 Then

myNum = InStr(UCase(myCodes), Left(myResponse, 1))

End If

Loop Until myNum > 0 And myNum < numItems + 1

mName = Mid(mcr(myNum), 3)

Application.Run MacroName:=mName

End Sub

**Sub MacroMenuForMac()**

' Paul Beverley - Version 04.07.20

' Offers a list of macros to launch

myList1 = "a=AccentAlyse, d=DocAlyse, h=HyphenAlyse"

myList2 = "w=WordPairAlyse, p=ProperNounAlyse, i=IZIScount"

myList3 = "z=IStoIZ, s=IZtoIS, l=SpellingErrorLister"

myList4 = "c=CopyTextSimple, m=MultiFileText, f=FRedit"

myList5 = "e=SpellingErrorHighlighter, u=UKUScount"

' myList5 = "e=SpellingErrorHighlighter, u=UKUScount, g=MegAlyse"

myList = "," & myList1 & "," & myList2 & "," & myList3 & "," \_

& myList4 & "," & myList5

myList = Replace(myList, ",,", ",")

mcr = Split(myList, ",")

myPrompt = ""

myCodes = ""

numItems = UBound(mcr)

For i = 1 To numItems

mcr(i) = Trim(mcr(i))

myCode = Left(mcr(i), 1)

myCodes = myCodes & myCode

mName = Mid(mcr(i), 3)

myPrompt = myPrompt & Trim(Str(i)) & " " & ChrW(8211) & " "

myPrompt = myPrompt & mName & " (" & myCode & ")" & vbCr

Next i

Do

myResponse = UCase(InputBox(myPrompt, "MacroMenu"))

myNum = Val(myResponse)

If myResponse = "" Then Exit Sub

If myNum = 0 Then

myNum = InStr(UCase(myCodes), Left(myResponse, 1))

End If

Loop Until myNum > 0 And myNum < numItems + 1

mName = Mid(mcr(myNum), 3)

Select Case nName

Case "AccentAlyse"

Call AccentAlyse

Case "CopyTextSimple"

Call CopyTextSimple

Case "DocAlyse"

Call DocAlyse

Case "FRedit"

Call FRedit

Case "HyphenAlyse"

Call HyphenAlyse

Case "IStoIZ"

Call IStoIZ

Case "IZISCount"

Call IZIScount

Case "IZtoIS"

Call IZtoIS

Case "MegAlyse"

Call MegAlyse

Case "MultiFileText"

Call MultiFileText

Case "ProperNounAlyse"

Call ProperNounAlyse

Case "SpellingErrorHighlighter"

Call SpellingErrorHighlighter

Case "SpellingErrorLister"

Call SpellingErrorLister

Case "UKUScount"

Call UKUScount

Case "WordPairAlyse"

Call WordPairAlyse

Case Else

Beep

MsgBox "Not on list"

End Select

End Sub

**Sub TitlesShowHide()**

**'** Paul Beverley - Version 25.09.24

' Toggles between light grey font colour text and full colour

' PB blue

myColorBlue = &HFF0000

' JY blue

myColorBlue = &HF0B000

preBlue = &HF6F6F6

preRed = &HF4F4F4

preBlack = &HF5F5F5

Set pbTitlesFile = ActiveDocument

For i = 0 To ActiveDocument.Shapes.Count

doIt = False

If i = 0 Then

doIt = True

Set rng = ActiveDocument.Content

Else

' Trying to catch type 1 or 9 - hence Mod 8 = 1

If ActiveDocument.Shapes(i).Type Mod 8 = 1 Then

If ActiveDocument.Shapes(i).TextFrame.HasText Then

doIt = True

Set rng = ActiveDocument.Shapes(i).TextFrame.TextRange

End If

End If

End If

If doIt = True Then

If pbShowHide = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = ""

.Font.Color = preBlue

.MatchWildcards = False

.Replacement.Text = ""

.Replacement.Font.Color = myColorBlue

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Font.Color = preRed

.Replacement.Font.Color = wdColorRed

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Wrap = wdFindContinue

.Replacement.ClearFormatting

.Font.Color = preBlack

.Replacement.Font.Color = wdColorBlack

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

End With

Else

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = ""

.Font.Color = myColorBlue

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Color = preBlue

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Font.Color = wdColorRed

.Replacement.Text = ""

.Replacement.Font.Color = preRed

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Font.Color = wdColorBlack

.Replacement.Text = ""

.Replacement.Font.Color = preBlack

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Font.Color = wdColorBlack

.Replacement.Text = ""

.Replacement.Font.Color = preBlack

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(160)

.Replacement.Text = ""

.Replacement.Font.Underline = False

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Color = wdColorBlack

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

End With

End If

End If

Next i

pbShowHide = Not (pbShowHide)

End Sub

**Sub TitlesReveal()**

**'** Paul Beverley - Version 08.08.24

' Switches the next light grey font colour to full colour

myColorBlue = &HF0B000

myColorBlue = &HFF0000

preBlue = &HF6F6F6

preRed = &HF4F4F4

preBlack = &HF5F5F5

Set docNow = ActiveDocument

pbTitlesFile.Activate

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

stopNow = False

doneSome = False

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Collapse wdCollapseStart

Do While stopNow = False And doneSome = False

Do

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

ln = Len(rng)

DoEvents

If rng.start = ActiveDocument.Content.End Then

Beep

Exit Sub

End If

DoEvents

Loop Until ln > 1 And Asc(rng) <> 12

gotBox = False

rng.Select

Selection.Collapse wdCollapseStart

For Each wd In rng.Words

newColor = 99

myColour = wd.Font.Color

If myColour = 9999999 Then

For Each ch In wd.Characters

myColour = ch.Font.Color

Select Case myColour

Case preBlue: newColor = myColorBlue

Case preRed: newColor = wdColorRed

Case preBlack: newColor = wdColorBlack

End Select

If newColor <> 99 Then

ch.Font.Color = newColor

doneSome = True

End If

Next ch

Else

Select Case myColour

Case preBlue: newColor = myColorBlue

Case preRed: newColor = wdColorRed

Case preBlack: newColor = wdColorBlack

End Select

If newColor <> 99 Then

wd.Font.Color = newColor

doneSome = True

End If

End If

stopMiddle = (AscW(Right(wd, 1)) = 160 Or Asc(wd) = 11)

If doneSome = True And (stopMiddle Or wd = vbCr) Then

stopNow = True

Exit For

End If

Next wd

DoEvents

Loop

If stopMiddle Then

wd.Select

Else

rng.Select

End If

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

docNow.Activate

End Sub

**Sub FixedFontRed()**

' Paul Beverley - Version 13.05.20

' Sets font to an 'almost' red font colour

Selection.Font.Color = &H1111FF

End Sub

**Sub FixedFontBlue()**

' Paul Beverley - Version 13.05.20

' Sets font to an 'almost' blue font colour

Selection.Font.Color = &HFF1111

End Sub

**Sub FixedFontBlack()**

' Paul Beverley - Version 13.05.20

' Sets font to an 'almost' black font colour

Selection.Font.Color = &H111111

End Sub

**Sub FixedFontSwitch()**

' Paul Beverley - Version 25.09.24

' Switches font colour red -> blue -> black ->

' JY blue

myBlue = &HF0B011

' PB blue

myBlue = &HF0B011

Set rng = Selection.Range.Duplicate

rng.End = rng.start + 1

Select Case rng.Font.Color

Case &H111111: Selection.Font.Color = &H1111FF

Case &H1111FF: Selection.Font.Color = myBlue

Case myBlue: Selection.Font.Color = &H111111

Case Else: Selection.Font.Color = myBlue

End Select

End Sub

**Sub FileOpener()**

' Paul Beverley - Version 08.08.20

' Menu system to select and load a file(s)

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff"

myPrompt = "L = FRedit library, A = Appendices, V = Video list,"

myPrompt = myPrompt & "M = MTR, M2 = MTR 2, MM = MTR\_Mac, B = Book,"

myPrompt = myPrompt & "BM = Book + Macros,"

findThis = ""

gotoBM = False

myPrompt = Replace(myPrompt, ", ", vbCr)

myPrompt = Replace(myPrompt, ",", vbCr)

myFile = UCase(InputBox(myPrompt, "FileOpener"))

Select Case myFile

Case "L"

Documents.Open FileName:=myFolder & "aFRedit\5\_Library.docx"

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Resize Width:=1000, Height:=500

Application.ActiveWindow.View.Zoom.Percentage = 160

Case "A"

Documents.Open FileName:=myFolder & "ComputerTools4Eds\_Appendices.docx"

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Resize Width:=1100, Height:=420

Application.ActiveWindow.View.Zoom.Percentage = 160

gotoBM = True

Case "V"

Documents.Open FileName:=myFolder & "VideoList.docx"

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Resize Width:=500, Height:=700

Application.ActiveWindow.View.Zoom.Percentage = 160

findThis = "=="

Case "M"

Documents.Open FileName:=myFolder & "Macros\_by\_the\_tourist\_route.docx"

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Resize Width:=800, Height:=900

Application.ActiveWindow.View.Zoom.Percentage = 160

Case "M2"

Documents.Open FileName:=myFolder & "Macros\_by\_the\_tourist\_route\_2.docx"

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Resize Width:=800, Height:=700

Application.ActiveWindow.View.Zoom.Percentage = 160

Case "MM"

Documents.Open FileName:=myFolder & "Macros\_by\_the\_tourist\_route\_Mac.docx"

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Resize Width:=600, Height:=900

Application.ActiveWindow.View.Zoom.Percentage = 160

Case "B"

Documents.Open FileName:=myFolder & "ComputerTools4Eds.docx"

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Resize Width:=1200, Height:=600

Application.ActiveWindow.View.Zoom.Percentage = 160

gotoBM = True

Case "BM"

Documents.Open FileName:=myFolder & "ComputerTools4Eds.docx"

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Move Left:=20, Top:=0

Application.Resize Width:=1200, Height:=350

Application.ActiveWindow.View.Zoom.Percentage = 160

findThis = "1. Bookmarks"

Documents.Open FileName:=myFolder & "TheMacros.docx"

ActiveDocument.ActiveWindow.WindowState = wdWindowStateNormal

Application.Move Left:=20, Top:=350

Application.Resize Width:=1300, Height:=350

Application.ActiveWindow.View.Zoom.Percentage = 160

Case Else: Beep

End Select

If findThis > "" Then

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = findThis

.Replacement.Text = ""

.MatchCase = False

.MatchWildcards = False

.Execute

End With

End If

If gotoBM = True Then

If ActiveDocument.Bookmarks.Exists("myTempMark") Then

ActiveDocument.Bookmarks("myTempMark").Select

Else

Beep

End If

End If

End Sub

**Sub DIYFormatHeadwords()**

' Paul Beverley - Version 24.11.20

' Adds a character style to the first word of every para in a given style

Set rng = ActiveDocument.Content

' Set rng = Selection.Range.Duplicate

' For each of the paragraphs in the range

For Each myPara In rng.Paragraphs

If Len(myPara.Range.Text) > 1 Then

If myPara.Range.Style = ActiveDocument.Styles("Normal") Then

myPara.Range.Words(1).Style = ActiveDocument.Styles("Emphasis")

End If

' myPara.Range.Select

End If

DoEvents

Next myPara

Beep

End Sub

**Sub DIYColourLongSentences()**

' Paul Beverley - Version 24.11.20

' Adds a highlight to sentences longer than 40 words

Set rng = ActiveDocument.Content

For Each snt In rng.Sentences

If snt.Words.Count > 60 Then

snt.Select

snt.Font.ColorIndex = wdBlue

End If

DoEvents

Next snt

Beep

End Sub

**Sub DIYColourLongWords()**

' Paul Beverley - Version 24.11.20

' Adds a highlight to sentences longer than 40 words

Set rng = ActiveDocument.Content

For Each wd In rng.Words

If wd.Characters.Count > 12 Then

wd.Font.ColorIndex = wdBlue

wd.Font.Bold = True

End If

wd.Select

DoEvents

Next wd

Beep

End Sub

**Sub TheFindAshortMacro()**

' Paul Beverley - Version 24.11.20

' Finds a macro shorter than a certain length

myLen = 20

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13Sub\*\(\)"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rng.MoveEnd Unit:=wdParagraph, Count:=myLen

If InStr(rng.Text, "End" & " Sub") Then

rng.Select

Selection.Collapse wdCollapseStart

Selection.MoveDown Unit:=wdLine, Count:=myLen

Selection.MoveUp Unit:=wdLine, Count:=myLen

myResponse = MsgBox("Fancy this one?", vbQuestion + vbYesNo)

If myResponse = vbYes Then

Selection.Collapse wdCollapseStart

Exit Sub

End If

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

End Sub

Sub CountTwoPhrases()

' Paul Beverley - Version 28.09.20

' Counts this and that word or phrase

' Ben Dare, based on PB's CountPhrase

''original CountPhrase

doFormatCount = True

doCountWhole = True

limitTimeSpent = True

maxTime = 10

old2Find = Selection.Find.Text

old2Replace = Selection.Find.Replacement.Text

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If limitTimeSpent = True Then

myTime = Timer

Else

myTime = Timer + 999

End If

' If nothing is selected, select the current word

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Selection.MoveEndWhile cset:=ChrW(8217) & "' ", Count:=wdBackward

End If

oldStart = Selection.Start

oldEnd = Selection.End

thisBit = Trim(Selection)

thisBit = Replace(thisBit, "^", "^^")

thisBit = Replace(thisBit, Chr(13), "^p")

If Right(thisBit, 1) = ChrW(8217) Then thisBit \_

= Left(thisBit, Len(thisBit) - 1)

CR = vbCr: CR2 = CR & CR

' Find whether we're in a footnote

InANote = Selection.Information(wdInFootnote)

If InANote = True Then

lineJump = 0

Do

Selection.MoveUp Unit:=wdLine, Count:=1

lineJump = lineJump + 1

Loop Until Selection.Information(wdInFootnote) = False

oldStart = Selection.Start

oldEnd = Selection.Start

End If

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

myTot = ActiveDocument.Range.End

' Count all occurences

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = thisBit

.MatchCase = False

.Replacement.Text = "^& "

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

allCount = ActiveDocument.Range.End - myTot

If allCount > 0 Then WordBasic.EditUndo

nts = ""

' Are there any footnotes?

If ActiveDocument.Footnotes.Count > 0 Then

nts = ActiveDocument.StoryRanges(wdFootnotesStory)

End If

If ActiveDocument.Endnotes.Count > 0 Then

nts = nts & ActiveDocument.StoryRanges(wdEndnotesStory)

nts = Replace(nts, Chr(2), "")

End If

If Len(nts) > 0 Then

lcNts = LCase(nts)

lcBit = LCase(thisBit)

lnWas = Len(nts)

lnNow = Len(Replace(lcNts, lcBit, lcBit & "!"))

allCount = allCount + lnNow - lnWas

End If

myText = "All:" & Str(allCount) & CR

If Timer > myTime + maxTime Then

myText = myText & CR & "Time out!"

GoTo printResult

End If

' Count case sensitively

With rng.Find

.MatchCase = True

.Execute Replace:=wdReplaceAll

End With

caseCount = ActiveDocument.Range.End - myTot

If caseCount > 0 Then WordBasic.EditUndo

' Now notes, if any

If Len(nts) > 0 Then

lnNow = Len(Replace(nts, thisBit, thisBit & "!"))

caseCount = caseCount + lnNow - lnWas

End If

myText = myText & "Case sensitive:" & Str(caseCount) & CR

If Timer > myTime + maxTime Then

myText = myText & CR & "Time out!"

GoTo printResult

End If

If doFormatCount = True Then

' Count bold italic

With rng.Find

.ClearFormatting

.MatchCase = False

.Font.Italic = True

.Font.Bold = True

.Execute Replace:=wdReplaceAll

End With

biCount = ActiveDocument.Range.End - myTot

If biCount > 0 Then

WordBasic.EditUndo

myText = myText & "Bold italic (main text only):" & Str(biCount) & CR

End If

' Count italic

With rng.Find

.ClearFormatting

.MatchCase = False

.Font.Italic = True

.Execute Replace:=wdReplaceAll

End With

iCount = ActiveDocument.Range.End - myTot

If iCount > 0 Then

WordBasic.EditUndo

myText = myText & "Italic (main text only):" & Str(iCount - biCount) & CR

End If

' Count bold

With rng.Find

.ClearFormatting

.Font.Bold = True

.Execute Replace:=wdReplaceAll

End With

bCount = ActiveDocument.Range.End - myTot

If bCount > 0 Then

WordBasic.EditUndo

myText = myText & "Bold (main text only):" & Str(bCount - biCount) & CR

End If

End If

If Timer > myTime + maxTime Then

myText = myText & CR & "Time out!"

GoTo printResult

End If

If doCountWhole = True Then

' Count as whole words (case sensitive)

thisBit = Replace(thisBit, "(", "\(")

thisBit = Replace(thisBit, ")", "\)")

thisBit = Replace(thisBit, "^p", "^13")

' Add text of the notes at the end of main text

If Len(nts) > 0 Then

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=""

End If

Set rng = ActiveDocument.Content

myTot = ActiveDocument.Range.End

With rng.Find

.ClearFormatting

.Text = "[!a-zA-Z]" & thisBit & "[!a-zA-Z]"

.Replacement.Text = "^& "

.MatchWholeWord = False

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

wholeWdCaseCount = ActiveDocument.Range.End - myTot

If wholeWdCaseCount > 0 Then

WordBasic.EditUndo

myText = myText & "Whole words (case sensitive):" & \_

Str(wholeWdCaseCount) & CR

End If

With rng.Find

.ClearFormatting

.Text = ""

.Replacement.Text = ""

.MatchWholeWord = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

' Start of second phrase

thatBit = InputBox("Second word/phrase?", selText)

thatBit = Replace(thatBit, "^", "^^")

thatBit = Replace(thatBit, Chr(13), "^p")

If Right(thatBit, 1) = ChrW(8217) Then thatBit \_

= Left(thatBit, Len(thatBit) - 1)

CR = vbCr: CR2 = CR & CR

' Find whether we're in a footnote

InANote = Selection.Information(wdInFootnote)

If InANote = True Then

lineJump = 0

Do

Selection.MoveUp Unit:=wdLine, Count:=1

lineJump = lineJump + 1

Loop Until Selection.Information(wdInFootnote) = False

oldStart = Selection.Start

oldEnd = Selection.Start

End If

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

myTot = ActiveDocument.Range.End

' Count all occurences

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = thatBit

.MatchCase = False

.Replacement.Text = "^& "

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

allCount = ActiveDocument.Range.End - myTot

If allCount > 0 Then WordBasic.EditUndo

nts = ""

' Are there any footnotes?

If ActiveDocument.Footnotes.Count > 0 Then

nts = ActiveDocument.StoryRanges(wdFootnotesStory)

End If

If ActiveDocument.Endnotes.Count > 0 Then

nts = nts & ActiveDocument.StoryRanges(wdEndnotesStory)

nts = Replace(nts, Chr(2), "")

End If

If Len(nts) > 0 Then

lcNts = LCase(nts)

lcBit = LCase(thatBit)

lnWas = Len(nts)

lnNow = Len(Replace(lcNts, lcBit, lcBit & "!"))

allCount = allCount + lnNow - lnWas

End If

my2Text = "All:" & Str(allCount) & CR

If Timer > myTime + maxTime Then

my2Text = my2Text & CR & "Time out!"

GoTo printResult

End If

' Count case sensitively

With rng.Find

.MatchCase = True

.Execute Replace:=wdReplaceAll

End With

caseCount = ActiveDocument.Range.End - myTot

If caseCount > 0 Then WordBasic.EditUndo

' Now notes, if any

If Len(nts) > 0 Then

lnNow = Len(Replace(nts, thatBit, thatBit & "!"))

caseCount = caseCount + lnNow - lnWas

End If

my2Text = my2Text & "Case sensitive:" & Str(caseCount) & CR

If Timer > myTime + maxTime Then

my2Text = my2Text & CR & "Time out!"

GoTo printResult

End If

If doFormatCount = True Then

' Count bold italic

With rng.Find

.ClearFormatting

.MatchCase = False

.Font.Italic = True

.Font.Bold = True

.Execute Replace:=wdReplaceAll

End With

biCount = ActiveDocument.Range.End - myTot

If biCount > 0 Then

WordBasic.EditUndo

my2Text = my2Text & "Bold italic (main text only):" & Str(biCount) & CR

End If

' Count italic

With rng.Find

.ClearFormatting

.MatchCase = False

.Font.Italic = True

.Execute Replace:=wdReplaceAll

End With

iCount = ActiveDocument.Range.End - myTot

If iCount > 0 Then

WordBasic.EditUndo

my2Text = my2Text & "Italic (main text only):" & Str(iCount - biCount) & CR

End If

' Count bold

With rng.Find

.ClearFormatting

.Font.Bold = True

.Execute Replace:=wdReplaceAll

End With

bCount = ActiveDocument.Range.End - myTot

If bCount > 0 Then

WordBasic.EditUndo

my2Text = my2Text & "Bold (main text only):" & Str(bCount - biCount) & CR

End If

End If

If Timer > myTime + maxTime Then

my2Text = my2Text & CR & "Time out!"

GoTo printResult

End If

If doCountWhole = True Then

' Count as whole words (case sensitive)

thatBit = Replace(thatBit, "(", "\(")

thatBit = Replace(thatBit, ")", "\)")

thatBit = Replace(thatBit, "^p", "^13")

' Add text of the notes at the end of main text

If Len(nts) > 0 Then

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=""

End If

Set rng = ActiveDocument.Content

myTot = ActiveDocument.Range.End

With rng.Find

.ClearFormatting

.Text = "[!a-zA-Z]" & thatBit & "[!a-zA-Z]"

.Replacement.Text = "^& "

.MatchWholeWord = False

.MatchCase = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

wholeWdCaseCount = ActiveDocument.Range.End - myTot

If wholeWdCaseCount > 0 Then

WordBasic.EditUndo

my2Text = my2Text & "Whole words (case sensitive):" & \_

Str(wholeWdCaseCount) & CR

End If

With rng.Find

.ClearFormatting

.Text = ""

.Replacement.Text = ""

.MatchWholeWord = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

printResult:

Selection.End = oldStart

Selection.MoveRight Unit:=wdCharacter, Count:=1

Selection.Start = oldStart

Selection.End = oldEnd

If InANote = True Then

Selection.MoveDown Unit:=wdLine, Count:=lineJump

End If

'''msgBox original CountPhrase (for reference)

'myText = "Word searched: " & thisBit & CR2 & myText

'MsgBox myText, 0, "CountPhrase"

'With Selection.Find

' .Text = oldFind

' .Replacement.Text = oldReplace

' .MatchWildcards = False

'End With

''' msbBox for 2nd phrase (for reference)

'my2Text = "Word searched: " & thatBit & CR2 & my2Text

'MsgBox my2Text, 0, "CountTwoPhrases"

'With Selection.Find

' .Text = old2Find

' .Replacement.Text = old2Replace

' .MatchWildcards = False

'End With

'ActiveDocument.TrackRevisions = myTrack

''new message box combined

myText = "Word searched: " & thisBit & CR2 & myText

my2Text = "Word searched: " & thatBit & CR2 & my2Text

MsgBox myText & CR2 & my2Text, 0, "CountTwoPhrases"

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchWildcards = False

End With

With Selection.Find

.Text = old2Find

.Replacement.Text = old2Replace

.MatchWildcards = False

End With

'' back to new element for second word/phrsae

ActiveDocument.TrackRevisions = myTrack

End Sub

Sub Clipper()

' Paul Beverley - Version 06.02.21

' Collects, manipulates and pastes items from the ClipStore

myHighlight = 0

myHighlight = wdYellow

myClipFile = "zClipStore.docx"

myFolder = "C:\VirtualAcorn\VirtualRPC-SA\HardDisc4\MyFiles2\WIP\zzzTheBook\"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff"

maxClips = 12

CR = vbCr: CR2 = CR & CR

macroCopy = True

cs = ""

cs = cs & "+/- = switch formatting on/off|"

cs = cs & "s = store a copy of all clips into the BS|"

cs = cs & "r = restore all back clips to numbered|"

cs = cs & "n = create new blank clip|"

cs = cs & "|"

cs = cs & "n = create new blank clip|"

cs = cs & "|"

cs = cs & "p = paste this clip to target file|"

cs = cs & "c = copy this clip into the clipboard|"

cs = cs & "b = blank the working clip|"

cs = cs & "u = undo DictaFRedit changes|"

cs = cs & "x = delete all numbered clips!|"

cs = cs & " = |"

tf = ""

tf = tf & "+/- = switch formatting on/off|"

tf = tf & "s = store a copy of all clips in BS|"

tf = tf & "r = restore all back clips to numbered|"

tf = tf & "n = create new blank clip|"

tf = tf & "x = delete all numbered clips!|"

tf = tf & "|"

tf = tf & "<N°> = paste clip n into text|"

tf = tf & "<N°>+ = clip n with formatting|"

tf = tf & "<N°>- = clip n withOUT formatting|"

tf = tf & "|"

tf = tf & "<Enter> = paste same clip as last time|"

tf = tf & " = |"

tf = tf & " = |"

endPos = InStr(cs, "= |")

If endPos > 0 Then cs = Left(cs, endPos - 3)

cs = Replace(cs, "|", vbCr)

endPos = InStr(tf, "= |")

If endPos > 0 Then tf = Left(tf, endPos - 3)

tf = Replace(tf, "|", vbCr)

' Backstore items

Dim bStart(99) As Integer

Dim bEnd(99) As Integer

Dim bName(99) As String

' Main clip items

Dim cTitle(99) As Integer

Dim cStart(99) As Integer

Dim cEnd(99) As Integer

Dim cName(99) As String

' Remember start selection

Set rngWas = Selection.Range.Duplicate

Set startDoc = ActiveDocument

myFileName = startDoc.Name

dotPos = InStr(myFileName, ".")

If dotPos > 0 Then myFileName = Left(myFileName, dotPos - 1)

' Look for an open ClipStore file

On Error GoTo ReportIt

gottaStoreFile = False

For i = 1 To Application.Windows.Count

Set thisFile = Application.Windows(i).Document

If InStr(thisFile.Name, myClipFile) > 0 Then

Set clipDoc = Application.Windows(i).Document

gottaStoreFile = True

Exit For

End If

Next i

' If it's not open, load it

If gottaStoreFile = False Then

Documents.Open myFolder & myClipFile

Set clipDoc = ActiveDocument

End If

On Error GoTo 0

cursorInClipFile = (startDoc = clipDoc)

If cursorInClipFile Then

myPrompt = cs

Else

myPrompt = tf

End If

' Interpret the contents of the ClipStore

inBackStore = True

bNum = 0

For i = 1 To clipDoc.Paragraphs.Count

Set pa = clipDoc.Paragraphs(i).Range

tx = Replace(pa.Text, CR, "")

If Left(tx, 4) = "last" Then lastUsedText = Right(tx, 2)

If Left(tx, 3) = "max" Then maxClips = Val(Right(tx, 2))

If Left(tx, 3) = "for" Then textOnly = (Right(tx, 1) = "-")

If Left(tx, 4) = "999" Then

inBackStore = False

bEnd(bNum) = pa.Start - 1

endBackClips = pa.Start

startClips = pa.Start + 3

End If

If Left(tx, 3) = "000" Then

bEnd(bNum) = pa.Start - 1

startBackClips = pa.End

End If

If Left(tx, 1) = "\_" Then

If inBackStore = True Then

' store parameters for backstore items

bNum = bNum + 1

bStart(bNum) = pa.End

If bNum > 1 Then bEnd(bNum - 1) = pa.Start - 1

bNumMax = bNum

Else

' store parameters for clipstore items

cNum = Val(Right(pa.Text, 3))

cTitle(cNum) = pa.Start

cStart(cNum) = pa.End

If cNum > cNumMax Then cNumMax = cNum

End If

End If

cEnd(cNum) = pa.End

Next i

bEnd(0) = startBackClips

startClips = endBackClips + 4

endClips = clipDoc.Content.End

cEnd(cNum) = endClips

newClipNum = (cNum Mod maxClips) + 1

newClipNumText = Right(Trim(Str(100 + newClipNum)), 2)

' "What do you want to do today?!"

myInput = InputBox(myPrompt, "ClipPaste")

' Changes format/text-only status

If InStr("+-", myInput) > 0 And Len(myInput) > 0 Then

Set rng = clipDoc.Content

mySignPos = InStr(rng, "mat =")

rng.Start = mySignPos

rng.End = rng.Start + 7

mySignPos = InStr(rng, CR)

rng.Start = rng.Start + mySignPos - 2

rng.End = rng.Start + 1

rng.Text = myInput

Exit Sub

End If

' If the user is specifying formatting/text-only

If InStr(myInput, "+") > 0 Then

myInput = Replace(myInput, "+", "")

textOnly = False

End If

If InStr(myInput, "-") > 0 Then

myInput = Replace(myInput, "-", "")

textOnly = True

End If

If myInput = "s" Then

Set rng = clipDoc.Range(startClips, endClips)

rng.Copy

rng.MoveStart , -4

rng.Collapse wdCollapseStart

rs = rng.Start

rng.Paste

rng.Start = rng.End - 1

rng.Delete

rng.Start = rs

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\_^#^#"

.Wrap = wdFindStop

.Replacement.Text = "\_xx"

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Exit Sub

End If

If myInput = "x" Then

Beep

Set rng = clipDoc.Range(startClips, endClips)

rng.Select

myResponse = MsgBox("Delete ALL your numbered clips?!", \_

vbYesNoCancel, "ClipPaste")

If myResponse = vbYes Then Selection.Delete

Selection.InsertAfter Text:=CR

Selection.Collapse wdCollapseEnd

Exit Sub

End If

If myInput = "r" Then

Beep

Set rng = clipDoc.Range(startBackClips, endBackClips)

rng.Select

rng.Copy

Selection.EndKey Unit:=wdStory

Selection.Paste

Set rng = clipDoc.Range(endClips, Selection.End)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\_xx"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

cNum = cNumMax

Do While rng.Find.Found = True

endNow = rng.End

cNum = cNum + 1

newNumText = Right(Trim(Str(100 + cNum)), 2)

rng.Text = "\_" & newNumText

rng.Start = endNow

rng.Find.Execute

rng.Select

Loop

Exit Sub

End If

If InStr("ne", myInput) > 0 Then

' Has this new clip number already been used?

Set rng = clipDoc.Content

oldClipPos = InStr(rng.Text, "\_" & newClipNumText & CR)

If oldClipPos > 0 Then

Set rng = clipDoc.Range(cTitle(newClipNum), cEnd(newClipNum))

rng.Delete

End If

Set rng = clipDoc.Content

rng.Collapse wdCollapseEnd

' Add a new clip to numbered clip area ...

rng.InsertAfter Text:="\_\_" & newClipNumText & vbCr

rng.HighlightColorIndex = myHighlight

rng.Collapse wdCollapseEnd

rng.InsertAfter Text:=CR

rng.Select

Selection.EndKey Unit:=wdStory

Selection.MoveLeft , 1

If myInput = "n" Then Exit Sub

' Edit an email text

rng.Paste

End If

' Where is the cursor?

If cursorInClipFile = False Then GoTo cursorInTarget

' Doing things from the ClipStore file

If InStr("bpc", myInput) > 0 Then

myPos = rngWas.Start

For i = 1 To cNumMax

If myPos > cStart(i) - 5 And myPos < cEnd(i) + 1 Then Exit For

Next i

Set rngClip = clipDoc.Range(cStart(i), cEnd(i))

' Blank the working clip

If myInput = "b" Then

rngClip.MoveStart , 1

rngClip.Select

Selection.Delete

Selection.TypeText Text:=CR

Selection.MoveLeft , 1

Selection.TypeBackspace

Exit Sub

End If

rngClip.HighlightColorIndex = wdNoHighlight

rngClip.Copy

If myInput = "p" Then

st = rngClip.Start - 1

Set rng = clipDoc.Range(st, st)

rng.Expand wdParagraph

rng.MoveStart , 1

rng.MoveEnd , -4

myFileName = rng.Text

' Look for this file

gottaStoreFile = False

For i = 1 To Application.Windows.Count

Set thisFile = Application.Windows(i).Document

If InStr(thisFile.Name, myFileName) > 0 Then

Application.Windows(i).Document.Activate

Selection.Paste

gottaStoreFile = True

End If

Next i

If gottaStoreFile = False Then

Beep

myResponse = MsgBox("File " & myFileName & \_

" not open.", vbOKOnly, "ClipPaste")

End If

End If

If myInput = "c" Then

rngClip.Copy

Exit Sub

End If

Exit Sub

End If

' Doing things from the target file

cursorInTarget:

' Paste in numbered clip

If myInput = "" Then

myInputText = lastUsedText

Else

myInputText = Right(Trim(Str(100 + Val(myInput))), 2)

Set rngTest = clipDoc.Content

lastPos = InStr(rngTest, "lastClip")

If lastPos = 0 Then

rngTest.InsertBefore Text:="lastClip = " & myInputText & CR

Beep

rngTest.Select

Exit Sub

Else

rngTest.Start = lastPos

rngTest.Collapse wdCollapseStart

rngTest.Expand wdParagraph

rngTest.End = rngTest.End - 1

rngTest.Start = rngTest.End - 2

rngTest.Text = myInputText

End If

End If

myPos = InStr(clipDoc.Content, "\_" & myInputText & CR)

If myPos = 0 Then

Beep

myResponse = MsgBox("Can't find clip number: " & \_

myInputText, vbOKOnly, "ClipPaste")

Exit Sub

End If

For i = 1 To cNumMax

If myPos > cStart(i) - 5 And myPos < cEnd(i) + 1 Then Exit For

Next i

Set rngClip = clipDoc.Range(cStart(i), cEnd(i) - 1)

rngClip.Copy

startDoc.Activate

If textOnly = True Then

Selection.PasteSpecial DataType:=wdPasteText

If Right(rngClip.Text, 1) = CR Then \_

Selection.TypeText Text:=CR

Else

Selection.Paste

End If

Exit Sub

ReportIt:

If Err.Number = 5174 Then

Err.Clear

Beep

myClipFile = Replace(myClipFile, ".docx", "")

myResponse = MsgBox("Can't find your clipstore file: """ \_

& myClipFile & """", vbOKOnly, "ClipPaste")

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub LoadFilesInList()**

' Paul Beverley - Version 29.06.22

' Loads a series of files in a Word file list

myMarker = "!!Here!!"

' Loads the first file in the list

' Searches for the text in myMarker

' Loads each of the other files

' Jumps back to the first file

Dim myFile(100) As String

numFiles = 0

For Each myPara In ActiveDocument.Paragraphs

myName = myPara.Range.Text

Debug.Print myName

If InStr(myName, ".docx") > 0 Then

numFiles = numFiles + 1

myName = Replace(myName, vbCr, "")

myFile(numFiles) = myName

DoEvents

End If

Next myPara

Documents.Open fileName:=myFile(1)

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myMarker

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute

DoEvents

End With

Set myLeadFile = ActiveDocument

For i = 2 To numFiles

Documents.Open fileName:=myFile(i)

DoEvents

Next i

myLeadFile.Activate

End Sub

**Sub CopyFullFilename()**

' Paul Beverley - Version 29.06.22

' Loads full filename of current document into clipboard

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=ActiveDocument.FullName & vbCr

Selection.MoveLeft , 1

Selection.Expand wdParagraph

Selection.Font.Reset

Selection.Style = wdStyleNormal

Selection.Copy

Selection.HomeKey Unit:=wdStory

End Sub

**Sub ColonUppercaseAll()**

' Paul Beverley - Version 23.07.22

' Changes the initial letter after every colon to lowercase

showHowMany = True

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-zA-Z]: [a-z]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

endNow = rng.End

rng.MoveStart , 2

rng.Text = UCase(rng.Text)

If myCount Mod 50 = 0 Then rng.Select

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Selection.HomeKey Unit:=wdStory

If showHowMany = True Then MsgBox "Changed: " & myCount

End Sub

**Sub ColonLowercaseAll()**

' Paul Beverley - Version 17.02.23

' Changes the initial letter after every colon to lowercase

doHighlight = True

myColour = wdTurquoise

showHowMany = True

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-zA-Z]: [A-Z]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

endNow = rng.End

rng.MoveStart , 2

If doHighlight = True Then rng.HighlightColorIndex = myColour

rng.Text = LCase(rng.Text)

If myCount Mod 50 = 0 Then rng.Select

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Selection.HomeKey Unit:=wdStory

If showHowMany = True Then MsgBox "Changed: " & myCount

End Sub

**Sub OddQuoteSpacingCorrectSingle()**

' Paul Beverley - Version 26.07.22

' Corrects the odd spacing on double quotation marks

myQuote = "'"

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myQuote

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.MoveStart , -1

rng.MoveEnd , 1

If Right(rng, 1) = vbCr Then rng.MoveEnd , -1

rng.Cut

rng.PasteSpecial DataType:=wdPasteText

rng.Start = endNow - 1

rng.Font.Superscript = False

rng.Start = endNow + 2

rng.End = endNow + 2

rng.Find.Execute

DoEvents

DoEvents

q = q + 1

If q Mod 100 = 0 Then rng.Select

Loop

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myQuote

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = myQuote

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub ProperNounAlyse()**

' Paul Beverley - Version 26.07.22

' Analyses similar proper nouns

minLengthCheck = 4

includeAcronyms = True

ignoreWords = "Also And Even Every From Here However That Through "

ignoreWords = ignoreWords & " The This There Those Their They Then These"

ignoreWords = ignoreWords & " What Whatever Where Wherever When Whenever "

ignoreWords = ignoreWords & " Whatever Where Why With Without Where Wherever "

ignoreWords = ignoreWords & " When Whenever Where Why With Without Your You"

similarChars = "bb,b; b,p; sch,sh; ch,sh; c,k; ph,f; ss,z; s,z;" & \_

" mp,m; ll,l; nn,n; nd,n; nt,n;"

' With non-English languages, you might need to make this False

ignorePlurals = True

myScreenOff = True

ignoreWords = Replace(ignoreWords, " ", " ")

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox(" ProperNounAlyse" & vbCr & vbCr & \_

"Analyse this document?", vbQuestion \_

+ vbYesNoCancel, "ProperNounAlyse")

If myResponse <> vbYes Then Exit Sub

End If

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

myDummy = ChrW(222)

For i = 1 To 100

spcs = " " & spcs

Next i

dummyText = ChrW(197) & "zzzx "

For i = 65 To 90

dummyText = dummyText & ChrW(i) & "zzzz "

Next i

checkFinalLetters = True

' checkFinalLetters = False

' Grey on word only

thisHighlight = wdGray25

doMissingLetter = True

' doMissingLetter = False

' Bold And blue

switchTest = True

' switchTest = False

' double strikethrough

doSimilarLetters = True

' doSimilarLetters = False

' various highlight colours + underline

doVowelTest = True

' doVowelTest = False

' various highlight colours + italic

' These last two tests cycle through these colours:

maxCol = 6

ReDim myCol(maxCol) As Integer

myCol(1) = wdYellow

myCol(2) = wdBrightGreen

myCol(3) = wdTurquoise

myCol(4) = wdRed

myCol(5) = wdPink

myCol(6) = wdGray25

colcode = 0

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = wdGray25

leadDots = " . . . "

title1 = "Proper noun list"

title2 = "Proper noun queries"

CR = vbCr: CR2 = CR & CR

convCharsUC = "AAAAAA..EEEEIIII..OOOOO..UUUU" & \_

"...aaaaaa..eeeeiiiio.ooooo.ouuuu......"

convCharsLC = LCase(convCharsUC)

timeStart = Timer

' collect notes text, if any

endText = ""

footText = ""

If ActiveDocument.Endnotes.Count > 0 Then

endText = ActiveDocument.StoryRanges(wdEndnotesStory).Text

End If

If ActiveDocument.Footnotes.Count > 0 Then

footText = ActiveDocument.StoryRanges(wdFootnotesStory).Text

End If

' collect text in all the textboxes (if any)

sh = ActiveDocument.Shapes.Count

If sh > 0 Then

ReDim shText(sh)

i = 0

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

i = i + 1

shText(i) = shp.TextFrame.TextRange.Text

End If

End If

Next

shCount = i

End If

' Create various documents

Set rng = ActiveDocument.Content

Documents.Add

Set firstDoc = ActiveDocument

Set fnl = ActiveDocument.Content

Documents.Add

Set tempDoc = ActiveDocument

Set tmp = ActiveDocument.Content

Documents.Add

Set allText = ActiveDocument

Selection.TypeText dummyText & vbCr

Selection.FormattedText = rng.FormattedText

Selection.Collapse wdCollapseEnd

' Add notes + shape text

Selection.TypeText endText & CR & footText & CR

If shCount > 0 Then

For i = 1 To shCount

Selection.TypeText shText(i) & CR

Next i

End If

Selection.HomeKey Unit:=wdStory

Set rng = allText.Content

rng.Revisions.AcceptAll

DoEvents

StatusBar = spcs & "Preparing copied file - 1"

DoEvents

' Delete struck-through text

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.MatchWildcards = False

.Font.StrikeThrough = True

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "["

.MatchWildcards = False

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

' Remove strange unicode characters

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & ChrW(&HA000) & "-" & ChrW(&HD6FF) & "]{1,}"

.MatchWildcards = True

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

DoEvents

StatusBar = spcs & "Preparing copied file - 2"

DoEvents

' Cut all and replace as pure text

Set rng = allText.Content

tmp.FormattedText = rng.FormattedText

rng.Text = tmp.Text

tmp.Delete

DoEvents

StatusBar = spcs & "Preparing copied file - 3"

' Use qqq for apostrophe

Set rng = allText.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "n" & ChrW(8217) & "t"

.MatchWildcards = False

.Replacement.Text = "nqqqt"

.Execute Replace:=wdReplaceAll

End With

' Use qq for apostrophe

With rng.Find

.Text = "O'"

.MatchCase = True

.Replacement.Text = "Oqqq"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Find initial cap words

DoEvents

StatusBar = spcs & "Preparing copied file - 4"

DoEvents

myChopNum = minLengthCheck - 2

If myChop < 1 Then myChop = 1

myChop = Trim(Str(myChopNum))

myFind = "<[A-Z][a-z][a-zA-Z]{" & myChop & ",}"

If includeAcronyms = True Then myFind = \_

"<[A-Z][a-zA-Z][a-zA-Z]{" & myChop & ",}"

Set rng = allText.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.MatchWildcards = True

.MatchCase = True

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Replacement.Font.StrikeThrough = True

.Execute Replace:=wdReplaceAll

End With

' Delete all non-strikethrough words

DoEvents

StatusBar = spcs & "Preparing copied file - 5"

DoEvents

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = False

.MatchWildcards = False

.MatchCase = True

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

' Delete the unwanted "proper nouns"

DoEvents

StatusBar = spcs & "Preparing copied file - 6"

igWords = Split(Trim(ignoreWords), " ")

For Each wd In igWords

Set rng = allText.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = wd & "^p"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next wd

StatusBar = spcs & "Sorting whole file"

DoEvents

i = 0

For ch = 65 To 90

For Each myPara In allText.Paragraphs

If Asc(myPara.Range) = ch Then

DoEvents

myPara.Range.Font.StrikeThrough = False

tmp.InsertAfter myPara.Range.Text

End If

Next myPara

tmp.InsertAfter Text:="Zzzzz" & CR

Set rng = tempDoc.Content

rng.Sort SortOrder:=wdSortOrderAscending, CaseSensitive:=True

' delete initial blank line

If Len(tempDoc.Paragraphs(1)) < 3 Then \_

tempDoc.Paragraphs(1).Range.Delete

' Create a frequency for each highlighted word

thisWord = ""

myCount = 0

For Each myPara In tempDoc.Paragraphs

Set rng = myPara.Range.Words(1)

DoEvents

nextWord = rng

If nextWord <> thisWord Then

' This is a new word

If Len(thisWord) > 1 Then

fnl.InsertAfter Text:=thisWord \_

& leadDots & Trim(Str(myCount)) & CR

End If

thisWord = nextWord

myCount = 1

Else

myCount = myCount + 1

End If

If nextWord = "Zzzzz" Then Exit For

i = i + 1:

If i Mod 400 = 4 Then

DoEvents

prmt = Left(thisWord, 1) & " "

prmt = prmt & prmt & prmt & prmt

StatusBar = spcs & \_

"Preparing words for frequency list - " & prmt

DoEvents

End If

Next myPara

' Remove all words except frequency counts

Set rng = tempDoc.Content

rng.Delete

Next ch

' Find any unaccounted-for words, e.g. Ångstrom

For Each myPara In allText.Paragraphs

If myPara.Range.Words(1).Font.StrikeThrough = True Then

tmp.InsertAfter myPara.Range.Text

End If

Next myPara

tempDoc.Close SaveChanges:=False

allText.Close SaveChanges:=False

firstDoc.Activate

' Remove blank lines

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{2,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Resort case insensitively

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

CaseSensitive:=False

' Delete rubbish from top and bottom of list

Do

Set rng = ActiveDocument.Paragraphs(1).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until myLen > 9

Do

lastLine = ActiveDocument.Paragraphs.Count

Set rng = ActiveDocument.Paragraphs(lastLine).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until Len(rng.Text) >= 2

' Word list now has freq. count.

Do

lastLine = ActiveDocument.Paragraphs.Count

Set rng = ActiveDocument.Paragraphs(lastLine).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until Len(rng.Text) >= 2

' Create another copy for doing extra tests

Set rng = ActiveDocument.Content

Documents.Add

Set finalList = ActiveDocument

finalList.Range.Text = rng.Text

Selection.HomeKey Unit:=wdStory

' Prepare data for other tests

numWords = ActiveDocument.Paragraphs.Count

For i = 1 To numWords

aWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(aWord)

thisChar = ChrW(n)

If n > 129 Then

If n >= 217 Then aWord = Replace(aWord, thisChar, "U")

If n >= 210 Then aWord = Replace(aWord, thisChar, "O")

If n >= 204 Then aWord = Replace(aWord, thisChar, "I")

If n >= 200 Then aWord = Replace(aWord, thisChar, "E")

If n >= 192 Then aWord = Replace(aWord, thisChar, "A")

End If

allWords = allWords & aWord

jmp = 100

If i Mod jmp = 1 Then

pq = pq + 1

DoEvents

StatusBar = spcs & \_

"Preparing data for other tests - 1 - " & pq

DoEvents

End If

Next i

' ...for the vowel test below

DoEvents

StatusBar = spcs & "Preparing data for other tests - 2"

DoEvents

noVowelWords = " " & allWords

noVowelWords = Replace(noVowelWords, " A", "\_1")

noVowelWords = Replace(noVowelWords, " E", "\_2")

noVowelWords = Replace(noVowelWords, " I", "\_3")

noVowelWords = Replace(noVowelWords, " O", "\_4")

noVowelWords = Replace(noVowelWords, " U", "\_5")

noVowelWords = Replace(noVowelWords, " Y", "\_6")

For k = 2 To Len(noVowelWords) - 1

thisChar = Mid(noVowelWords, k, 1)

n = AscW(thisChar)

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsLC, n - 191, 1)

If myNewChar <> "." Then noVowelWords = \_

Replace(noVowelWords, thisChar, myNewChar)

End If

Next k

noVowelWords = Replace(noVowelWords, "a", "")

noVowelWords = Replace(noVowelWords, "e", "")

noVowelWords = Replace(noVowelWords, "i", "")

noVowelWords = Replace(noVowelWords, "o", "")

noVowelWords = Replace(noVowelWords, "u", "")

noVowelWords = Replace(noVowelWords, "y", "")

noVowelWords = Replace(noVowelWords, "A", "")

noVowelWords = Replace(noVowelWords, "E", "")

noVowelWords = Replace(noVowelWords, "I", "")

noVowelWords = Replace(noVowelWords, "O", "")

noVowelWords = Replace(noVowelWords, "U", "")

noVowelWords = Replace(noVowelWords, "Y", "")

noVowelWords = Replace(noVowelWords, "\_1", " A")

noVowelWords = Replace(noVowelWords, "\_2", " E")

noVowelWords = Replace(noVowelWords, "\_3", " I")

noVowelWords = Replace(noVowelWords, "\_4", " O")

noVowelWords = Replace(noVowelWords, "\_5", " U")

noVowelWords = Replace(noVowelWords, "\_6", " Y")

' ...for the similar words test

DoEvents

StatusBar = spcs & "Preparing data for other tests - 3"

DoEvents

similarAllWords = " " & LCase(allWords)

similarChars = Replace(similarChars, " ", "")

sChars = Replace(similarChars, " ", "")

Do

commaPos = InStr(sChars, ",")

charWas = Left(sChars, commaPos - 1)

sChars = Mid(sChars, commaPos + 1)

semicolonPos = InStr(sChars, ";")

charNew = Left(sChars, semicolonPos - 1)

sChars = Mid(sChars, semicolonPos + 1)

similarAllWords = Replace(similarAllWords, charWas, charNew)

Loop Until Len(sChars) < 2

' Change all the accented characters to non-accented

DoEvents

StatusBar = spcs & "Preparing data for other tests - 4"

DoEvents

sWd = similarAllWords

For k = 1 To Len(sWd) - 1

thisChar = Mid(sWd, k, 1)

n = AscW(thisChar)

myNewChar = "."

If n > 191 And n < 256 Then

myNewChar = Mid(convCharsLC, n - 191, 1)

If myNewChar <> "." Then sWd = Replace(sWd, \_

thisChar, myNewChar)

End If

Next k

similarAllWords = sWd

' Catch words with only the final two letters the same

i = 0

If checkFinalLetters = True Then

For Each myPara In ActiveDocument.Paragraphs

gotOne = False

myWord = Trim(myPara.Range.Words(1))

myLen = Len(myWord)

If myLen > 6 Then

myTarget = "^p" & Left(myWord, myLen - 2) & "^$^$ "

myCut = 2

Else

myTarget = "^p" & Left(myWord, myLen - 1) & "^$ "

myCut = 1

End If

Set rng = ActiveDocument.Content

rng.Start = myPara.Range.End - 3

rng.Collapse wdCollapseStart

With rng.Find

.Replacement.ClearFormatting

.ClearFormatting

.Text = myTarget

.Replacement.Text = ""

.Forward = True

.MatchCase = True

.MatchWildcards = False

.Wrap = wdFindStop

End With

rng.Find.Execute

Do While rng.Find.Found

gotOne = True

rng.MoveStart 1

rng.End = rng.Start + myLen - myCut

rng.HighlightColorIndex = thisHighlight

rng.Font.Bold = True

rng.Find.Execute

Loop

If gotOne = True Then

Set rng = myPara.Range.Words(1)

rng.End = rng.Start + myLen - myCut

rng.HighlightColorIndex = thisHighlight

rng.Font.Bold = True

End If

i = i + 1

If i Mod 100 = 1 Then

DoEvents

StatusBar = spcs & "Doing test (5) on " & myWord

DoEvents

End If

Next myPara

End If

If doMissingLetter = True Then

' Start of test

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Change the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (4) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' Check if word reappears with one letter missing (1)

For k = 2 To Len(myWord) - 1

testWord = " " & Left(myWord, k - 1) & Mid(myWord, k + 1)

wordPos = InStr(allWords, testWord)

If wordPos > 0 Then

lastLetter = Mid(myWord, Len(myWord) - 1, 1)

' but not "s" at the end, unless it's a spelling error

If lastLetter = "s" Then

ignoreIt = (Application.CheckSpelling(myWord, \_

MainDictionary:=Languages(Selection.LanguageID).NameLocal) \_

= True)

Else

ignoreIt = False

End If

If ignoreIt = False And ignorePlurals = True Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

' mark the pair

leftBit = Left(allWords, InStr(allWords, testWord) \_

+ Len(testWord) - 1)

j = Len(leftBit) - Len(Replace(leftBit, " ", ""))

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Bold = True

rng.Font.Color = wdColorBlue

Set rng = ActiveDocument.Paragraphs(j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Bold = True

rng.Font.Color = wdColorBlue

End If

End If

Next k

If Left(myWord, 2) = "Mc" Or Left(myWord, 3) = "Mac" Or \_

Left(myWord, 3) = "Mag" Then

McList = McList & ActiveDocument.Paragraphs(i).Range

End If

Next i

End If

If doSimilarLetters = True Then

doneWords = ""

doneSimilarWords = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Change the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (3) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check similar spellings: Perutz/Peruts or Chebyshev/Chevychev

similarWord = " " & LCase(myWord)

sChars = similarChars

Do

commaPos = InStr(sChars, ",")

charWas = Left(sChars, commaPos - 1)

sChars = Mid(sChars, commaPos + 1)

semicolonPos = InStr(sChars, ";")

charNew = Left(sChars, semicolonPos - 1)

sChars = Mid(sChars, semicolonPos + 1)

similarWord = Replace(similarWord, charWas, charNew)

Loop Until Len(sChars) < 2

' Change all the accented characters to non-accented

For k = 1 To Len(myWord) - 1

thisChar = Mid(myWord, k, 1)

n = AscW(thisChar)

If n > 191 And n < 256 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

Next k

similarAllWords = Mid(similarAllWords, Len(similarWord))

theseWords = similarAllWords

If InStr(doneSimilarWords, similarWord) = 0 And \_

InStr(theseWords, similarWord) > 0 Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Underline = True

doneSimilarWords = doneSimilarWords & similarWord

' search through all the following words

theseWords = similarAllWords

For j = 1 To numWords - i

spPos = InStr(Trim(theseWords) & " ", " ")

If Left(theseWords, spPos + 1) = similarWord Then

Set rng = ActiveDocument.Paragraphs(i + j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Underline = True

End If

theseWords = Mid(theseWords, spPos + 1)

capThisLetter = Mid(theseWords, 2, 1)

If capThisLetter <> LCase(captestLetters) Then Exit For

Next j

End If

Next i

End If

If switchTest = True Then

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Change the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (2) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check for switched chars

wordLen = Len(myWord) - 1

For k = 1 To Len(myWord) - 3

otherWord = Left(myWord, k) & Mid(myWord, k + 2, 1) & \_

Mid(myWord, k + 1, 1) & Mid(myWord, k + 3)

wordPos = InStr(testWords, otherWord)

If wordPos > 0 Then

' Find the position of the matching word

matchWord = Mid(testWords, wordPos, Len(myWord))

leftBit = Left(allWords, InStr(allWords, matchWord) + 1)

j = Len(leftBit) - Len(Replace(leftBit, " ", "")) + 1

ActiveDocument.Paragraphs(i).Range.Font.DoubleStrikeThrough \_

= True

ActiveDocument.Paragraphs(i).Range.HighlightColorIndex \_

= thisCol

ActiveDocument.Paragraphs(j).Range.Font.DoubleStrikeThrough \_

= True

ActiveDocument.Paragraphs(j).Range.HighlightColorIndex \_

= thisCol

End If

Next k

Next i

End If

If doVowelTest = True Then

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Change the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (1) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check if there's a word with different vowels

otherWord = " " & Replace(myWord, "a", "")

otherWord = Replace(otherWord, "e", "")

otherWord = Replace(otherWord, "i", "")

otherWord = Replace(otherWord, "o", "")

otherWord = Replace(otherWord, "u", "")

otherWord = Replace(otherWord, "y", "")

' Delete all the accented characters

For k = 3 To Len(otherWord) - 1

thisChar = Mid(otherWord, k, 1)

n = AscW(thisChar)

If InStr("AEIOUY", thisChar) > 0 Then

otherWord = Left(otherWord, k - 1) & "=" & Mid(otherWord, k + 1)

Else

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then

otherWord = Replace(otherWord, thisChar, "=")

End If

End If

End If

Next k

otherWord = Replace(otherWord, "=", "")

' otherWord is now the word under test (vowel-less)

otherWord = Replace(otherWord, ".", "")

noVowelWords = Mid(noVowelWords, Len(otherWord))

If Left(noVowelWords, 1) <> " " Then noVowelWords = \_

" " & noVowelWords

theseWords = noVowelWords

wordPos = InStr(noVowelWords, otherWord)

If InStr(doneWords, otherWord) = 0 And wordPos > 0 Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Italic = True

doneWords = doneWords & otherWord

For j = 1 To numWords - i

spPos = InStr(Trim(theseWords) & " ", " ")

firstWord = Left(theseWords, spPos + 1)

theseWords = Mid(theseWords, spPos + 1)

If firstWord = otherWord Then

Set rng = ActiveDocument.Paragraphs(i + j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Italic = True

End If

capThisLetter = Mid(theseWords, 2, 1)

If capThisLetter > "" And capThisLetter <> \_

captestLetters Then Exit For

Next j

End If

Next i

End If

finishOff:

Selection.EndKey Unit:=wdStory

Selection.TypeText CR2 & McList

Selection.HomeKey Unit:=wdStory

Selection.TypeText title1 & CR

Do

Selection.Expand wdParagraph

If Len(Selection) < 3 Or LCase(Selection) = \_

UCase(Selection) Then Selection.Delete

Loop Until LCase(Selection) <> UCase(Selection)

Selection.HomeKey Unit:=wdStory, Extend:=wdExtend

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

' Restore apostrophes

Set rng = finalList.Range

With rng.Find

.Text = "qqq"

.MatchCase = False

.Replacement.Text = "'"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Find first highlight

Set rng = finalList.Content

With rng.Find

.Text = "Zzzzz"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

Set rng = finalList.Content

With rng.Find

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

rng.Select

Selection.Collapse wdCollapseStart

Set finalList = ActiveDocument

firstDoc.Activate

' Find sets of sounds-like words

StatusBar = spcs & "Sounds-like tests"

k = 0

For Each myPara In ActiveDocument.Paragraphs

myWord = Trim(myPara.Range.Words(1))

k = k + 1

If k Mod 40 = 1 Then

DoEvents

StatusBar = spcs & "Sounds-like test: " & myWord

DoEvents

End If

hasAccent = False

For i = 1 To Len(myWord)

ascChar = AscW(Mid(myWord, i))

If ascChar > 128 Or ascChar = Asc("?") Then hasAccent = True

Next i

' Go and find the first sounds-like word

initLetter = Left(myWord, 1)

If Len(myWord) > 2 And myPara.Range.HighlightColorIndex > 0 And \_

hasAccent = False And InStr(allSets, myWord & leadDots) \_

= 0 Then

Set rng = ActiveDocument.Content

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myWord

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = False

.MatchSoundsLike = True

.Execute

End With

Set myPara = rng.Paragraphs(1).Range

rng.Collapse wdCollapseEnd

Loop Until Left(myPara, 1) = initLetter

setOfWords = myPara

gottaSet = False

rng.Collapse wdCollapseEnd

rng.Find.Execute

Do While rng.Find.Found = True

Set myPara = rng.Paragraphs(1).Range

If Left(myPara, 1) = initLetter Then

gottaSet = True

setOfWords = setOfWords & myPara

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

If gottaSet = True Then allSets = allSets & setOfWords & CR

End If

Next myPara

Selection.WholeStory

If Len(allSets) < 2 Then

Selection.TypeText "None found with this test"

Else

Selection.TypeText allSets

End If

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Proper nouns by sound" & CR

Selection.HomeKey Unit:=wdStory, Extend:=wdExtend

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = 0

rng.Copy

ActiveDocument.Close SaveChanges:=False

finalList.Activate

' Remove highlighting from second half of words

' that are only case changes of one another

totParas = ActiveDocument.Paragraphs.Count

For i = 1 To totParas - 1

A = Trim(ActiveDocument.Paragraphs(i).Range.Words(1))

b = Trim(ActiveDocument.Paragraphs(i + 1).Range.Words(1))

A = Mid(A, 2)

b = Mid(b, 2)

If LCase(A) = LCase(b) And Len(A) > 2 Then

If (UCase(A) = A And LCase(b) = b) Or (UCase(b) = b And \_

LCase(A) = A) Then

ActiveDocument.Paragraphs(i).Range.Words(1).HighlightColorIndex = 0

ActiveDocument.Paragraphs(i + 1).Range.Words(1).HighlightColorIndex \_

= 0

End If

End If

If i Mod 50 = 0 Then

DoEvents

StatusBar = spcs & "Final checks: " & totParas - i

DoEvents

End If

Next i

myOnames = ""

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13O[!a-z]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchSoundsLike = False

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rng.Collapse wdCollapseEnd

rng.Expand wdWord

wd = Mid(rng.Text, 3)

rng.Expand wdParagraph

pa = rng.Text

Set rng2 = ActiveDocument.Content

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13" & wd

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

If rng2.Find.Found Then

rng2.Collapse wdCollapseEnd

rng2.Expand wdParagraph

pa2 = rng2.Text

myOnames = myOnames & pa2 & pa & vbCr

End If

rng.Collapse wdCollapseEnd

rng.End = rng.End - 2

rng.Find.Execute

Loop

If myOnames > "" Then

Selection.EndKey Unit:=wdStory

Selection.TypeText "Possible O'<something> errors" & vbCr

Selection.MoveUp , 1

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

Selection.EndKey Unit:=wdStory

Selection.TypeText myOnames

Selection.HomeKey Unit:=wdStory

End If

Set rng = ActiveDocument.Content

finalList.Activate

Selection.EndKey Unit:=wdStory

Selection.TypeText vbCr & vbCr & vbCr

Selection.Paste

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myDummy

.Wrap = wdFindContinue

.Replacement.Text = " "

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^$zzz^$" & leadDots & "1" & vbCr

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Clear clipboard

Set rng = ActiveDocument.Content

rng.End = 2

rng.Copy

Set finalList = ActiveDocument

StatusBar = "Creating queries list"

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Set queriesDoc = ActiveDocument

ActiveDocument.Paragraphs(1).Range.Delete

Set rng = ActiveDocument.Content

rng.Font.StrikeThrough = True

For Each myPara In ActiveDocument.Paragraphs

Set ch = myPara.Range.Characters(1)

chCol = ch.HighlightColorIndex

If chCol > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

myLen = Len(myPara.Range.Text)

If myLen > 4 Then

If chCol > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

Set che = myPara.Range.Characters(myLen - 2)

If che.HighlightColorIndex > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

End If

Next myPara

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = True

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

rng.Font.StrikeThrough = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{3,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p^p"

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

For Each myPara In ActiveDocument.Paragraphs

myText = myPara.Range.Text

If Len(myText) > 4 Then

Set ch = myPara.Range.Characters(1)

numChars = myPara.Range.Characters.Count

Set myEnd = myPara.Range.Characters(numChars)

colNum = ch.HighlightColorIndex Mod 8

If ch.Font.Bold = True Then

myTxt = "qcqc " & Str(colNum + 1) & " = zczc"

Else

myTxt = "qcqc zczc"

End If

If ch.Font.Underline > 0 And colNum > 0 Then

myBit = "\* "

myTxt = Replace(myTxt, " = ", "")

Else

myBit = ""

End If

myPara.Range.InsertBefore myBit & myTxt

If ch.Font.Italic = True Then

myEnd.InsertBefore "qpqp= " & Chr(65 + colNum)

End If

End If

i = i + 1

If i Mod 20 = 0 And Len(myText) > 4 Then

myText = Replace(myText, vbCr, "")

StatusBar = spcs & "Creating queries list: " & myText

End If

DoEvents

Next myPara

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\\* qcqc(\*)zczc"

.Wrap = wdFindContinue

.Replacement.Text = "\* \1^t"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "qcqc(\*)zczc"

.Wrap = wdFindContinue

.Replacement.Text = "\1^t"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "qpqp(\*)^13"

.Replacement.Text = "^t\1^p"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "= ^$"

.Replacement.Text = ""

.Replacement.Font.Bold = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

rng.Font.Bold = False

rng.Font.Italic = False

rng.Font.DoubleStrikeThrough = False

rng.Font.Underline = False

rng.Font.Color = wdColorBlack

Selection.HomeKey Unit:=wdStory

Selection.TypeText title2 & CR

Set rng = ActiveDocument.Content.Paragraphs(2).Range

If rng.Text = vbCr Then rng.Delete

Set rng = ActiveDocument.Content.Paragraphs(1).Range

rng.Style = ActiveDocument.Styles(wdStyleHeading1)

StatusBar = " "

Options.DefaultHighlightColorIndex = oldColour

lighterColour = wdGray25

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "= ^$"

.Replacement.Text = ""

.Replacement.Font.ColorIndex = lighterColour

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^# ="

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

DoEvents

End With

If doingSeveralMacros = False Then

Debug.Print Timer - timeStart

myTime = (Int(10 \* (Timer - timeStart) / 60) / 10)

Beep

If myTime > 0 Then MsgBox myTime & " minutes"

Else

FUT.Activate

End If

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub StylesPaneCustomize()**

' Paul Beverley - Version 12.01.21

' Moves the Styles pane where and how you want

h = 600

w = 400

' doSetUp = True

doSetUp = False

If doSetUp = True Then

w = Application.CommandBars("Styles").Width

h = Application.CommandBars("Styles").Height

MsgBox "H: " & h & vbCr & vbCr & "W: " & w

Exit Sub

End If

'Application.CommandBars("Styles").Position = msoBarRight

'Application.CommandBars("Styles").Position = msoBarLeft

Application.CommandBars("Styles").Position = msoBarFloating

Application.CommandBars("Styles").Height = h

Application.CommandBars("Styles").Width = w

End Sub

**Sub WelcheKorrigiere()**

' Paul Beverley - Version 04.03.24

' Finds "welche" after a comma and corrects it

myColour = wdYellow

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ",[ acdhifmrtuü]{1,}welch[emnrs]{1,2}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

' Note where the end of the found item is

endNow = rng.End

rng.MoveStart , 2

If myCount Mod 10 = 1 Then rng.Select

rng.HighlightColorIndex = myColour

Select Case rng.Text

Case "welcher": rng.Text = "der"

Case "auf welcher": rng.MoveStart , 4: rng.Text = "der"

Case "mit welcher": rng.MoveStart , 4: rng.Text = "der"

Case "welches": rng.Text = "das"

Case "auf welches": rng.MoveStart , 4: rng.Text = "das"

Case "durch welches": rng.MoveStart , 6: rng.Text = "das"

Case "für welches": rng.MoveStart , 4: rng.Text = "das"

Case "welchen": rng.Text = "denen"

Case "auf welchen": rng.MoveStart , 4: rng.Text = "den"

Case "mit welchen": rng.MoveStart , 4: rng.Text = "denen"

Case "durch welchen": rng.MoveStart , 6: rng.Text = "den"

Case "für welchen": rng.MoveStart , 4: rng.Text = "den"

Case "welchem": rng.Text = "dem"

Case "auf welchem": rng.MoveStart , 4: rng.Text = "dem"

Case "mit welchem": rng.MoveStart , 4: rng.Text = "dem"

Case "welche": rng.Text = "die"

Case "auf welche": rng.MoveStart , 4: rng.Text = "die"

Case "durch welche": rng.MoveStart , 6: rng.Text = "die"

Case "für welche": rng.MoveStart , 4: rng.Text = "die"

End Select

rng.Start = endNow

rng.End = endNow

rng.Find.Execute

DoEvents

Loop

rng.Select

Selection.Collapse wdCollapseEnd

MsgBox "Changed: " & myCount

End Sub

**Sub AllCapsToItalic()**

' Paul Beverley - Version 08.08.22

' Finds words in all caps and makes them italic

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z]{2,}>"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

endNow = rng.End

rng.Text = LCase(rng.Text)

rng.Font.Italic = True

rng.Select

rng.Start = endNow

rng.End = endNow

rng.Find.Execute

DoEvents

Loop

MsgBox "Changed: " & myCount

End Sub

**Sub AllCapsToItalicNext()**

' Paul Beverley - Version 11.08.22

' Finds next word in all caps and makes it italic

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z]{2,}>"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

If Selection.Find.Found = True Then

Selection.Text = LCase(Selection.Text)

Selection.Font.Italic = True

Selection.Collapse wdCollapseEnd

Else

Beep

End If

End Sub

**Sub ItalicHighlightNotQuotes()**

' Paul Beverley - Version 31.08.22

' Finds italic text (but not in quotes) and highlights it

myHighlightColor = wdYellow

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

Set rngNext = rng.Duplicate

rngNext.Collapse wdCollapseEnd

With rngNext.Find

.ClearFormatting

.Text = """"

.Execute

End With

If rngNext.Find.Found Then

If AscW(rngNext.Text) = 8220 Then

rng.HighlightColorIndex = myHighlightColor

myCount = myCount + 1

End If

Else

rng.HighlightColorIndex = myHighlightColor

myCount = myCount + 1

End If

If myCount Mod 5 = 0 Then rng.Select

rng.Collapse wdCollapseEnd

With rng.Find

.Text = ""

.Font.Italic = True

.Execute

End With

DoEvents

Loop

rng.Select

MsgBox "Changed: " & myCount

End Sub

**Sub AsWellAsToAnd()**

' Paul Beverley - Version 24.09.22

' Changes 'as well as' to 'and', and the comma, if necessary

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Start = Selection.Start

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ", as well as"

.Replacement.Text = " and"

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

.Text = "as well as"

.Replacement.Text = "and"

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

End Sub

**Sub HighlightMissingDialoguePunctuation()**

' Paul Beverley - Version 22.09.22

' Highlights missing dialogue punctuation (from an idea by Katherine Kirk)

myColour = wdYellow

ActiveDocument.TrackRevisions = False

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^$"""

.Replacement.Highlight = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Replacement.ClearFormatting

.Text = "'"

.Replacement.Text = "'"

.Execute Replace:=wdReplaceAll

.Text = "([a-z]" & ChrW(8217) & "[!a-z])"

.Replacement.Highlight = True

.Replacement.Text = "\1"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub SentenceReverse()**

' Paul Beverley - Version 02.10.22

' Switches the order of the two sentences in the bullet point

myBullet = Chr(149)

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

s1 = Trim(Mid(rng.Sentences(1).Text, 3))

s2 = rng.Sentences(2).Text

newtext = Replace(myBullet & " " & s2 & " " & s1, vbCr, "")

rng.Text = newtext & vbCr

End Sub

**Sub FourDigitHundreds()**

' Paul Beverley - Version 04.10.22

' Changes four-figure numbers to words

Set rng = Selection.Range.Duplicate

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^#"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute

DoEvents

End With

rng.MoveEnd , 4

ch = Right(rng, 1)

If InStr("0123456789", ch) = 0 Then rng.MoveEnd , -1

myNum = rng

myNum = Replace(myNum, ",", "")

If Right(myNum, 2) <> "00" Then

rng.Select

Beep

Exit Sub

End If

rng.Text = Left(myNum, 2) & " hundred"

rng.Collapse wdCollapseStart

rng.Select

Application.Run macroName:="Normal.NewMacros.NumberToTextCMOS"

End Sub

**Sub PunctuationAfterCitation()**

' Paul Beverley - Version 17.10.22

' Moves the full point (period) to after the citation

Selection.Collapse wdCollapseStart

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Start = Selection.Start

paraEnd = rng.End

periodPos = InStr(rng, ".")

rng.MoveStart , periodPos - 1

rng.End = rng.Start + 1

rng.Cut

rng.End = paraEnd

closeParenPos = InStr(rng, ")")

rng.MoveStart , closeParenPos

rng.Collapse wdCollapseStart

rng.Select

Selection.Paste

End Sub

**Sub CommentAddAndPasteSelected()**

**'** Paul Beverley - Version 26.02.24

' Copies text, adds a comment and pastes text

introText = ""

' introText = "Suggested alternative: "

If Selection.Start = Selection.End Then Selection.Expand wdSentence

Selection.Copy

Dim cmt As Comment

Set cmt = Selection.Comments.Add(Range:=Selection.Range)

If introText > "" Then Selection.TypeText Text:=introText

Selection.Paste

ActiveWindow.ActivePane.Close

cmt.Edit

End Sub

**Sub MarkupModeSwitch()**

' Paul Beverley - Version 12.10.22

' Switches between inline and in-balloon markup

If ActiveWindow.View.MarkupMode = wdInLineRevisions Then

ActiveWindow.View.MarkupMode = wdBalloonRevisions

Else

ActiveWindow.View.MarkupMode = wdInLineRevisions

End If

End Sub

**Sub CambridgeDictionaryFetch()**

' Paul Beverley - Version 24.10.22

' Launches selected text on Cambridge Dictionary

mySite = "https://dictionary.cambridge.org/dictionary/english/"

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub NumberRangeHyphenToDash()**

**'** Paul Beverley - Version 14.11.24

' Changes all hyphens in number ranges to en dashes

notThese = "ISBN ISO BS EN doi www http"

wdsBefore = 8

thisArray = Split(" " & notThese & " ", " ")

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]-[0-9]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.Font.StrikeThrough = False

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

Set rng2 = rng.Duplicate

rng2.start = 0

If rng2.Words.Count < wdsBefore Then wdsBefore = rng2.Words.Count

rng2.MoveStart wdWord, -wdsBefore

doThisOne = True

doThisOne = True

For i = 1 To UBound(thisArray)

If Len(thisArray(i)) > 0 Then

If InStr(rng2.Text, thisArray(i)) > 0 Then doThisOne = False

End If

Next i

If doThisOne = True Then

rng.MoveEnd , -1

rng.MoveStart , 1

' The range is now just the hyphen

rng.Text = ChrW(8211)

rng.End = endNow + 1

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

End Sub

**Sub StrikeItalicParagraphs()**

' Paul Beverley - Version 04.11.22

' Add single strike to all italic paragraphs

maxWords = 20

For Each pa In ActiveDocument.Paragraphs

If pa.Range.Font.Italic = True And pa.Range.Words.count < maxWords Then \_

pa.Range.Font.StrikeThrough = True

DoEvents

Next pa

End Sub

**Sub QuoteSingleToDoubleGlobal()**

' Paul Beverley - Version 15.01.23

' Changes single quotes to double, avoiding apostrophes

myColour = wdBrightGreen

myFind = "',!t,!ve,!m,!s,!d,!r,!l,s!,'"

myFind = Replace(myFind, "!", ChrW(8217))

myReplace = myReplace & "',^&,^&,^&,^&,^&,^&,^&,^&,"""

fnd = Split(myFind, ",")

rpl = Split(myReplace, ",")

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

Set rngT = ActiveDocument.Content

If ActiveDocument.Footnotes.Count > 0 Then \_

Set rngF = ActiveDocument.StoryRanges(wdFootnotesStory)

If ActiveDocument.Endnotes.Count > 0 Then \_

Set rngE = ActiveDocument.StoryRanges(wdEndnotesStory)

For i = 0 To UBound(fnd)

Debug.Print fnd(i), rpl(i)

With rngT.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

If rpl(i) = """" Then .Font.StrikeThrough = False

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

If rpl(i) = "^&" Then .Replacement.Font.StrikeThrough = True

If Asc(fnd(i)) = Asc("s") Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

If ActiveDocument.Footnotes.Count > 0 Then

With rngF.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

If rpl(i) = """" Then .Font.StrikeThrough = False

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

If rpl(i) = "^&" Then .Replacement.Font.StrikeThrough = True

If Asc(fnd(i)) = Asc("s") Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

If ActiveDocument.Endnotes.Count > 0 Then

With rngE.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

If rpl(i) = """" Then .Font.StrikeThrough = False

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

If rpl(i) = "^&" Then .Replacement.Font.StrikeThrough = True

If Asc(fnd(i)) = Asc("s") Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

Next i

rngT.Font.StrikeThrough = False

If ActiveDocument.Footnotes.Count > 0 Then \_

rngF.Font.StrikeThrough = False

If ActiveDocument.Endnotes.Count > 0 Then \_

rngE.Font.StrikeThrough = False

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub HeadersFootersRemove()**

' Paul Beverley - Version 11.11.22

' Removes all headers and footers in all sections

For Each mySec In ActiveDocument.Sections

For Each myHead In mySec.Headers

If myHead.Exists Then myHead.Range.Text = ""

Next myHead

For Each myFoot In mySec.Footers

If myFoot.Exists Then myFoot.Range.Text = ""

Next myFoot

Next mySec

End Sub

**Sub HighlightComments()**

' Paul Beverley - Version 16.11.22

' Highlights all comments that have specific prefixes

wholeComment = True

myPrefixA = "AQ:"

myPrefixB = "TS:"

myColourA = wdBrightGreen

myColourB = wdYellow

For Each cmt In ActiveDocument.Comments

Set rng = cmt.Range

If InStr(rng, myPrefixA) = 1 Then

If wholeComment = False Then rng.End = rng.Start + Len(myPrefixA)

rng.HighlightColorIndex = myColourA

End If

If InStr(rng, myPrefixB) = 1 Then

If wholeComment = False Then rng.End = rng.Start + Len(myPrefixB)

rng.HighlightColorIndex = myColourB

End If

DoEvents

Next cmt

End Sub

**Sub OddQuoteSpacingCorrectAll()**

' Paul Beverley - Version 16.11.22

' Corrects the odd spacing on quotation marks (due to RTL fonts)

If ActiveDocument.Footnotes.Count > 0 Then

ActiveDocument.StoryRanges(wdFootnotesStory).Select

Selection.LtrRun

End If

If ActiveDocument.Endnotes.Count > 0 Then

ActiveDocument.StoryRanges(wdEndnotesStory).Select

Selection.LtrRun

End If

ActiveDocument.StoryRanges(wdMainTextStory).Select

Selection.LtrRun

Selection.Collapse wdCollapseStart

If ActiveDocument.Comments.Count > 0 Then

ActiveDocument.ActiveWindow.View.SplitSpecial = wdPaneComments

ActiveDocument.StoryRanges(wdCommentsStory).Select

Selection.LtrRun

ActiveDocument.ActiveWindow.View.SplitSpecial = wdPaneNone

End If

ActiveDocument.StoryRanges(wdMainTextStory).Select

Selection.Collapse wdCollapseStart

End Sub

**Sub HardSpaceAfterSuperscriptNumbers()**

' Paul Beverley - Version 17.11.22

' Switches space to hard space after all superscripted number

addHighlight = True

myColour = wdBrightGreen

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9][ ^0160]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

rng.MoveEnd , -1

If rng.Font.Superscript = True Then

myCount = myCount + 1

rng.MoveStart , 1

rng.MoveEnd , 1

rng.Text = ChrW(160)

rng.Font.Superscript = False

If addHighlight = True Then rng.HighlightColorIndex = myColour

If myCount Mod 20 = 0 Then rng.Select

rng.Collapse wdCollapseEnd

End If

rng.MoveStart , 2

rng.Find.Execute

DoEvents

Loop

MsgBox "Changed spaces: " & myCount

End Sub

**Sub AbbreviationAdd()**

' Paul Beverley - Version 15.06.22

' Creates an abbreviation of roughly selected text (in parenthesis)

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.MoveEnd , -1

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

myAbbr = ""

For Each wd In rng.Words

myAbbr = myAbbr & UCase(Left(wd, 1))

Next wd

rng.Select

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=" (" & myAbbr & ")"

End Sub

**Sub QQAddMenu()**

' Paul Beverley - Version 01.06.21

' Adds a QQ comment from a user-set menu

'addHighlight = wdYellow

addHighlight = wdNoHighlight

'addColour = wdColorBlue

addColour = wdColorBlack

addUline = True

addPNum = False

addLNum = False

qqBold = True

qqColour = wdBrightGreen

' qqColour = wdNoHighlight

qqSizeAdd = 2

' qqSizeAdd = 0

qqTitle = "QQcomments"

CR = vbCr

CR2 = CR & CR

' Find the QQcomments file

gottaFile = False

For Each myDoc In Documents

myText = myDoc.Content.Text

If InStr(Left(myText, 3), "QQ") > 0 Then

gottaFile = True

Exit For

End If

DoEvents

Next myDoc

If gottaFile = False Then

Set rng = myDoc.Content

Beep

rng.Start = rng.End - 4

If InStr(rng, "qq") > 0 Then

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

MsgBox "Are you sure that your QQcomments file says" \_

& CR2 & """QQcomments"" at the top?", , "QQAddMenu"

Exit Sub

Else

MsgBox "Can't find a comments file.", , "QQAddMenu"

rng.Text = rng.Text & CR & "qq"

Exit Sub

End If

End If

' Now find all the QQcomments in the menu

Dim cmts(100) As String

Dim cd(100) As String

i = 1

Do

myLine = myDoc.Paragraphs(i).Range.Text

i = i + 1

Loop Until Len(myLine) > 3 And InStr(myLine, "QQ") = 0

i = i - 1

For j = i To myDoc.Paragraphs.Count

Set myPar = myDoc.Paragraphs(j)

myLine = Replace(myPar.Range.Text, CR, "")

If Len(myLine) < 5 Then Exit For

cmts(j - i + 1) = myLine

Next j

cmTot = j - i

myFullPrompt = ""

myAllCodes = "|"

For cmNo = 1 To cmTot

myLine = cmts(cmNo)

sqPos = InStr(myLine, "[")

If sqPos > 0 Then

cmts(cmNo) = Left(myLine, sqPos - 2)

myPrompt = Replace(Mid(myLine, sqPos + 1), "]", "")

spPos = InStr(myPrompt, " ")

myCode = Left(myPrompt, spPos - 1)

' Debug.Print myPrompt

' Debug.Print cmts(cmNo) & "|" & CR

Else

myPrompt = myLine

spPos = InStr(myPrompt, " ")

myCode = Left(myPrompt, spPos - 1)

cmts(cmNo) = Mid(myPrompt, spPos + 1)

End If

myFullPrompt = myFullPrompt & myPrompt & CR

DoEvents

myAllCodes = myAllCodes & UCase(myCode) & "|" \_

& Trim(Str(cmNo)) & "|"

Next cmNo

' Debug.Print myAllCodes

' Choose the comment

codePos = 0

Do While codePos = 0

myText = InputBox(myFullPrompt, "MacroUpdater")

If myText = "" Then Beep: Exit Sub

myChosenCode = UCase(myText)

codePos = InStr(myAllCodes, "|" & myChosenCode & "|")

If codePos = 0 Then Beep

' Debug.Print Mid(myAllCodes, codePos + Len(myChosenCode) + 2)

cmntNum = Val(Mid(myAllCodes, codePos + Len(myChosenCode) + 2))

Loop

toType = cmts(cmntNum)

If toType = "Blank" Then toType = ""

If InStr(toType, "<>") > 0 Then Selection.Copy

If Selection.Information(wdInEndnote) Then

Beep

MsgBox "Place cursor in text.", , "QQadd"

Exit Sub

End If

' Record present cursor position

Set wasHere = Selection.Range.Duplicate

myFilename = ActiveDocument.Name

dotPos = InStr(myFilename, ".")

myFilename = Left(myFilename, dotPos - 1)

' If this is the first ever QQcomment, set things up

If ActiveDocument.Endnotes.Count = 0 Then

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=CR2

' Place bookmark just in front of "QQcomments"

ActiveDocument.Bookmarks.Add Name:="qqStart"

Selection.TypeText Text:=qqTitle & CR2 & CR2

Selection.MoveStart , -14

Selection.Range.Style = ActiveDocument.Styles(wdStyleNormal)

Selection.Font.Reset

' Set up endnotes style as numeric

With Selection.EndnoteOptions

.Location = wdEndOfDocument

.NumberingRule = wdRestartContinuous

.StartingNumber = 1

.NumberStyle = wdNoteNumberStyleArabic

End With

wasHere.Select

End If

' Is the cursor above the comments?

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.End = ActiveDocument.Content.End

If InStr(rng.Text, qqTitle) = 0 Then

Beep

MsgBox "Place cursor in main text," & CR & \_

"not in QQcomments area.", , "QQAddSimple"

Exit Sub

End If

If Selection.Start <> Selection.End Then

If doCopyText = True Then Selection.Copy

If addUline = True Then \_

Selection.Range.Font.Underline = True

If addHighlight > 0 Then \_

Selection.Range.HighlightColorIndex = addHighlight

If addColour > 0 Then \_

Selection.Range.Font.Color = addColour

End If

' Locate previous endnote

Set rng = Selection.Range.Duplicate

rng.Start = 0

enNum = rng.Endnotes.Count

If enNum > 0 Then

' Get the previous note text

prevNumText = ActiveDocument.Endnotes(enNum).Range.Text

Else

' The cursor must be above the first QQcomment

prevNumText = qqTitle

End If

' At the input point add an endnote

Selection.Endnotes.Add Range:=Selection.Range

' Find what is the max number used so far

newNum = 0

For i = 1 To ActiveDocument.Endnotes.Count

Etext = ActiveDocument.Endnotes(i).Range.Text

qqPos = InStr(Etext, "[qq")

thisNum = Val(Mid(Etext, qqPos + 3))

If thisNum > newNum Then newNum = thisNum

DoEvents

Next i

' Build up the new qq index number text

newNum = newNum + 1

numString = Right(Trim(Str(1000 + newNum)), 3)

qqString = "[qq" & numString & "]"

' Type it into the endnote

Selection.InsertAfter Text:=qqString

wasHere.Select

Selection.Collapse wdCollapseEnd

Selection.MoveEnd , 1

Selection.Range.HighlightColorIndex = qqColour

If qqBold Then Selection.Range.Bold = True

nowSize = Selection.Range.Font.Size

If qqSizeAdd > 0 Then

Selection.Range.Font.Size = nowSize + qqSizeAdd

End If

Selection.Collapse wdCollapseStart

' Switch windows or splits

' Are the multiple windows open on the file?

compMode = " - Compatibility Mode"

numBit = Replace(ActiveWindow.Caption, compMode, "")

numBit = Replace(numBit, myFilename, "")

' If so then switch to the alternate window

' i.e. the window showing the main text

If numBit > "" Then

winNum = Val(Right(numBit, 1))

If winNum = 1 Then

Windows(myFilename & " - 2").Activate

Else

Windows(myFilename & " - 1").Activate

End If

End If

If ActiveWindow.Panes.Count = 2 Then

If ActiveWindow.ActivePane.Index = 3 Then

ActiveWindow.Panes(1).Activate

Else

ActiveWindow.Panes(3).Activate

End If

End If

wasHere.Select

Selection.Collapse wdCollapseEnd

' In the QQcomments area, looking for previous comment

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = prevNumText

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

Selection.Collapse wdCollapseEnd

With Selection.Find

.Text = "[qq"

.Execute

End With

Selection.Collapse wdCollapseStart

If Selection.Find.Found = False Then

Selection.EndKey Unit:=wdStory

Selection.MoveLeft , 2

End If

pageNum = Selection.Information(wdActiveEndAdjustedPageNumber)

lineNum = Selection.Information(wdFirstCharacterLineNumber)

pNumText = ""

lNumText = ""

If addPNum = True Then

pNumText = " [p." & Trim(Str(pageNum)) & "] "

End If

If addLNum = True Then

lNumText = " [ln." & Trim(Str(lineNum)) & "] "

End If

qqText = Replace(qqString & " " & pNumText \_

& lNumText & CR2, " ", " ")

Selection.TypeText Text:=qqText

Selection.MoveLeft , 2

Selection.InsertAfter Text:=toType

Set rng = Selection.Range.Duplicate

copyPos = InStr(rng, "<>")

If copyPos > 0 Then

Selection.Start = rng.Start + copyPos - 1

Selection.End = rng.Start + copyPos + 1

Selection.Paste

End If

copyPos = InStr(rng, "<>")

If copyPos > 0 Then

Selection.Start = rng.Start + copyPos - 1

Selection.End = rng.Start + copyPos + 1

Selection.Paste

End If

rng.Select

Selection.Collapse wdCollapseEnd

cursorPos = InStr(rng, "|")

If cursorPos > 0 Then

Selection.Start = rng.Start + cursorPos - 1

Selection.End = rng.Start + cursorPos

Selection.Delete

End If

Selection.Collapse wdCollapseEnd

End Sub

**Sub NotesFootEndSwitch()**

' Paul Beverley - Version 22.11.22

' Switches individual footnote <-> endnote (in note or in text)

If Selection.StoryType = wdFootnotesStory Then

Selection.Footnotes.Convert

Exit Sub

End If

If Selection.StoryType = wdEndnotesStory Then

Selection.Endnotes.Convert

Exit Sub

End If

'If in main text, find the next foot/endnote marker

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^2"

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute

DoEvents

End With

If Selection.Find.Found = True Then

If Selection.Footnotes.Count > 0 Then

ActiveDocument.ActiveWindow.View.SeekView = wdSeekFootnotes

Selection.Footnotes.Convert

Exit Sub

End If

If Selection.Endnotes.Count > 0 Then

ActiveDocument.ActiveWindow.View.SeekView = wdSeekEndnotes

Selection.Endnotes.Convert

Exit Sub

End If

Else

Beep

End If

End Sub

**Sub** **HeadingStylesAutomatic()**

' Paul Beverley - Version 13.12.22

' Automatically adds styles to headings

maxWords = 10

convertAutoLists = False

removeSectionNumbers = True

ActiveDocument.Paragraphs(1).Range.Style = "Title"

' nmlSize = ActiveDocument.Styles(wdStyleNormal).Font.Size

nmlSize = 12

If convertAutoLists = True Then Call AutoListOff

myBullets = "\*-–—•"

bannedLastChars = ":;-/\_"

allHeads = "abstract, conclusions, discussion, introduction,"

allHeads = allHeads & "materials and methods, references, results,"

allHeads = allHeads & "results and discussion, results with discussion"

h1 = "Heading 1"

h2 = "Heading 2"

h3 = "Heading 3"

allHeads = Replace("," & allHeads & ",", " ", "")

For i = 2 To ActiveDocument.Paragraphs.Count

Set pa = ActiveDocument.Paragraphs(i).Range

Set rngText = pa.Duplicate

rngText.MoveEnd , -1

' rngText.Select

If rngText.Text > "" Then

Do While LCase(rngText.Characters(1)) = UCase(rngText.Characters(1))

rngText.MoveStart , 1

DoEvents

Loop

totWords = rngText.Words.Count

gotNumbers = False

myCap = rngText.Characters(1)

lastChar = rngText.Characters.Last

doThisOne = (myCap = UCase(myCap)) And \_

(LCase(rngText) <> UCase(rngText)) \_

And (InStr(bannedLastChars, lastChar) = 0)

' Debug.Print rngText, doThisOne

adsfgvsdf = 0

Else

doThisOne = False

End If

If Not (totWords > maxWords) And \_

pa.Information(wdWithInTable) = False And \_

doThisOne = True Then

fstChar = pa.Characters.First

fstWord = Trim(pa.Words.First)

wd2 = ""

wd3 = ""

wd4 = ""

wd5 = ""

If pa.Words.Count > 1 Then wd2 = pa.Words(2)

If pa.Words.Count > 2 Then wd3 = pa.Words(3)

If pa.Words.Count > 3 Then wd4 = pa.Words(4)

If pa.Words.Count > 4 Then wd5 = pa.Words(5)

' Try to exclude things that are not headings

' or are \*already\* Heading styles

' Debug.Print pa.Style, (InStr(pa.Style, "Heading") = 0), (InStr(LCase(pa.Style), "list") = 0), (InStr(myBullets, fstChar) = 0)

If InStr(pa.Style, "Heading") = 0 And \_

InStr(LCase(pa.Style), "list") = 0 And \_

InStr(myBullets, fstChar) = 0 Then

' Debug.Print fstWord, wd2, wd3, wd4, wd5

myText = LCase(Replace(Trim(pa), vbCr, ""))

myText = Replace(myText, " ", "")

If InStr(allHeads, myText) > 0 Then

pa.Style = h1

Else

' Add more definitions of heading 1

Set rng = pa.Duplicate

rng.MoveEnd , -1

Debug.Print rng.Text

If rng.Font.Size > nmlSize And rng.Font.Bold = True Then pa.Style = h1

If rng.Font.Size = nmlSize And rng.Font.Bold = True Then pa.Style = h2

If rng.Font.Size = nmlSize And rng.Font.Italic = True Then pa.Style = h3

If Val(fstWord) > 0 Then

gotNumbers = True

If wd2 = "." Then

If wd4 = "." Then

If Val(wd5) > 0 Then

pa.Style = h3

Else

pa.Style = h2

End If

Else

If Val(wd3) > 0 Then

pa.Style = h2

Else

pa.Style = h1

End If

End If

Else

pa.Style = h1

End If

End If

End If

End If

End If

If removeSectionNumbers = True And gotNumbers = True Then

rngText.Collapse wdCollapseStart

rngText.Start = pa.Start

If InStr(rngText, ".") Then rngText.Delete

End If

DoEvents

Next i

Beep

End Sub

**Sub** **FullNameAlyseCaseCount()**

' Paul Beverley - Version 23.12.22

' Counts each of a list of words/phrases

Set testDoc = ActiveDocument

' Find the word list

For Each myDoc In Documents

listText = myDoc.Content.Text

If Left(listText, 8) = "Fullname" Then

Set listDoc = myDoc

Exit For

End If

Next myDoc

' Find the start of the second listing

Set rng = listDoc.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Sorted by first name"

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute

DoEvents

End With

rng.Start = 0

startPara = rng.Paragraphs.Count + 1

myResults = ""

' From there to the end

For i = startPara To listDoc.Paragraphs.Count Step 3

Set rng = testDoc.Content

myText = listDoc.Paragraphs(i).Range.Text

If Len(myText) > 4 Then

myText = Left(myText, Len(myText) - 2)

myCount = Val(listDoc.Paragraphs(i + 1).Range.Text)

' Only if there's more than one,

' check if there's odd capitalisation

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

' Remember the first of a set

myFirst = rng.Text

useThisOne = False

myLine = ""

Do While rng.Find.Found = True

myLine = myLine & rng.Text & vbCr

Debug.Print rng.Text, myFirst

' If another one has different caps, note the fact

If rng.Text <> myFirst Then useThisOne = True

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

' If there are oddly capitalised ones,

' add them into the final output

If useThisOne = True Then myResults = myResults & myLine

End If

DoEvents

Next i

Documents.Add

Selection.TypeText Text:=myResults

Set rng = ActiveDocument.Content

rng.Sort

rng.InsertAfter Text:=vbCr

rng.Characters(1) = ""

numPars = ActiveDocument.Paragraphs.Count

myNum = 1

For j = numPars - 1 To 2 Step -1

Set rng1 = ActiveDocument.Paragraphs(j).Range

Set rng2 = ActiveDocument.Paragraphs(j - 1).Range

addCR = (LCase(rng1) <> LCase(rng2))

If rng1 = rng2 Then

rng1.Delete

myNum = myNum + 1

Else

rng1.MoveEnd , -1

rng1.InsertAfter Text:=vbTab & Trim(Str(myNum))

myNum = 1

End If

DoEvents

Debug.Print rng1, rng2 & vbCr

If addCR Then rng1.InsertBefore Text:=vbCr

Next j

Set rng = ActiveDocument.Paragraphs(1).Range

rng.MoveEnd , -1

rng.InsertAfter Text:=vbTab & Trim(Str(myNum))

Selection.WholeStory

Selection.ConvertToTable Separator:=wdSeparateByTabs

Set tb = ActiveDocument.Tables(1)

tb.AutoFitBehavior (wdAutoFitContent)

tb.Borders(wdBorderTop).LineStyle = wdLineStyleNone

tb.Borders(wdBorderLeft).LineStyle = wdLineStyleNone

tb.Borders(wdBorderBottom).LineStyle = wdLineStyleNone

tb.Borders(wdBorderRight).LineStyle = wdLineStyleNone

tb.Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

tb.Borders(wdBorderVertical).LineStyle = wdLineStyleNone

Selection.HomeKey Unit:=wdStory

End Sub

**Sub** **ScreenColourSwitch()**

' Paul Beverley - Version 01.01.23

' Switches colour text and background on/off

myBackColour = wdColorLightBlue

' myForeColour = wdColorYellow

myForeColour = RGB(255, 255, 240)

With ActiveDocument.Styles(wdStyleNormal).ParagraphFormat

myColourNow = .Shading.BackgroundPatternColor

If myColourNow = wdColorAutomatic Then

.Shading.BackgroundPatternColor = myBackColour

Else

.Shading.BackgroundPatternColor = wdColorAutomatic

End If

End With

With ActiveDocument.Styles(wdStyleNormal).Font

If myColourNow = wdColorAutomatic Then

.Color = myForeColour

myColNew = myBackColour

Else

myColNew = RGB(255, 255, 255)

.Color = wdAuto

End If

End With

With ActiveDocument.Background.Fill

.ForeColor.RGB = myColNew

.Visible = msoTrue

.Solid

End With

End Sub

**Sub NumberTextFigureCount()**

' Paul Beverley - Version 14.01.23

' Counts numbers as figures vs. words

noCount = "Jan,Feb,Mar,Apr,May,Jun,Jul,Aug,Sep,Oct,Nov,Dec,"

noCount = noCount & "hou,min,sec,day,"

maxFigs = 3

' 3 = 0-999

' 2 = 0-99

' 1 = 0-9

Select Case maxFigs

Case 1: mySearch = "<[0-9] "

Case 2: mySearch = "<[0-9]{1,2} "

Case 3: mySearch = "<[0-9]{1,3} "

End Select

noCount = "," & noCount

CR = vbCr

'Application.ScreenUpdating = False

'On Error GoTo ReportIt

'Use FindAndDo to check and count each 1-, 2- or 3-digit number

' that's NOT followed by Jan-Dec or units

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySearch

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

numFigs = 0

Do While rng.Find.Found = True

rng.Collapse wdCollapseEnd

rng.MoveEnd , 3

myTest = "," & rng.Text & ","

If InStr(noCount, myTest) = 0 Then numFigs = numFigs + 1

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

myRslt = "Numbers as digits" & vbTab & \_

Trim(Str(numFigs)) & CR

Set rng = ActiveDocument.Content

myTot = ActiveDocument.Range.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<ten "

.Wrap = wdFindContinue

.Replacement.Text = "^&x"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

aa = ActiveDocument.Range.End - myTot

If aa > 0 Then WordBasic.EditUndo

' spelt-out lower-case numbers over nine

DoEvents

' Find twenty, thirty etc

rng.Find.Text = "<[efnst][efghinorvwx]{2,4}ty "

rng.Find.Execute Replace:=wdReplaceAll

ab = ActiveDocument.Range.End - myTot

If ab > 0 Then WordBasic.EditUndo

DoEvents

rng.Find.Text = "<eleven "

rng.Find.Execute Replace:=wdReplaceAll

ac = ActiveDocument.Range.End - myTot

If ac > 0 Then WordBasic.EditUndo

DoEvents

rng.Find.Text = "<twelve "

rng.Find.Execute Replace:=wdReplaceAll

ad = ActiveDocument.Range.End - myTot

If ad > 0 Then WordBasic.EditUndo

rng.Find.Text = "<[efnst][efghinuorvwx]{2,4}teen "

rng.Find.Execute Replace:=wdReplaceAll

ae = ActiveDocument.Range.End - myTot

If ae > 0 Then WordBasic.EditUndo

DoEvents

rng.Find.Text = "<hundred>"

rng.Find.Execute Replace:=wdReplaceAll

af = ActiveDocument.Range.End - myTot

If af > 0 Then WordBasic.EditUndo

myRslt11Plus = "Spelt-out numbers (eleven etc.)" & vbTab & \_

Trim(Str(ab + ac + ad + ae + af)) & CR

myRslt10 = "Spelt-out numbers (ten)" & vbTab & \_

Trim(Str(aa)) & CR

myCount = 0

rng.Find.Text = "<one "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

DoEvents

rng.Find.Text = "<two "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

DoEvents

rng.Find.Text = "<three "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

rng.Find.Text = "<four "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

DoEvents

rng.Find.Text = "<five "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

rng.Find.Text = "<four "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

DoEvents

rng.Find.Text = "<four "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

rng.Find.Text = "<five "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

DoEvents

rng.Find.Text = "<six "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

rng.Find.Text = "<seven "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

DoEvents

rng.Find.Text = "<eight "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

rng.Find.Text = "<nine "

rng.Find.Execute Replace:=wdReplaceAll

x = ActiveDocument.Range.End - myTot

If x > 0 Then WordBasic.EditUndo

myCount = myCount + x

DoEvents

Selection.Collapse wdCollapseStart

myRslt = myRslt & "Spelt-out numbers (one-nine)" & vbTab & \_

Trim(Str(myCount)) & CR

myRslt = myRslt & myRslt10 & myRslt11Plus

Beep

' MsgBox myRslt

Documents.Add

Selection.TypeText Text:="NumberTextFigureCount" & CR & CR

Selection.Collapse wdCollapseEnd

startTable = Selection.End

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

Selection.TypeText Text:=myRslt

Selection.Start = startTable

Selection.End = ActiveDocument.Range.End

Selection.ConvertToTable Separator:=wdSeparateByTabs

Selection.Tables(1).AutoFitBehavior (wdAutoFitContent)

Selection.Tables(1).Borders(wdBorderTop).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderLeft).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderBottom).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderRight).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

Selection.Tables(1).Borders(wdBorderVertical).LineStyle = wdLineStyleNone

Selection.HomeKey Unit:=wdStory

Exit Sub

' Switch the screen back on if there's an error

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

End Sub

**Sub GoogleFetchCookie()**

' Paul Beverley - Version 25.01.23

' Launches selected text to Google website with cookie prompt

mySite = "http://www.google.co.uk/"

numTabs = 4

myWait = 2.5

myWait2 = 2.5

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

Selection.Copy

ActiveDocument.FollowHyperlink Address:=mySite

myTime = Timer

Do

Loop Until Timer > myTime + myWait

For i = 1 To numTabs

SendKeys "{TAB}"

Next i

myTime = Timer

Do

Loop Until Timer > myTime + myWait2

SendKeys "^v"

SendKeys "{ENTER}"

End Sub

**Sub ContentControlsRemove()**

' Paul Beverley - Version 14.04.21

' Moves the partly selected words one word right

' Code provided by Gary Smith

For Each myControl In ActiveDocument.ContentControls

myControl.Delete

Next myControl

End Sub

**Sub** **SentenceRoughRepeatChecker()**

' Paul Beverley - Version 14.02.23

' Unprotects the current file

firstWords = 3

finalWords = 3

myLink = "and"

' myLink = "or"

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.Text = rngOld.Text

For Each sn In ActiveDocument.Sentences

myText = sn.Text

Debug.Print myText

For Each snTest In ActiveDocument.Sentences

If sn.Start <> snTest.Start And \_

sn.Words.Count > firstWords + finalWords \_

And snTest.Words.Count > firstWords + finalWords Then

startSame = True

For i = 1 To firstWords

If sn.Words(i) <> snTest.Words(i) Then startSame = False

DoEvents

Next i

endSame = True

numWordsSN = sn.Words.Count

numWordsSNtest = snTest.Words.Count

For i = 1 To finalWords

If sn.Words(numWordsSN - i) <> snTest.Words(numWordsSNtest - i) \_

Then endSame = False

DoEvents

Next i

gotMatch = False

If LCase(myLink) = "and" Then

If endSame = True And startSame = True Then gotMatch = True

Else

If endSame = True Or startSame = True Then gotMatch = True

End If

If gotMatch = True Then

sn.Select

sn.HighlightColorIndex = wdYellow

snTest.HighlightColorIndex = wdBrightGreen

End If

End If

DoEvents

Next snTest

DoEvents

Next sn

Beep

End Sub

**Sub** **SentenceExactRepeatChecker()**

' Paul Beverley - Version 14.02.23

' Unprotects the current file

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.Text = rngOld.Text

For Each sn In ActiveDocument.Sentences

myText = sn.Text

Debug.Print myText

For Each snTest In ActiveDocument.Sentences

If sn.Start <> snTest.Start Then

sn.Select

If snTest.Text = myText Then

sn.HighlightColorIndex = wdYellow

snTest.HighlightColorIndex = wdBrightGreen

End If

End If

DoEvents

Next snTest

DoEvents

Next sn

Beep

End Sub

**Sub WordHippoFetch()**

' Paul Beverley - Version 03.03.23

' Launches selected text on Google

mySite = "https://www.wordhippo.com/what-is/another-word-for/"

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject & ".html"

Selection.Collapse wdCollapseEnd

End Sub

**Sub SuperscriptOnOff()**

' Paul Beverley - Version 05.03.23

' Toggles superscript ON/OFF

Selection.Font.Superscript = Not Selection.Font.Superscript

End Sub

**Sub SubscriptOnOff()**

' Paul Beverley - Version 05.03.23

' Toggles subscript ON/OFF

Selection.Font.Subscript = Not Selection.Font.Subscript

End Sub

**Sub BordersAddParagraphs()**

' Paul Beverley - Version 08.03.23

' Adds colour borders to (partly) selected paragraphs

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

myStart = rng.Start

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

rng.Start = myStart

Do

myText = InputBox("1: Copy this macro" & vbCr & \_

"2: Restore keystroke", "MacroUpdater")

myNumber = Val(myText)

If myNumber = 0 Then Beep: Exit Sub

DoEvents

Loop Until myNumber < 9

Select Case myNumber

Case 1:

With Options

.DefaultBorderLineStyle = wdLineStyleSingle

.DefaultBorderLineWidth = wdLineWidth150pt

.DefaultBorderColor = wdColorBrightGreen

End With

Case 2:

With Options

.DefaultBorderLineStyle = wdLineStyleSingle

.DefaultBorderLineWidth = wdLineWidth150pt

.DefaultBorderColor = wdColorRed

End With

Case 3:

With Options

.DefaultBorderLineStyle = wdLineStyleSingle

.DefaultBorderLineWidth = wdLineWidth150pt

.DefaultBorderColor = wdColorBlue

End With

Case 4:

With Options

.DefaultBorderLineStyle = wdLineStyleSingle

.DefaultBorderLineWidth = wdLineWidth300pt

.DefaultBorderColor = wdColorPink

End With

Case 5:

With Options

.DefaultBorderLineStyle = wdLineStyleSingleWavy

.DefaultBorderColor = wdColorBlack

End With

Case 6:

With Options

.DefaultBorderLineStyle = wdLineStyleDouble

.DefaultBorderLineWidth = wdLineWidth150pt

.DefaultBorderColor = wdColorTurquoise

End With

Case 7:

With Options

.DefaultBorderLineStyle = wdLineStyleDot

.DefaultBorderLineWidth = wdLineWidth225pt

.DefaultBorderColor = wdColorRed

End With

Case 8:

With Options

.DefaultBorderLineStyle = wdLineStyleEmboss3D

.DefaultBorderColor = wdColorTurquoise

End With

Case Else

With Options

.DefaultBorderLineStyle = wdLineStyleSingle

.DefaultBorderLineWidth = wdLineWidth150pt

.DefaultBorderColor = wdColorBlack

End With

End Select

With rng.Font.Borders(1)

.LineStyle = Options.DefaultBorderLineStyle

.LineWidth = Options.DefaultBorderLineWidth

.Color = Options.DefaultBorderColor

End With

End Sub

**Sub WildcardSave()**

**'** Paul Beverley - Version 31.03.23

' Reads the current 'Selection.Find' and saves at the top of the file

addBookmark = True

Set rng = Selection.Range.Duplicate

If addBookmark = True Then \_

ActiveDocument.Bookmarks.Add Name:="myTempMark"

Selection.HomeKey Unit:=wdStory

myFind = Selection.Find.Text

Selection.TypeText Text:=myFind & vbCr

For i = 1 To 100

DoEvents

Next i

myResponse = MsgBox("Find saved", vbOKOnly, "WildcardSave")

rng.Select

Selection.MoveLeft , 1

Selection.MoveRight , 1

End Sub

**Sub WildcardType()**

**'** Paul Beverley - Version 31.03.23

' Reads the current 'Selection.Find' and types it in the file

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Selection.TypeText Text:=Selection.Find.Text & vbCr

End Sub

**Sub FontNameSizeScanAndFix()**

' Paul Beverley - Version 13.03.23

' Assesses and lists (& optionally fixes) the font sizes & names

Dim descrip(100) As String

Dim count(100) As Integer

If Selection.Start = Selection.End Then

Beep

myResponse = MsgBox("Please select some text!", vbOKOnly, \_

"FontNameSizeScanAndFix")

Exit Sub

End If

numDescrips = 0

For Each wd In Selection.Range.Words

If wd.Font.Name > "" And wd.Font.Size < 100 Then

myDescript = "$" & wd.Font.Name & "!" & Trim(Str(wd.Font.Size)) & "="

If numDescrips = 0 Then

numDescrips = 1

descrip(1) = myDescript

count(1) = 1

allDescript = myDescript

Else

myStart = InStr(allDescript, myDescript)

If myStart = 0 Then

allDescript = allDescript & myDescript

numDescrips = numDescrips + 1

descrip(numDescrips) = myDescript

count(numDescrips) = 1

Else

myText = Left(allDescript, myStart + 1)

ptr = Len(myText) - Len(Replace(myText, "$", ""))

count(ptr) = count(ptr) + 1

End If

End If

End If

DoEvents

Next wd

myScores = ""

topScore = 0

For i = 1 To numDescrips

myScores = myScores & descrip(i) & Trim(Str(count(i))) & vbCr

If count(i) > topScore Then

topScore = count(i)

topFontNum = i

End If

Next i

myScores = Replace(myScores, "$", "")

myScores = Replace(myScores, "!", " ")

myScores = Replace(myScores, "=", " .....")

myRecommend = descrip(topFontNum)

myRecommend = Replace(myRecommend, "$", "")

myRecommend = Replace(myRecommend, "!", " ")

myRecommend = Replace(myRecommend, "=", "")

myPrompt = myScores & vbCr & vbCr & "Change to:" & vbCr \_

& vbCr & myRecommend

myResponse = MsgBox(myPrompt, vbQuestion + vbOKCancel, \_

"FontNameSizeScanAndFix")

If myResponse <> vbOK Then Beep: Exit Sub

myEnd = InStr(descrip(topFontNum), "!")

myNewFont = Mid(descrip(topFontNum), 2, myEnd - 2)

myNewSize = Val(Mid(descrip(topFontNum), myEnd + 1))

Selection.Range.Font.Name = myNewFont

Selection.Range.Font.Size = myNewSize

End Sub

**Sub NoteMarkerMoveEndSentence()**

' Paul Beverley - Version 13.03.23

' Moves the foot/endnote marker to the end of the sentence

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Do

Selection.MoveLeft , 1

DoEvents

Loop Until Asc(Selection) = 2

hereNow = Selection.Start

Selection.Expand wdSentence

Selection.Start = hereNow + 1

Selection.MoveEnd , -1

Selection.Cut

Selection.MoveLeft , 1

Selection.Paste

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub FontNameSizeScanAndFix()**

' Paul Beverley - Version 13.03.23

' Assesses and lists (& optionally fixes) the font sizes & names

Dim descrip(100) As String

Dim count(100) As Integer

If Selection.Start = Selection.End Then

Beep

myResponse = MsgBox("Please select some text!", vbOKOnly, \_

"FontNameSizeScanAndFix")

Exit Sub

End If

numDescrips = 0

For Each wd In Selection.Range.Words

If wd.Font.Name > "" And wd.Font.Size < 100 Then

myDescript = "$" & wd.Font.Name & "!" & Trim(Str(wd.Font.Size)) & "="

If numDescrips = 0 Then

numDescrips = 1

descrip(1) = myDescript

count(1) = 1

allDescript = myDescript

Else

myStart = InStr(allDescript, myDescript)

If myStart = 0 Then

allDescript = allDescript & myDescript

numDescrips = numDescrips + 1

descrip(numDescrips) = myDescript

count(numDescrips) = 1

Else

myText = Left(allDescript, myStart + 1)

ptr = Len(myText) - Len(Replace(myText, "$", ""))

count(ptr) = count(ptr) + 1

End If

End If

End If

DoEvents

Next wd

myScores = ""

topScore = 0

For i = 1 To numDescrips

myScores = myScores & descrip(i) & Trim(Str(count(i))) & vbCr

If count(i) > topScore Then

topScore = count(i)

topFontNum = i

End If

Next i

myScores = Replace(myScores, "$", "")

myScores = Replace(myScores, "!", " ")

myScores = Replace(myScores, "=", " .....")

myRecommend = descrip(topFontNum)

myRecommend = Replace(myRecommend, "$", "")

myRecommend = Replace(myRecommend, "!", " ")

myRecommend = Replace(myRecommend, "=", "")

myPrompt = myScores & vbCr & vbCr & "Change to:" & vbCr \_

& vbCr & myRecommend

myResponse = MsgBox(myPrompt, vbQuestion + vbOKCancel, \_

"FontNameSizeScanAndFix")

If myResponse <> vbOK Then Beep: Exit Sub

myEnd = InStr(descrip(topFontNum), "!")

myNewFont = Mid(descrip(topFontNum), 2, myEnd - 2)

myNewSize = Val(Mid(descrip(topFontNum), myEnd + 1))

Selection.Range.Font.Name = myNewFont

Selection.Range.Font.Size = myNewSize

End Sub

**Sub MarkIt()**

**'** Paul Beverley - Version 25.10.23

' Applies various attributes by (wildcard) F&R

confirmMarkItList = True

' confirmMarkItList = False

On Error GoTo ReportIt

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set workDoc = ActiveDocument

myResponse = vbNo

For Each listDoc In Application.Documents

DoEvents

pNum = listDoc.Paragraphs.Count

myNum = 3

If pNum < 3 Then myNum = pNum

Set rng = listDoc.Paragraphs(myNum).Range

rng.Start = 0

If InStr(LCase(rng.Text), "markit") Then

listDoc.Activate

listDoc.Windows(1).WindowState = wdWindowStateNormal

If confirmMarkItList = True Then

myResponse = MsgBox("Is this your MarkIt list?", \_

vbQuestion + vbYesNo, "MarkIt")

Else

myResponse = vbYes

End If

If myResponse = vbYes Then Exit For

End If

Next listDoc

If myResponse <> vbYes Then

Beep

myResponse = MsgBox("Can't find a MarkIt list." & vbCr & vbCr & \_

"Please ensure that your MarkIt list is open and starts with:" \_

& vbCr & vbCr & "| MarkIt", vbExclamation + vbOKOnly, "MarkIt")

Exit Sub

End If

If listDoc = workDoc Then

Beep

myResponse = MsgBox("Please place the cursor in the text to be marked." \_

& vbCr & vbCr & "and rerun MarkIt.", vbExclamation + vbOKOnly, \_

"MarkIt")

Exit Sub

End If

' Now do the F&Rs

workDoc.Activate

Set rng = Selection.Range.Duplicate

Selection.HomeKey Unit:=wdStory

Set main = ActiveDocument.Content

doFoots = ActiveDocument.Footnotes.Count > 0

doEnds = ActiveDocument.Endnotes.Count > 0

If doFoots Then Set foots = workDoc.StoryRanges(wdFootnotesStory)

If doEnds Then Set ends = workDoc.StoryRanges(wdEndnotesStory)

oldColour = Options.DefaultHighlightColorIndex

numLines = listDoc.Paragraphs.Count

ReDim myText(numLines) As String

ReDim isItalic(numLines) As Boolean

ReDim isBold(numLines) As Boolean

ReDim isUline(numLines) As Boolean

ReDim isWild(numLines) As Boolean

ReDim doMatch(numLines) As Boolean

ReDim hiColour(numLines) As Long

ReDim textColour(numLines) As Long

For i = 1 To 40

mySPs = mySPs & " "

Next i

mySPs = mySPs & "To go: "

' First fill arrays for all the F&Rs needed

i = 0

For Each ma In listDoc.Paragraphs

' Only do anything if there's text on the line

doThis = (Len(ma) > 2)

Debug.Print ma.Range.Text

If doThis = True And Left(ma.Range.Text, 1) <> ChrW(124) \_

And ma.Range.Characters(1).Font.StrikeThrough = False Then

i = i + 1

myText(i) = Replace(ma.Range.Text, vbCr, "")

' Check for any wildcard characters

isWild(i) = False

If InStr(myText(i), "[") > 0 Then isWild(i) = True

If InStr(myText(i), "<") > 0 Then isWild(i) = True

If InStr(myText(i), ">") > 0 Then isWild(i) = True

' Check for bent pipe = any case

If Left(myText(i), 1) = ChrW(172) Then

myText(i) = Mid(myText(i), 2)

doMatch(i) = False

Else

doMatch(i) = True

End If

isItalic(i) = ma.Range.Font.Italic

isBold(i) = ma.Range.Font.Bold

isUline(i) = ma.Range.Font.Underline

hiColour(i) = ma.Range.HighlightColorIndex

textColour(i) = ma.Range.Font.Color

' First three are neg if true;

' second two are positive if true.

' So if you're asking for none of these, ignore this line

If (isItalic(i) + isBold(i) + isUline(i) + hiColour(i) = 0) And \_

(textColour(i) < 1) Then i = i - 1

End If

Next ma

numFRs = i

For i = 1 To numFRs

Options.DefaultHighlightColorIndex = hiColour(i)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText(i)

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

If isItalic(i) Then .Replacement.Font.Italic = True

If isBold(i) Then .Replacement.Font.Bold = True

If isUline(i) Then .Replacement.Font.Underline = True

If hiColour(i) > 0 Then .Replacement.Highlight = True

If textColour(i) > 0 Then .Replacement.Font.Color = textColour(i)

.MatchCase = doMatch(i)

.MatchWildcards = isWild(i)

.Execute Replace:=wdReplaceAll

DoEvents

End With

If doFoots Then

With foots.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText(i)

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

If isItalic(i) Then .Replacement.Font.Italic = True

If isBold(i) Then .Replacement.Font.Bold = True

If isUline(i) Then .Replacement.Font.Underline = True

If hiColour(i) > 0 Then .Replacement.Highlight = True

If textColour(i) > 0 Then .Replacement.Font.Color = textColour(i)

.MatchCase = doMatch(i)

.MatchWildcards = isWild(i)

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

If doEnds Then

With ends.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

If isItalic(i) Then .Replacement.Font.Italic = True

If isBold(i) Then .Replacement.Font.Bold = True

If isUline(i) Then .Replacement.Font.Underline = True

If hiColour(i) > 0 Then .Replacement.Highlight = True

If textColour(i) > 0 Then .Replacement.Font.Color = textColour

.MatchCase = doMatch(i)

.MatchWildcards = isWild(i)

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

StatusBar = mySPs & Str(numFRs - i)

Debug.Print mySPs & Str(numFRs - i)

Next i

Options.DefaultHighlightColorIndex = oldColour

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchCase = False

.MatchWildcards = False

DoEvents

End With

ActiveDocument.TrackRevisions = myTrack

StatusBar = ""

rng.Select

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

ReportIt:

Application.ScreenUpdating = True

ern = Err.Number

If ern = 9118 Or ern = 5590 Or ern = 5560 Or ern = 5692 Then

ActiveDocument.ActiveWindow.LargeScroll down:=1

ma.Range.Select

ActiveDocument.ActiveWindow.SmallScroll down:=1

Beep

MsgBox "Wildcard error"

Err.Clear

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub MacroNameShow(macroName As String)**

**'** Paul Beverley - Version 10.05.24

' Displays the current macro name in a display file

myFile = "Macro\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_display.docx"

bigFont = 14

smallFont = 9

bigColor = wdColorBlack

smallColour = wdColorGray50

For Each myDoc In Documents

myName = myDoc.Name

If myName = myFile Then Exit For

Next myDoc

If myName <> myFile Then Exit Sub

myScreen = Application.ScreenUpdating

Application.ScreenUpdating = True

Set myDoc = ActiveDocument

Documents(myFile).Activate

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.Font.Size = smallFont

rng.Font.Color = smallColour

macroName = Replace(macroName, " (", vbCr & "(")

Selection.InsertBefore Text:=macroName & vbCr & vbCr

Selection.Collapse wdCollapseStart

rng.Paragraphs(1).Range.Font.Color = bigColor

rng.Paragraphs(1).Range.Font.Size = bigFont

If rng.Paragraphs.Count > 1 Then \_

rng.Paragraphs(2).Range.Font.Color = bigColor

Documents(myDoc).Activate

Application.ScreenUpdating = myScreen

End Sub

**Sub HyphenSpaceWordCount()**

**'** Paul Beverley - Version 30.03.23

' Counts hyphenated word forms

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

tx = Selection.Text

For i = 2 To Len(tx) - 2

wd1 = Left(tx, i)

wd2 = Mid(tx, i + 1)

myPair = tx

If Application.CheckSpelling(wd1) And \_

Application.CheckSpelling(wd2) Then

myPair = wd1 & " " & wd2

Exit For

End If

Next i

myPair = InputBox("OK? (Add a space if necessary)", "HyphenSpaceWordCount", myPair)

If InStr(myPair, " ") = 0 Then Beep: Exit Sub

Else

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

rng.Select

myPair = Selection

End If

brkPos = InStr(myPair, " ")

If brkPos = 0 Then brkPos = InStr(myPair, "-")

If brkPos = 0 Then brkPos = InStr(myPair, ChrW(8211))

If brkPos = 0 Then brkPos = InStr(myPair, "/")

If brkPos = 0 Then

Beep

myResponse = MsgBox("Invalid input", vbOKOnly, \_

"HyphenSpaceWordCount")

Exit Sub

End If

wd1 = LCase(Left(myPair, brkPos - 1))

wd2 = LCase(Mid(myPair, brkPos + 1))

Set rng = ActiveDocument.Content

allText = LCase(rng.Text)

myTot = Len(allText)

p = wd1 & " " & wd2

spaceCount = Len(Replace(allText, p, p & "!")) - myTot

p = wd1 & "-" & wd2

hyphenCount = Len(Replace(allText, p, p & "!")) - myTot

p = wd1 & ChrW(8211) & wd2

dashCount = Len(Replace(allText, p, p & "!")) - myTot

p = wd1 & "/" & wd2

slashCount = Len(Replace(allText, p, p & "!")) - myTot

p = wd1 & wd2

oneWordCount = Len(Replace(allText, p, p & "!")) - myTot

myResult = wd1 & wd2 & ": " & Str(oneWordCount) & vbCr

myResult = myResult & wd1 & " " & wd2 & ": " & \_

Str(spaceCount) & vbCr

myResult = myResult & wd1 & "-" & wd2 & ": " & Str(hyphenCount) \_

& vbCr & vbCr

myResult = myResult & wd1 & "<dash>" & wd2 & ": " & \_

Str(dashCount) & vbCr

myResult = myResult & wd1 & "/" & wd2 & ": " & Str(slashCount)

' Load WC find box

mySch = Left(wd1, Len(wd1) - 1) & "[" & Right(wd1, 1) & \_

"^32/\-" & ChrW(8211) & Left(wd2, 1) & "]{2,3}" & Mid(wd2, 2)

Debug.Print mySch

Selection.Collapse wdCollapseStart

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySch

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

End With

MsgBox myResult, 0, "HyphenSpaceWordCount"

End Sub

**Sub CommentAddToTrackedChanges()**

**'** Paul Beverley - Version 06.04.23

' Adds a blank comment to every tracked change in the selected text

If Selection.Start = Selection.End Then

Beep

MsgBox "Please select an area of text."

Exit Sub

End If

Set rng = Selection.Range.Duplicate

i = 0

For Each rev In rng.Revisions

myType = rev.Type

Set rng2 = rev.Range.Duplicate

If myType = 2 Then

rng2.MoveStart , -1

rng2.MoveEnd , 1

End If

Set cmt = rng2.Comments.Add(Range:=rng2)

DoEvents

i = i + 1

Next rev

Beep

MsgBox Str(i) & " comments added"

End Sub

**Sub MultiFileLoader()**

**'** Paul Beverley - Version 06.04.23

' Loads a set of files

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

' First find if this is a Mac or a PC!

myFullName = ActiveDocument.FullName

myName = ActiveDocument.Name

myFolder = Replace(myFullName, myName, "")

myDelimiter = Application.PathSeparator

If myDelimiter = "/" Then

InAMac = True

Else

InAMac = False

End If

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 \_

And InStr(LCase(rng.Text), ".rtf") = 0 \_

And InStr(LCase(rng.Text), ".pdf") = 0 Then

' If not a file list then navigate to the required folder

If remindAboutCancel = True Then myResponse = \_

MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "MultiFileLoader")

docCount = Documents.count

Dialogs(wdDialogFileOpen).Show

If Documents.count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & myDelimiter)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or \_

InStr(LCase(myFile), ".rtf") > 0 Or \_

InStr(LCase(myFile), ".pdf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=myDelimiter, count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Collect unformatted text from ALL the files in" & \_

" directory: " & dirName & "?", vbQuestion + vbYesNoCancel, \_

"Multifile Text Collection")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=myDelimiter, count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "=" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

For i = 1 To numFiles

' Get the folder name, and open the file

StatusBar = allMyFiles(i)

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set myDoc = Application.Documents.Open(fileName:=thisFile)

Next i

End Sub

**Sub HighlightOffNext()**

**'** Paul Beverley - Version 15.04.23

' Unhighlights the next area of highlighting

doTrack = False

myTrack = ActiveDocument.TrackRevisions

If doTrack = False Then ActiveDocument.TrackRevisions = False

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindStop

.Forward = True

.Highlight = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute

DoEvents

End With

If Selection.Find.Found = True Then

Selection.Range.HighlightColorIndex = wdNoHighlight

Selection.Collapse wdCollapseEnd

Else

Beep

End If

If doTrack = False Then ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub HighlightOffNextWord()**

**'** Paul Beverley - Version 15.04.23

' Unhighlights the next word that has some highlighting

doTrack = False

myTrack = ActiveDocument.TrackRevisions

If doTrack = False Then ActiveDocument.TrackRevisions = False

Do

Selection.Collapse wdCollapseEnd

Selection.MoveRight wdWord, 1, Extend:=wdExtend

myCol = Selection.Range.HighlightColorIndex

DoEvents

Loop Until myCol <> wdNoHighlight Or Selection = vbCr

Selection.Range.HighlightColorIndex = wdNoHighlight

If Selection = vbCr Then Beep

Selection.Collapse wdCollapseEnd

If doTrack = False Then ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub OddLanguageHighlighter()**

**'** Paul Beverley - Version 12.04.23

' Highlights any characters NOT in the chosen language

myLanguage = Selection.LanguageID

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

For h = 1 To rng.Paragraphs.count

If rng.Paragraphs(h).Range.LanguageID <> myLanguage Then

For i = 1 To rng.Paragraphs(h).Range.Words.count

If rng.Paragraphs(h).Range.Words(i).LanguageID <> myLanguage Then

For j = 1 To Len(rng.Paragraphs(h).Range.Words(i))

If rng.Paragraphs(h).Range.Words(i).Characters(j).LanguageID \_

<> myLanguage Then

rng.Paragraphs(h).Range.Words(i).Characters(j).HighlightColorIndex \_

= wdYellow

End If

DoEvents

Next j

End If

DoEvents

Next i

End If

If h Mod 50 = 0 Then rng.Paragraphs(h).Range.Select

DoEvents

Next h

End Sub

**Sub MultiFileDocToDocxSave()**

**'** Paul Beverley - Version 12.05.23

' Save any .doc files in a folder also in .docx format

newFilesInFolder = False

' If you make the above 'True', then you must first create

' a folder of this name:

newFolderName = "WordFormat"

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.count

Dialogs(wdDialogFileOpen).Show

If Documents.count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If Right(myFile, 4) = ".doc" Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

If pdfFilesInFolder = True Then

mySubFolder = "\" & pdfFolderName

Else

mySubFolder = ""

End If

myResponse = MsgBox("Save ALL the .doc files in this folder: " & mySubFolder & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "Multifile DocToDocx")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

If newFilesInFolder = True Then

myResponse = MsgBox("Save just the files listed here, in this folder: " & \_

mySubFolder & myFolder & " ?", \_

vbQuestion + vbYesNoCancel, "Multifile DocToDocx")

If myResponse <> vbYes Then Exit Sub

End If

For i = 1 To numFiles

thisName = allMyFiles(i)

myFileType = Mid(thisName, InStr(thisName, "."))

justName = Replace(thisName, myFileType, "")

thisFile = myFolder & myDelimiter & thisName

Set myDoc = Application.Documents.Open(fileName:=thisFile)

StatusBar = allMyFiles(i)

If newFilesInFolder = True Then

justName = newFolderName & "\" & justName

End If

DoEvents

'ActiveDocument.ExportAsFixedFormat OutputFileName:=justName, \_

ExportFormat:=wdFormatXMLDocument, OpenAfterExport:=False

ActiveDocument.SaveAs2 fileName:=justName, FileFormat:=wdFormatXMLDocument \_

, LockComments:=False

myDoc.Close SaveChanges:=wdDoNotSaveChanges

Next

Selection.HomeKey Unit:=wdStory

' Dummy copy to clear clipboard

Set rng = ActiveDocument.Content

rng.End = rng.Start + 1

rng.Copy

MsgBox (numFiles & " files saved as .docx")

End Sub

**Sub DocAlyseUser()**

**'** Paul Beverley - Version 18.05.23

' Create your own (multilingual?) document analyses

' confirmCountItList = True

confirmCountItList = False

myScreenOff = True

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox(" DocAlyseUser" & vbCr & vbCr & \_

"Analyse this document?", vbQuestion \_

+ vbYesNoCancel, "DocAlyseUser")

If myResponse <> vbYes Then Exit Sub

End If

If myScreenOff = True Then Application.ScreenUpdating = False

On Error GoTo ReportIt

CR = vbCr

CR2 = CR & CR

CR3 = CR2 & CR

Set workDoc = ActiveDocument

myResponse = vbNo

For Each listDoc In Application.Documents

DoEvents

pNum = listDoc.Paragraphs.count

myNum = 3

If pNum < 3 Then myNum = pNum

Set rng = listDoc.Paragraphs(myNum).Range

rng.Start = 0

If InStr(LCase(rng.Text), "countit") Then

listDoc.Activate

listDoc.Windows(1).WindowState = wdWindowStateNormal

If confirmCountItList = True Then

myResponse = MsgBox("Is this your CountIt list?", \_

vbQuestion + vbYesNo, "DocAlyseUser")

Else

myResponse = vbYes

End If

If myResponse = vbYes Then Exit For

End If

Next listDoc

If myResponse <> vbYes Then

Beep

myResponse = MsgBox("Can't find a CountIt list." & CR2 & \_

"Please ensure that your CountIt list is open and starts with:" \_

& CR2 & "| CountIt", vbExclamation + vbOKOnly, "DocAlyseUser")

Exit Sub

End If

If listDoc = workDoc Then

Beep

myResponse = MsgBox("Please place the cursor in the text to be tested." \_

& CR2 & "and rerun DocAlyseUser.", vbExclamation + vbOKOnly, \_

"DocAlyseUser")

Exit Sub

End If

' Copy all text to output file

addText = ""

If workDoc.Footnotes.count > 0 Then addText = addText & \_

workDoc.StoryRanges(wdFootnotesStory).Text & CR

If workDoc.Endnotes.count > 0 Then addText = addText & \_

workDoc.StoryRanges(wdEndnotesStory).Text & CR

Documents.Add

Set output = ActiveDocument.Content

output.Text = workDoc.Content.Text & addText

myResults = "DocAlyseUser" & CR

For Each ma In listDoc.Paragraphs

' Only do anything if there's text on the line

padPosition = InStr(ma, "|")

If InStr(ma, "||") Then padPosition = 2

doCount = False

myText = Replace(ma.Range.Text, CR, "")

Select Case padPosition

Case 0: myResults = myResults & ma.Range.Text

Case 1: If InStr(LCase(myText), "countit") = 0 Then \_

myResults = myResults & ma.Range.Text

Case 2: ' Ignore

Case Else

doCount = True

' Check for any wildcard characters

isWild = False

lcaseWhole = False

If InStr(myText, "[") > 0 Then isWild = True

If InStr(myText, "<") > 0 Then isWild = True

If InStr(myText, ">") > 0 Then isWild = True

' Check for bent pipe = any case

If InStr(myText, ChrW(172)) > 0 Then \_

doMatch = False

If InStr(myText, ChrW(172) & "<") > 0 Then

lcaseWhole = True

Else

doMatch = True

End If

End Select

If doCount = True Then

' Process the item

itemDescript = Left(myText, padPosition - 1)

myFind = Mid(myText, padPosition + 1)

myFind = Replace(myFind, ChrW(172), "")

If lcaseWhole = True Then

myFind = Replace(myFind, "<", "")

myFind = Replace(myFind, ">", "")

newFind = "<"

For i = 1 To Len(myFind)

ch = Mid(myFind, i, 1)

newFind = newFind & "[" & ch & UCase(ch) & "]"

DoEvents

Next i

myFind = newFind & ">"

isWild = True

End If

wasTot = Len(output.Text)

With output.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindContinue

.Replacement.Text = "^&!"

.MatchCase = doMatch

.MatchWildcards = isWild

.Execute Replace:=wdReplaceAll

DoEvents

End With

DoEvents

numFinds = Len(output.Text) - wasTot

If numFinds > 0 Then WordBasic.EditUndo

' Now add result to output stream

myResults = myResults & itemDescript & vbTab & \_

Trim(Str(numFinds)) & CR

End If

Next ma

myResults = Replace(myResults, CR3, CR2)

output.Text = myResults

Selection.HomeKey Unit:=wdStory

output.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading1)

'output.Paragraphs(3).Range.Select

'Selection.End = output.End

Set rng = ActiveDocument.Content

rng.ParagraphFormat.TabStops.ClearAll

rng.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(4.5), \_

Alignment:=wdAlignTabRight, Leader:=wdTabLeaderSpaces

' Grey out the zero lines

DoEvents

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13([!^13]@)^t0"

.Wrap = wdFindContinue

.Replacement.Text = "^p\1^t^="

.Replacement.Font.Color = wdColorGray25

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

With Selection.Find

.Text = ""

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

Selection.HomeKey Unit:=wdStory

If doingSeveralMacros = False Then

Beep

Else

FUT.Activate

End If

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

errNum = Err.Number

If errNum = 9118 Or errNum = 5590 Or errNum = 5560 Or errNum = 5692 Then

ActiveDocument.ActiveWindow.LargeScroll down:=1

ma.Range.Select

ActiveDocument.ActiveWindow.SmallScroll down:=1

Beep

MsgBox "Wildcard error"

Err.Clear

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub LinksAllToEndnotes()**

**'** Paul Beverley - Version 18.05.23

' Change each active URL link into an endnote

stopAfterEachOne = True

For i = ActiveDocument.Fields.count To 1 Step -1

myAddress = ActiveDocument.Fields(i).Code

If ActiveDocument.Fields(i).Type = 88 Then

myAddress = Replace(myAddress, "HYPERLINK """, "")

myAddress = Replace(myAddress, " \t ""\_blank""", "")

myAddress = Replace(myAddress, """", "")

Debug.Print myAddress

ActiveDocument.Fields(i).Select

Selection.Collapse wdCollapseEnd

If InStr(".,;:!?", Selection) > 0 Then Selection.MoveRight , 1

ActiveDocument.Fields(i).Unlink

Selection.Endnotes.Add Range:=Selection.Range

Selection.TypeText Text:=myAddress & "."

Selection.Expand wdParagraph

Selection.MoveEnd , -2

Selection.MoveStart , 2

ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, Address:=myAddress

End If

If stopAfterEachOne = True Then

myResponse = MsgBox("Continue?", \_

vbQuestion + vbYesNo, "LinksAllToEndnotes")

If myResponse <> vbYes Then Exit Sub

End If

Next i

End Sub

**Sub MacroLauncherForMac()**

**'** Paul Beverley - Version 04.07.20

' Offers a list of macros to launch (was called "MacroMenuForMac")

myList1 = "a=AccentAlyse, d=DcAlyse, h=HyphenAlyse"

myList2 = "w=WordPairAlyse, p=ProperNounAlyse, i=IZISCount"

myList3 = "z=IStoIZ, s=IZtoIS, l=SpellingErrorLister"

myList4 = "c=CopyTextSimple, m=MultiFileText, f=FRedit"

myList5 = "e=SpellingErrorHighlighter, u=UKUSCount"

' myList5 = "e=SpellingErrorHighlighter, u=UKUSCount, g=MegAlyse"

myList = "," & myList1 & "," & myList2 & "," & myList3 & "," \_

& myList4 & "," & myList5

myList = Replace(myList, ",,", ",")

mcr = Split(myList, ",")

myPrompt = ""

myCodes = ""

numItems = UBound(mcr)

For i = 1 To numItems

mcr(i) = Trim(mcr(i))

myCode = Left(mcr(i), 1)

myCodes = myCodes & myCode

mName = Mid(mcr(i), 3)

myPrompt = myPrompt & Trim(Str(i)) & " " & ChrW(8211) & " "

myPrompt = myPrompt & mName & " (" & myCode & ")" & vbCr

Next i

Do

myResponse = UCase(InputBox(myPrompt, "MacroMenu"))

myNum = Val(myResponse)

If myResponse = "" Then Exit Sub

If myNum = 0 Then

myNum = InStr(UCase(myCodes), Left(myResponse, 1))

End If

Loop Until myNum > 0 And myNum < numItems + 1

mName = Mid(mcr(myNum), 3)

Select Case mName

Case "AccentAlyse"

Call AccentAlyse

Case "CopyTextSimple"

Call CopyTextSimple

Case "DocAlyse"

Call DocAlyse

Case "FRedit"

Call FRedit

Case "HyphenAlyse"

Call HyphenAlyse

Case "IStoIZ"

Call IStoIZ

Case "IZISCount"

Call IZIScount

Case "IZtoIS"

Call IZtoIS

Case "MultiFileText"

Call MultiFileText

Case "ProperNounAlyse"

Call ProperNounAlyse

Case "SpellingErrorHighlighter"

Call SpellingErrorHighlighter

Case "SpellingErrorLister"

Call SpellingErrorLister

Case "UKUSCount"

Call UKUScount

Case "WordPairAlyse"

Call WordPairAlyse

Case "MegAlyse"

Call MegAlyse

Case Else

Beep

MsgBox "Not on list"

End Select

End Sub

**Sub HighlightAndColourSame()**

' Paul Beverley - Version 13.06.17

' Highlights and/or colours all occurrences of this text

thinSpaceHighlightColour = wdYellow

nonTextHighlight = wdGray25

myFaveHighlight = wdNoHighlight

' myFaveHighlight = wdBrightGreen

' myFaveColour = wdColorAutomatic

myFaveColour = wdColorBlue

' Preserve TC status and existing highlight colour

oldColour = Options.DefaultHighlightColorIndex

oldFind = Selection.Find.Text

oldReplace = Selection.Find.Replacement.Text

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Dim v As Variable, nowHighlight As Long

varsExist = False

For Each v In ActiveDocument.Variables

If v.Name = "selStart" Then varsExist = True: Exit For

Next v

If varsExist Then

wasStart = ActiveDocument.Variables("selStart")

wasEnd = ActiveDocument.Variables("selEnd")

If (wasEnd - wasStart) > 100 Then wasEnd = wasStart + 1

If Selection.Start > wasStart - 1 And Selection.End < wasEnd + 1 Then

Selection.Start = wasStart

Selection.End = wasEnd

End If

End If

doWholeWord = False

If Selection.End > Selection.Start Then

' Some text is selected already

Selection.MoveEndWhile cset:=" ", Count:=wdBackward

nowHighlight = Selection.range.HighlightColorIndex

If nowHighlight = 0 Then

Selection.range.HighlightColorIndex = myFaveHighlight

End If

nowColour = Selection.range.Font.Color

If nowColour = 0 Then

Selection.range.Font.Color = myFaveColour

End If

Else

' No text is selected

Selection.MoveEnd , 1

myChar = Selection

nonText = (UCase(myChar) = LCase(myChar))

If nonText = True Then

nowHighlight = Selection.range.HighlightColorIndex

If nowHighlight > 0 Then

thinSpaceHighlightColour = 0

nonTextHighlight = 0

End If

If AscW(Selection) = 8201 Then

Selection.range.HighlightColorIndex = thinSpaceHighlightColour

Else

Selection.range.HighlightColorIndex = nonTextHighlight

End If

Else

' if it's an alpha character,

' we're inside a word, so select it

nowHighlight = Selection.range.HighlightColorIndex

nowColour = Selection.range.Font.Color

If nowColour > 0 Or nowHighlight > 0 Then

myFaveColour = 0

myFaveHighlight = 0

End If

doWholeWord = True

Selection.Expand wdWord

Selection.MoveEndWhile cset:=ChrW(8217) & "' ", Count:=wdBackward

Selection.range.HighlightColorIndex = myFaveHighlight

Selection.range.Font.Color = myFaveColour

End If

End If

' What we want to (un)highlight/(un)colour is now selected

thisHighlight = Selection.range.HighlightColorIndex

thisColour = Selection.range.Font.Color

doHighlight = (thisHighlight > 0)

doColour = (thisColour > 0)

' Just in case the highlighting is mixed

' highlight as per the first character

If thisHighlight > 1000 Then

Set rng = Selection.range.Duplicate

rng.End = Selection.Start + 1

Selection.range.HighlightColorIndex = rng.HighlightColorIndex

thisHighlight = rng.HighlightColorIndex

End If

Options.DefaultHighlightColorIndex = thisHighlight

findText = Selection

Select Case Asc(findText)

Case 9: findText = "^t"

Case 30: findText = "^~": partWord = True ' non-breaking hyphen

End Select

Selection.Collapse wdCollapseStart

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = findText

.MatchCase = True

.Forward = True

.Replacement.Text = "^&"

If doHighlight Then .Replacement.Highlight = True

If doColour Then .Replacement.Font.Color = thisColour

If Not (doColour) And Not (doHighlight) Then

.Replacement.Font.Color = wdColorAutomatic

.Replacement.Highlight = False

End If

.Wrap = wdFindContinue

If doWholeWord = True Then .MatchWholeWord = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Restore to original state

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = oldColour

ActiveDocument.TrackRevisions = nowTrack

With Selection.Find

.Text = oldFind

.Replacement.Text = oldReplace

.MatchCase = False

.MatchWholeWord = False

.Replacement.Highlight = False

End With

If Selection.End = Selection.Start Then

myChar = Selection

If UCase(myChar) <> LCase(myChar) Then

Selection.Move , 1

End If

End If

End Sub

**Sub InitialiseFornamesInList()**

**'** Paul Beverley - Version 10.02.20

' Changes forenames within a list to initials

Dot = "."

Dot = ""

initialSpaced = True

noChange = "van,der,de"

Selection.Expand wdParagraph

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseStart

numWds = rng.Words.count

noChange = "," & noChange & ","

n = False

wasHyp = False

For i = numWds To 1 Step -1

w = rng.Words(i)

nWas = n

lng = (Len(Trim(w)) > 1)

If w = "-" Then lng = True

' Debug.Print rng.Words(i), w, lng, n

If lng And nWas Then

If w = "-" Then

wasHyp = True

Else

If wasHyp = True Then

rng.Words(i) = Left(w, 1) & Dot

wasHyp = False

Else

If InStr(noChange, "," & Trim(w) & ",") = 0 Then

rng.Words(i) = Left(w, 1) & Dot & " "

Else

nWas = True

End If

End If

End If

End If

If Not (lng) Then n = False

If lng And Not (nWas) Then n = True

Next i

If initialSpaced = False Then

If Dot = "" Then

For i = 1 To numWds - 3

w = rng.Words(i)

w1 = rng.Words(i + 1)

If Right(w, 1) = " " And Right(w1, 1) = " " And \_

LCase(w) <> UCase(w) And LCase(w1) <> \_

UCase(w1) Then

rng.Words(i) = Replace(w, " ", "")

numWds = numWds - 1

End If

Next i

Else

numChars = rng.Characters.count

For i = numChars - 4 To 1 Step -1

c0 = rng.Characters(i)

c3 = rng.Characters(i + 3)

If c0 = "." And c3 = "." Then rng.Characters(i + 1) = ""

Next i

End If

End If

End Sub

**Sub TrackChangeOptions()**

**'** Paul Beverley - Version 16.07.15

' Set up track changes to taste

' All TC shown in bubbles

ActiveWindow.View.MarkupMode = wdBalloonRevisions

' Insertion/deletion shown inline

' Format changes show left margin line only

ActiveWindow.View.MarkupMode = wdInLineRevisions

' Iinsertion/deletion shown inline

' Format changes shown in bubble

ActiveWindow.View.MarkupMode = wdMixedRevisions

With ActiveWindow.View

' Do you want comments showing?

.ShowComments = True

' Do you want insertions and deletions showing?

.ShowInsertionsAndDeletions = True

' Do you want format changes showing?

.ShowFormatChanges = True

End With

ActiveWindow.View.RevisionsFilter.Markup = wdRevisionsMarkupAll

'ActiveWindow.View.RevisionsFilter.Markup = wdRevisionsMarkupSimple

End Sub

**Sub QuoteExtractor()**

**'** Paul Beverley - Version 23.03.20

' Copies paragraph that contain italic text and are indented or contain quotes

Set myDoc = ActiveDocument

myMix = 9999999

Documents.Add

For Each myPar In myDoc.Paragraphs

If myPar.Range.Italic = myMix Then

copyIt = False

If InStr(myPar.Range.Text, ChrW(8216)) Then copyIt = True

If InStr(myPar.Range.Text, ChrW(8220)) Then copyIt = True

If myPar.Range.ParagraphFormat.LeftIndent > 0 Then copyIt = True

If copyIt = True Then

Selection.Collapse wdCollapseEnd

Selection.FormattedText = myPar.Range.FormattedText

End If

End If

Next myPar

End Sub

**Sub RefCitationConvert()**

**'** Paul Beverley - Version 04.04.20

' Converts superscripted ref. citations to bracketed

myPuncts = ",.:;)"

doItalic = True

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = wdBrightGreen

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Superscript = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.Replacement.Font.Color = wdColorBlue

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9\-," & ChrW(8211) & "]{1,}"

.Wrap = False

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

Set preText = rng.Duplicate

preText.Start = preText.End - 1

preText.Select

If preText.Font.Superscript = False Then \_

rng.End = rng.End - 1

Set preText = rng.Duplicate

preText.Select

preText.End = preText.Start + 1

If preText.Font.Superscript = False Then \_

rng.Start = rng.Start + 1

preText.Select

myCol = rng.Font.Color

If myCol > 0 And rng.Font.Superscript = True Then

Set preText = rng.Duplicate

preText.End = preText.Start + 1

Do

preText.Start = preText.Start - 1

preText.End = preText.End - 1

DoEvents

Loop Until preText.Font.Superscript = False

pun = preText.Text

cit = rng.Text

Debug.Print pun, cit

' If citation follows punctuation

If InStr(myPuncts, pun) > 0 Then

preText.Delete

rng.Delete

cit = Replace(cit, ChrW(8211), "-")

cit = Replace(cit, ", ", ",")

rng.InsertAfter Text:=" (zczc" & cit & "czcz)" & pun

rng.Font.Subscript = False

rng.Font.Superscript = False

rng.Start = rng.Start + 1

rng.HighlightColorIndex = wdYellow

rng.Collapse wdCollapseEnd

cit = ""

End If

' If citation follows Lcase character

If Len(cit) > 1 And LCase(pun) = pun Then

rng.Delete

cit = Replace(cit, ChrW(8211), "-")

cit = Replace(cit, ", ", ",")

myString = "(zczc" & cit & "czcz)"

If pun = " " Then

myString = myString & " "

Else

myString = " " & myString

End If

rng.InsertAfter Text:=myString

rng.Font.Subscript = False

rng.Font.Superscript = False

rng.Start = rng.Start + 1

rng.HighlightColorIndex = wdYellow

rng.Collapse wdCollapseEnd

End If

Else

rng.Start = endNow

End If

rng.Find.Execute

DoEvents

Loop

Options.DefaultHighlightColorIndex = wdYellow

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zczc(\*)czcz"

.Wrap = wdFindContinue

.Replacement.Text = "\1"

If doItalic = True Then .Replacement.Font.Italic = True

.Replacement.Font.Subscript = False

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

Beep

End Sub

**Sub SpaceEquationsInPara()**

**'** Paul Beverley - Version 08.06.17

' Adds spaces to MathType equation in this para if necessary

okBeforeChars = " (."

okAfterChars = " ),!:;."

Selection.Expand wdParagraph

theEnd = Selection.End

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^1"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.Execute

End With

Do While rng.Find.Found = True And rng.Start < theEnd

rng.Select

Selection.Collapse wdCollapseStart

Selection.Move , -1

Selection.MoveEnd , 1

If InStr(okBeforeChars, Selection) = 0 Then

Selection.InsertAfter Text:=" "

theEnd = theEnd + 1

End If

Selection.Find.Execute

Selection.Collapse wdCollapseEnd

Selection.Move , 2

Selection.MoveStart , -1

If InStr(okAfterChars, Selection) = 0 Then

Selection.Collapse wdCollapseStart

Selection.TypeText " "

Selection.MoveStart , -1

Selection.Range.HighlightColorIndex = wdNoHighlight

theEnd = theEnd + 1

End If

rng.Collapse wdCollapseEnd

' Selection.Move, 1

'rng.Start = theEnd

rng.Find.Execute

Loop

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub GoogleFetchIndia()**

**'** Paul Beverley - Version 18.11.21

' Launches selected text on Google

mySite = "http://www.google.co.in/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub ColourPlusAttribute()**

**'** Paul Beverley - Version 06.06.23

' Applies font/highlight colour plus an attribute

' roundToWholeWord = False

roundToWholeWord = True

' myFontColour = 0

myFontColour = wdColorGreen

' myHighlightColour = wdYellow

myHighlightColour = wdNoHighlight

addItalic = True

addBold = False

addUnderline = False

If roundToWholeWord = True Then

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

Else

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

End If

If myFontColour <> 0 Then \_

Selection.Font.Color = myFontColour

If myHighlightColour <> 0 Then \_

Selection.Range.HighlightColorIndex = myHighlightColour

If addItalic = True Then Selection.Font.Italic = True

If addBold = True Then Selection.Font.Bold = True

If addUnderline = True Then Selection.Font.Underline = True

Selection.Collapse wdCollapseEnd

End Sub

**Sub LanguageSetMulti()**

**'** Paul Beverley - Version 07.06.23

' Toggles between different language country settings

' Loop through the first <this number> of languages

numLanguages = 2

' Adjust order and number of languages to taste

myLanguage1 = wdEnglishUK

myLanguage2 = wdEnglishUS

myLanguage3 = wdFrench

myLanguage4 = wdEnglishNZ

myLanguage5 = wdEnglishAU

On Error GoTo ReportIt

nowLanguage = Selection.LanguageID

ReDim myLang(numLanguages)

For i = 1 To numLanguages

Select Case i

Case 1: myLang(i) = myLanguage1

Case 2: myLang(i) = myLanguage2

Case 3: myLang(i) = myLanguage3

Case 4: myLang(i) = myLanguage4

Case 5: myLang(i) = myLanguage5

End Select

DoEvents

Next i

myLanguage = myLang(numLanguages)

For i = 1 To numLanguages

If nowLanguage = myLang(i) Then

nextNum = i Mod numLanguages + 1

myLanguage = myLang(nextNum)

End If

Next i

' If you want all the styles to be language-set as well,

' make it true:

setStyleLanguage = False

' setStyleLanguage = True

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

ActiveDocument.Content.LanguageID = myLanguage

For Each aStory In ActiveDocument.StoryRanges

Do

aStory.LanguageID = myLanguage

If aStory.NextStoryRange Is Nothing Then

MoreStoryRanges = False

Else

MoreStoryRanges = True

Set aStory = aStory.NextStoryRange

End If

Loop While MoreStoryRanges

Next aStory

If ActiveDocument.Shapes.count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

shp.TextFrame.TextRange.LanguageID = myLanguage

End If

End If

Next

End If

If setStyleLanguage = True Then

For i = 1 To ActiveDocument.Styles.count

Set sty = ActiveDocument.Styles(i)

If sty.Type = wdStyleTypeParagraph Then \_

sty.LanguageID = myLanguage

Next i

End If

ActiveDocument.Styles(wdStyleNormal).LanguageID = myLanguage

ActiveDocument.Styles(wdStyleCommentText).LanguageID = myLanguage

ActiveDocument.SpellingChecked = False

ActiveDocument.GrammarChecked = False

ActiveDocument.TrackRevisions = myTrack

Exit Sub

ReportIt:

myErr = Err.Number

If myErr = 5843 Then

myPrompt = "You don't have language ""myLanguage" & Trim(Str(i - 1)) & \_

""" installed."

myResponse = MsgBox(myPrompt, vbQuestion + vbOKOnly, "LanguageSetMulti")

DoEvents

myLanguage = myLanguage1

Resume Next

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub SelectionRoundOff()**

**'** Paul Beverley - Version 01.09.23

' Extends the existing selection to the nearest word end/start

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

rng.Select

End Sub

**Sub NotesEmbedSquareBrackets()**

**'** Paul Beverley - Version 09.06.23

' Creates endnotes from text-based notes, numbered in square brackets

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[[0-9]{1,}\]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

noteNum = rng.Text

rng.Cut

Set rngMarker = rng.Duplicate

With rng.Find

.Text = noteNum

.MatchWildcards = False

.Execute

End With

rng.End = rng.End + 1

rng.Delete

rng.Start = rng.Start + 1

rng.Expand wdParagraph

rng.Cut

With rngMarker

With .EndnoteOptions

.Location = wdBottomOfPage

.NumberingRule = wdRestartContinuous

.StartingNumber = 1

.NumberStyle = wdNoteNumberStyleArabic

End With

.Endnotes.Add Range:=rngMarker, Reference:=""

End With

numEnds = ActiveDocument.Endnotes.count

ActiveDocument.Endnotes(numEnds).Range.Select

Selection.Collapse wdCollapseEnd

Selection.Paste

' Restart search for [number]

rng.End = myMarker

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[[0-9]{1,}\]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

DoEvents

Loop

End Sub

**Sub NavigationPanePrepare()**

**'** Paul Beverley - Version 16.06.23

' Copies text and prepares for Navigation pane (only needed for Macs)

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

Selection.Copy

If Application.CommandBars("Navigation").Visible = False Then Application.CommandBars("Navigation").Visible = True

End Sub

**Sub PlurAlyse()**

**'** Paul Beverley - Version 29.01.24

' Creates a frequency list of single/plural pairs

minLength = 3

Dim myResult As String

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox("Run PlurAlyse ?", vbQuestion \_

+ vbYesNoCancel, "PlurAlyse")

If myResponse <> vbYes Then Exit Sub

End If

Set testDoc = ActiveDocument

strttime = Timer

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

Selection.HomeKey Unit:=wdStory

Application.ScreenUpdating = False

On Error GoTo ReportIt

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "'"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = " "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

rng.Text = " " & vbCr & vbCr & LCase(rng.Text)

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

Set rng2 = ActiveDocument.Content

wasLength = rng2.End

myWild = "<[a-zA-Z]@ies>"

For i = 3 To 1 Step -1

If i = 2 Then myWild = "<[a-zA-Z]@es>"

If i = 1 Then myWild = "<[a-zA-Z]@s>"

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myWild

.Font.StrikeThrough = False

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

End With

Do While rng.Find.Found = True

endWas = rng.End

myLong = rng.Text

rng.MoveEnd , -i

If i = 3 Then

myShort = rng.Text & "y"

Else

myShort = rng.Text

End If

Selection.MoveRight , 1

' Does the singular occur?...

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myShort

.MatchWholeWord = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

If rng2.Find.Found = True And Len(rng2.Text) > (minLength - 1) Then

rng2.Collapse wdCollapseEnd

' ... if so, then count occurrences

DoEvents

Application.ScreenUpdating = True

ActiveDocument.Paragraphs(1).Range.Text = myShort & vbCr

ActiveDocument.Paragraphs(1).Range.Select

Selection.Collapse wdCollapseStart

DoEvents

Application.ScreenUpdating = False

ActiveDocument.Paragraphs(1).Range.Text = vbCr

DoEvents

wasLength = ActiveDocument.Content.End

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myShort

.Wrap = wdFindContinue

.MatchWholeWord = True

.Replacement.Text = "^&!"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

numSingular = ActiveDocument.Content.End - wasLength

WordBasic.EditUndo

myResult = myResult & myShort & vbTab & Trim(Str(numSingular))

DoEvents

wasLength = ActiveDocument.Content.End

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myLong

.Wrap = wdFindContinue

.MatchWholeWord = True

.Replacement.Text = "^&!"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

numPlural = ActiveDocument.Content.End - wasLength

WordBasic.EditUndo

myResult = myResult & vbTab & myLong & vbTab & \_

Trim(Str(numPlural)) & vbCr

End If

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myLong

.Wrap = wdFindContinue

.MatchWholeWord = True

.Replacement.Text = "^&"

.Replacement.Font.StrikeThrough = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

rng.End = endWas

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Next i

Selection.WholeStory

Selection.Style = ActiveDocument.Styles(wdStyleNormal)

Selection.TypeText Text:=myResult

Set rng = ActiveDocument.Content

rng.Sort

Selection.HomeKey Unit:=wdStory

Selection.MoveEnd , 2

Selection.TypeText "Plurals use"

startTable = Selection.End + 1

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

Selection.Start = startTable

Selection.End = ActiveDocument.Range.End

Selection.ConvertToTable Separator:=wdSeparateByTabs

timNow = Timer

Application.ScreenUpdating = True

If doingSeveralMacros = False Then

timGone = timNow - strttime

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

m = Int(timGone / 60)

s = Int(timGone) - m \* 60

timeAll = "Time: " & Trim(Str(m)) & " m " & \_

Trim(Str(s)) & " s"

Selection.HomeKey Unit:=wdStory

numPairs = ActiveDocument.Tables(1).Rows.Count

MsgBox "Items: " & Trim(Str(numPairs)) & vbCr & vbCr \_

& timeAll

Else

FUT.Activate

End If

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub PlurAlyseColour()**

**'** Paul Beverley - Version 19.06.23

' Colours/highlights all words listed by PlurAlyse

' myHiColour = wdBrightGreen

myHiColour = 0

myFontColour = wdColorBlue

' myFontColour = 0

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myHiColour

gottaList = False

For Each myDoc In Application.Documents

DoEvents

pNum = myDoc.Paragraphs.count

myNum = 3

If pNum < 3 Then myNum = pNum

Set rng = myDoc.Paragraphs(myNum).Range

rng.Start = 0

If InStr(LCase(rng), "plural") Then

Set theList = myDoc

gottaList = True

End If

Next myDoc

If gottaList = False Then

Beep

myResponse = MsgBox("Can't find a PlurAlyse table." & CR2 & \_

myWarning, vbExclamation + vbOKOnly, "PlurAlyseColour")

Exit Sub

End If

Set rng = ActiveDocument.Content

For i = 1 To theList.Tables(1).Rows.count

myText = theList.Tables(1).Rows(i).Cells(1).Range.Text

myText = Left(myText, Len(myText) - 2)

DoEvents

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText

.Wrap = wdFindContinue

.Forward = True

If myHiColour <> 0 Then .Replacement.Highlight = True

If myFontColour <> 0 Then .Replacement.Font.Color = myFontColour

.Replacement.Text = ""

.MatchWildcards = False

.MatchWholeWord = True

.Execute Replace:=wdReplaceAll

myText = theList.Tables(1).Rows(i).Cells(3).Range.Text

myText = Left(myText, Len(myText) - 2)

DoEvents

.Text = myText

.Execute Replace:=wdReplaceAll

DoEvents

End With

Next i

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub AndToAmpersand()**

**'** Paul Beverley - Version 24.06.23

' Changes the next citation from "and" to "&"

myColour = wdColorBlue

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " and "

.Replacement.Text = " & "

.Wrap = wdFindStop

.Forward = True

.Replacement.Highlight = True

.MatchWildcards = True

.Execute

DoEvents

If .Found = True Then

.Execute Replace:=wdReplaceOne

Selection.Collapse wdCollapseEnd

Else

Beep

Exit Sub

End If

.ClearFormatting

.Replacement.ClearFormatting

.Text = "??[0-9]{4}"

.Replacement.Text = ""

.Wrap = wdFindStop

.Forward = True

.MatchWildcards = True

.Execute

DoEvents

If .Found = True Then

If Left(Selection, 1) <> "," Then

Selection.MoveStart , 1

Selection.InsertBefore Text:=","

Selection.Collapse wdCollapseStart

Else

Selection.Collapse wdCollapseEnd

End If

End If

End With

End Sub

**Sub AndToAmpersandGlobal()**

**'** Paul Beverley - Version 24.06.23

' Changes all citations from "and" to "&"

myColour = wdColorBlue

myResponse = MsgBox("Change ""and"" to ""&"" in all citations?!", \_

vbQuestion + vbYesNo, "AndToAmpersandGlobal")

If myResponse <> vbYes Then Exit Sub

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "([A-Z][a-z]{1,}) and ([A-Z][a-z]{1,}, [0-9]{4})"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "\1 & \2"

.Replacement.Font.Color = myColour

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Text = "([A-Z][a-z]{1,}) and ([A-Z][a-z]{1,}) ([0-9]{4})"

.Replacement.Text = "\1 & \2, \3"

.Execute Replace:=wdReplaceAll

DoEvents

End With

If ActiveDocument.Footnotes.count > 0 Then

Set foots = ActiveDocument.StoryRanges(wdFootnotesStory)

With foots.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "([A-Z][a-z]{1,}) and ([A-Z][a-z]{1,}, [0-9]{4})"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "\1 & \2"

.Replacement.Font.Color = myColour

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Text = "([A-Z][a-z]{1,}) and ([A-Z][a-z]{1,}) ([0-9]{4})"

.Replacement.Text = "\1 & \2, \3"

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

If ActiveDocument.Endnotes.count > 0 Then

Set ends = ActiveDocument.StoryRanges(wdEndnotesStory)

With ends.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "([A-Z][a-z]{1,}) and ([A-Z][a-z]{1,}, [0-9]{4})"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "\1 & \2"

.Replacement.Font.Color = myColour

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Text = "([A-Z][a-z]{1,}) and ([A-Z][a-z]{1,}) ([0-9]{4})"

.Replacement.Text = "\1 & \2, \3"

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub AmpersandAdd()**

**'** Paul Beverley - Version 23.06.23

' Adds an ampersand in front of the current name

serialComma = False

serialComma = True

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

If serialComma = False Then Selection.MoveStart , -1

Selection.TypeText Text:=" &"

End Sub

**Sub NTbackupDated()**

**'** Paul Beverley - Version 20.07.24

' Backs up the Normal template and the VBA code with date and time

backupFolder = "Backup"

mySuffix = " " & Format((Year(Now() + 1) Mod 100), "20##") & "-" & \_

Format((Month(Now() + 1) Mod 100), "0#") & "-" & \_

Format((Day(Now()) Mod 100), "0#") & "-" & \_

Format(Now(), "HH\_mm") 'add date & time

NTname = NormalTemplate.FullName

If InStr(NTname, "\") = 0 Then

backupFolder = backupFolder & "/"

Else

backupFolder = backupFolder & "\"

End If

newName = Replace(NTname, "Normal", backupFolder & "Normal" & mySuffix)

' Put the VBA code in a Word file

fileName = Replace(newName, "dotm", "docx")

fileName = Replace(fileName, "Normal", "The Macros")

Documents.Add

Selection.Paste

For i = 1 To 100

DoEvents

Next i

If InStr(ActiveDocument.Content.Text, "End Sub") = 0 Then

Beep

MsgBox ("You forgot to copy the macros!")

Exit Sub

End If

Debug.Print newName

Debug.Print fileName

ActiveDocument.SaveAs fileName

For i = 1 To 100

DoEvents

Next i

ActiveDocument.Close SaveChanges:=False

' Save the Normal template

ThisDocument.SaveAs newName

Beep

MsgBox ("Backup completed, including the macro text you selected.")

End Sub

**Sub SelectionEndExtend()**

**'** Paul Beverley - Version 03.07.23

' Moves the end of the selection further out to the right (i.e. extends it)

If Selection.Start = Selection.End Then

Selection.Expand wdWord

lenNow = Len(Selection)

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

If Len(Selection) <> lenNow Then Exit Sub

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.MoveStart , -1

preChar = rng.Text

If (LCase(preChar) = UCase(preChar)) Then Exit Sub

End If

Selection.MoveEnd , 1

preChar = Right(Selection, 1)

If LCase(preChar) <> UCase(preChar) Or \_

preChar = "0" Or Val(preChar) > 0 Then

Selection.MoveEnd , -1

Selection.MoveEnd wdWord, 1

End If

End Sub

**Sub SelectionEndShrink()**

**'** Paul Beverley - Version 03.07.23

' Pulls the end of the selection back to the left (i.e. shrinks it)

preChar = Right(Selection, 1)

If LCase(preChar) <> UCase(preChar) \_

Or preChar = "0" Or Val(preChar) <> 0 Then

Selection.MoveEnd wdWord, -1

Else

Selection.MoveEnd wdCharacter, -1

End If

End Sub

**Sub SelectionStartExtend()**

**'** Paul Beverley - Version 03.07.23

' Moves the start of the selection further out to the left (i.e. extends it)

If Selection.Start = Selection.End Then

Selection.Expand wdWord

lenNow = Len(Selection)

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

If Len(Selection) <> lenNow Then Exit Sub

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.MoveStart , -1

preChar = rng.Text

If (LCase(preChar) = UCase(preChar)) Then Exit Sub

End If

Selection.MoveStart , -1

preChar = Left(Selection, 1)

If LCase(preChar) = UCase(preChar) Then

If preChar <> "0" And Val(preChar) = 0 Then

Exit Sub

End If

End If

Selection.MoveStart , 1

Selection.MoveStart wdWord, -1

End Sub

**Sub SelectionStartShrink()**

**'** Paul Beverley - Version 03.07.23

' Pulls the start of the selection back to the right (i.e. shrinks it)

preChar = Left(Selection, 1)

If LCase(preChar) <> UCase(preChar) \_

Or preChar = "0" Or Val(preChar) <> 0 Then

Selection.MoveStart wdWord, 1

Else

Selection.MoveStart wdCharacter, 1

End If

End Sub

**Sub QQConvert()**

**'** Paul Beverley - Version 14.06.21

' Converts QQ comments to ordinary comments

Dim qqTag(999) As String

allText = ActiveDocument.Content

For i = ActiveDocument.Endnotes.count To 1 Step -1

thisTag = ActiveDocument.Endnotes(i).Range.Text

If InStr(allText, thisTag) = 0 Then \_

ActiveDocument.Endnotes(i).Delete

Next i

For i = 1 To ActiveDocument.Endnotes.count

qqTag(i) = ActiveDocument.Endnotes(i).Range.Text

Next i

Set rng = ActiveDocument.Content

For i = 1 To ActiveDocument.Endnotes.count

Debug.Print ActiveDocument.Endnotes.count, "|", i

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = qqTag(i) & " "

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = tagText

.MatchWildcards = False

.Execute

DoEvents

End With

rng.Collapse wdCollapseEnd

rng.End = ActiveDocument.Content.End

qqPos = InStr(rng, "[qq")

If qqPos > 0 Then rng.End = rng.Start + qqPos - 1

Do While Right(rng.Text, 1) = vbCr

rng.End = rng.End - 1

Loop

Debug.Print rng.Text

' rng.Select

rng.Copy

ActiveDocument.Endnotes(i).Reference.Select

Selection.Collapse wdCollapseStart

Set rng = Selection.Range.Duplicate

myEnd = rng.End

j = 0

Do

j = j + 1

rng.End = myEnd - j

rng.Start = rng.End - 1

Loop Until rng.Font.Underline = False

Selection.Start = rng.End

Selection.End = myEnd

DoEvents

Selection.Comments.Add Range:=Selection.Range

Selection.Paste

Next i

ActiveWindow.ActivePane.Close

CommandBars("Comments").Visible = True

End Sub

**Sub QQUpdate()**

**'** Paul Beverley - Version 28.05.21

' Renumbers QQ comments

Dim qqTag(999) As String

allText = ActiveDocument.Content

For i = ActiveDocument.Endnotes.count To 1 Step -1

thisTag = ActiveDocument.Endnotes(i).Range.Text

If InStr(allText, thisTag) = 0 Then \_

ActiveDocument.Endnotes(i).Delete

Next i

For i = 1 To ActiveDocument.Endnotes.count

qqTag(i) = ActiveDocument.Endnotes(i).Range.Text

Next i

' Update the numbering, starting at [qq001]

Set rng = ActiveDocument.Content

For i = 1 To ActiveDocument.Endnotes.count

tagText = "[qq" & Right(Trim(Str(1000 + i)), 3) & "]"

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = qqTag(i)

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = tagText

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

DoEvents

End With

ActiveDocument.Endnotes(i).Range.Text = tagText

DoEvents

Next i

' Read the page/line numbering, starting at [qq001]

Set rng = ActiveDocument.Content

For i = 1 To ActiveDocument.Endnotes.count

ActiveDocument.Endnotes(i).Range.Select

ActiveWindow.View.SeekView = wdSeekMainDocument

tagText = "[qq" & Right(Trim(Str(1000 + i)), 3) & "]"

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = qqTag(i)

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = tagText

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

DoEvents

End With

DoEvents

Next i

' Update the page/line numbering, starting at [qq001]

End Sub

**Sub** **HighlightSpecialCharacters()**

**'** Paul Beverley - Version 06.07.23

' Highlights all special characters in a file

myHighlightColour = wdBrightGreen

' myHighlightColour = wdNoHighlight

' myTextColour = wdColorBlue

myTextColour = wdColorBlack

oldColour = Options.DefaultHighlightColorIndex

If myHighlightColour > 0 Then \_

Options.DefaultHighlightColorIndex = myHighlightColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[!A-Za-z 0-9.,;:\-\?\!^0001-^0064+/" & \_

ChrW(8216) & ChrW(8217) & ChrW(8220) & ChrW(8221) \_

& "^=^+=~\<\>\{\}\@…]"

.Wrap = wdFindContinue

.Replacement.Text = "^&"

If myTextColour > 0 Then \_

.Replacement.Font.Color = myTextColour

If myHighlightColour > 0 Then \_

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub FReditListMenuFixedFiles()**

**'** Paul Beverley - Version 23.01.24

' Allows fixed filenames, for if FReditListMenu doesn't work for you

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\MyLists\"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/MyLists/"

numLIsts = 5

ReDim myList(numLIsts) As String

ReDim myLetter(numLIsts) As String

myList(1) = "CleanupList.doc"

myList(2) = "h - HighlightThe.docx"

myList(3) = "FinalDoMacros.docx"

myList(4) = "f - HighlightAllETs.docx"

myList(5) = "InitialDoMacros.docx"

n = 0

s = ";"

allLetters = s

For i = 1 To numLIsts

myLetter(i) = LCase(Left(myList(i), 1))

If InStr(allLetters, s & myLetter(i) & s) > 0 Then

myLetter(i) = myLetter(i) & myLetter(i)

If InStr(allLetters, s & myLetter(i) & s) > 0 Then

myResponse = MsgBox("Please don't have three items" & \_

" with the same initial letter.", \_

vbOKCancel, "FReditListMenu")

Exit Sub

End If

End If

allLetters = allLetters & myLetter(i) & s

DoEvents

Next i

For i = 1 To numLIsts

thisName = Replace(myList(i), ".docx", "")

thisName = Replace(myList(i), ".doc", "")

myMenu = myMenu & myLetter(i) & \_

" - " & thisName & vbCr

DoEvents

Next i

myMenu = myMenu & vbCr & "Which list (letter)?"

myChoice = InputBox(myMenu, "FReditListMenu")

For i = 1 To numLIsts

If myLetter(i) = LCase(myChoice) Then

goodChoice = True

doList = myList(i)

Exit For

End If

DoEvents

Next i

If i < numLIsts + 1 Then myChoice = myList(i)

If myChoice = "" Then Exit Sub

Set myDoc = ActiveDocument

Documents.Open fileName:=myFolder & myChoice

Set listDoc = ActiveDocument

myDoc.Activate

Call FRedit

listDoc.Close SaveChanges:=False

End Sub

**Sub MovePreviousSentence()**

**'** Paul Beverley - Version 12.07.23

' Moves the cursor ro the start of the next sentence

Selection.Expand wdSentence

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.Expand wdSentence

Selection.Collapse wdCollapseStart

End Sub

**Sub MoveNextSentence()**

**'** Paul Beverley - Version 18.10.22

' Moves the cursor ro the start of the next sentence

Selection.Expand wdSentence

Selection.Collapse wdCollapseEnd

End Sub

**Sub AcronymsToSmallCaps()**

**'** Paul Beverley - Version 01.02.20

' Finds all acronyms (in text or selection) and changes to small caps

minLength = 3

If Selection.Start = Selection.End Then

myResponse = MsgBox("Whole of the text?", \_

vbQuestion + vbYesNoCancel, "AcronymsToSmallCaps")

If myResponse <> vbYes Then Exit Sub

theStart = ActiveDocument.Content.Start

theEnd = ActiveDocument.Content.End

Else

theStart = Selection.Range.Start

theEnd = Selection.Range.End

End If

mySearch = "<[A-Z]{" & Trim(Str(minLength)) & ",}>"

Set rng = ActiveDocument.Content

rng.Start = theStart

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = mySearch

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

If rng.End < theEnd Then

myCount = myCount + 1

myEnd = rng.End

rng.Text = LCase(rng.Text)

rng.Font.SmallCaps = True

rng.Start = myEnd

rng.Find.Execute

DoEvents

Else

Exit Do

End If

Loop

MsgBox "Changed: " & myCount & " small caps"

End Sub

**Sub CountWordsToCursor()**

**'** Paul Beverley - Version 29.01.18

' Counts how many words in the paragraph up to the cursor

myStart = Selection.Start

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

i = 0

Do

i = i + 1

Selection.MoveRight wdWord, 1

Loop Until Selection.Start > myStart

MsgBox i

End Sub

**Sub DeSexer()**

**'** Paul Beverley - Version 18.01.20

' Finds one of your words, then calls MultiSwitch to change it

myWords = "man,he,him,his"

myRange = 200

myWords = "," & myWords & ","

For i = 1 To myRange

Selection.MoveRight wdWord, 1

Selection.Expand wdWord

Selection.MoveEndWhile cset:=ChrW(8217) & " '", count:=wdBackward

If InStr(myWords, "," & Selection & ",") > 0 Then Exit For

DoEvents

Next i

If i > myRange Then

Beep

Else

Application.Run macroName:="MultiSwitch"

End If

End Sub

**Sub DialoguePunctuationSwitch()**

**'** Paul Beverley - Version 12.09.19

' Switches adjacent word link between comma and full point

newBit = "." & ChrW(8221) & " "

' US users:

' newBit = ChrW(8221) & ". "

stopCaseChange = (Selection.Start <> Selection.End)

Selection.Words(1).Select

Selection.Collapse wdCollapseEnd

myStart = Selection.Start

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-zA-Z]"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute

End With

If stopCaseChange = False Then

If LCase(Selection) = Selection Then

Selection = UCase(Selection)

Else

Selection = LCase(Selection)

End If

End If

Selection.Collapse wdCollapseStart

Selection.Start = myStart

If InStr(Selection, ChrW(8217)) > 0 Then \_

newBit = Replace(newBit, ChrW(8221), ChrW(8217))

If InStr(Selection, ".") > 0 Then \_

newBit = Replace(newBit, ".", ",")

Selection.TypeText newBit

End Sub

**Sub DraftViewToggle()**

**'** Paul Beverley - Version 05.12.18

' Switches between draft view and print view

If ActiveWindow.ActivePane.View.Type = wdPrintView Then

ActiveWindow.ActivePane.View.Type = wdNormalView

Else

ActiveWindow.View.Type = wdPrintView

End If

End Sub

**Sub FeatureCounter()**

**'** Paul Beverley - Version 24.08.17

' Counts various feature of the current Word file

myResult = ""

CR = vbCr

n = ActiveDocument.Endnotes.count

If n > 0 Then myResult = myResult & "Endnotes: " & Str(n) & CR

n = ActiveDocument.Footnotes.count

If n > 0 Then myResult = myResult & "Footnotes: " & Str(n) & CR

' count all the textboxes that have text

shCount = myDoc.Shapes.count

If shCount > 0 Then

n = 0

For j = 1 To shCount

Set shp = ActiveDocument.Shapes(j)

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then n = n + 1

End If

Next

myResult = myResult & "Textboxes: " & Str(n) & CR

End If

' Count all equations

n = ActiveDocument.InlineShapes.count

If n > 0 Then myResult = myResult & "MathType: " & Str(n) & CR

n = ActiveDocument.OMaths.count

If n > 0 Then myResult = myResult & "Equation Editor: " & Str(n) & CR

Selection.HomeKey Unit:=wdStory

Selection.TypeText myResult & CR & CR

End Sub

**Sub FindSamePageLine()**

**'** Paul Beverley - Version 17.12.12

' Find the same place in another file by page and line

matchNameChars = 6

Selection.Collapse wdCollapseEnd

myPage = Selection.Information(wdActiveEndAdjustedPageNumber)

myLine = Selection.Information(wdFirstCharacterLineNumber)

Set firstDoc = ActiveDocument

firstDocPartName = Left(firstDoc.Name, matchNameChars)

For Each myWnd In Application.Windows

Set secondDoc = myWnd.Document

If Left(secondDoc.Name, matchNameChars) = firstDocPartName \_

And secondDoc.Name <> firstDoc.Name Then

secondDoc.Activate

If myWnd.WindowState = 2 Then myWnd.WindowState = wdWindowStateNormal

Selection.GoTo What:=wdGoToPage, count:=myPage

Selection.MoveDown Unit:=wdLine, count:=myLine

Selection.MoveUp Unit:=wdLine, count:=1, Extend:=wdExtend

Exit Sub

End If

Next myWnd

End Sub

**Sub FontMixFind()**

**'** Paul Beverley - Version 16.05.21

' Finds the next paragraph that has mixed fonts

Set rng = ActiveDocument.Range(0, Selection.End)

myStart = rng.Paragraphs.count

For i = myStart + 1 To ActiveDocument.Paragraphs.count

Set rng = ActiveDocument.Paragraphs(i).Range

If rng.Font.Name = "" Then

nowFont = rng.Characters(1).Font.Name

For Each wd In rng.Words

If wd.Font.Name <> nowFont Then

wd.Select

Exit Sub

End If

Next wd

End If

Next i

Selection.EndKey Unit:=wdStory

Beep

End Sub

**Sub FontMixHighlight()**

**'** Paul Beverley - Version 17.05.21

' Highlights paragraphs with mixed fonts, showing odd fonts

myParaColour = wdGray25

myDiffFontColour = wdBrightGreen

For i = 1 To ActiveDocument.Paragraphs.count

Set rng = ActiveDocument.Paragraphs(i).Range

If rng.Font.Name = "" Then

rng.HighlightColorIndex = myParaColour

nowFont = rng.Characters(1).Font.Name

For Each wd In rng.Words

If wd.Font.Name <> nowFont Then

wd.HighlightColorIndex = myDiffFontColour

End If

Next wd

End If

Next i

Selection.EndKey Unit:=wdStory

Beep

End Sub

**Sub FontNameAndSizeReader()**

**'** Paul Beverley - Version 21.11.18

' Reads style font Name + any applied Name

myMixName = ""

paraName = ActiveDocument.Styles(Selection.Range.Style).Font.Name

myName = Selection.Range.Font.Name

myMessage = ""

myMessage = myMessage & "Style font name = " & paraName & vbCr

If myName = myMixName Then

myMessage = myMessage & "Mixed names" & vbCr

Else

If paraName = myName Then

myMessage = myMessage & "No applied name" & vbCr

Else

myMessage = myMessage & "Applied name = " & myName & vbCr

End If

End If

MsgBox myMessage

myMix = 9999999

paraSize = ActiveDocument.Styles(Selection.Range.Style).Font.Size

mySize = Selection.Range.Font.Size

myMessage = ""

myMessage = myMessage & "Style font Size = " & paraSize & vbCr

If mySize = myMix Then

myMessage = myMessage & "Mixed Sizes" & vbCr

Else

If paraSize = mySize Then

myMessage = myMessage & "No applied size" & vbCr

Else

myMessage = myMessage & "Applied size = " & mySize & vbCr

End If

End If

MsgBox myMessage

End Sub

**Sub FootnoteReferenceExtract()**

**'** Paul Beverley - Version 11.03.17

' Extracts references from footnotes, leaving Harvard citation

minYear = 1900

maxYear = 2020

myColour = wdBrightGreen

numFNs = ActiveDocument.Footnotes.count

CR = vbCr

CR2 = vbCr & vbCr

Set rng = ActiveDocument.Content

rng.Collapse wdCollapseEnd

rng.InsertAfter CR2 & "References" & CR2

For i = numFNs To 1 Step -1

isAref = False

Set fn = ActiveDocument.Footnotes(i)

If fn.Range.Words(3).HighlightColorIndex = 0 Then

numWds = fn.Range.Words.count

For j = 1 To numWds

myWd = Trim(fn.Range.Words(j))

If Len(myWd) = 4 Then

myYear = Val(myWd)

If myYear > minYear And myYear < maxYear Then

isAref = True

Exit For

End If

End If

Next j

End If

If isAref Then

fn.Range.Copy

Selection.EndKey Unit:=wdStory

myEnd = Selection.End

Selection.Paste

Selection.TypeText CR

Selection.End = myEnd

Selection.MoveEndUntil cset:=",", count:=wdForward

Selection.Copy

Selection.GoTo What:=wdGoToFootnote, count:=i

Selection.MoveStart , -1

If InStr(",.:;!?", Selection) > 0 Then

Selection.Collapse wdCollapseStart

Else

Selection.Collapse wdCollapseEnd

End If

myStart = Selection.Start

Selection.TypeText " ("

Selection.Paste

Selection.TypeText ", " & Trim(Str(myYear)) & ")"

Selection.Start = myStart

Selection.Range.HighlightColorIndex = myColour

fn.Delete

End If

Next i

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub FullPointMissing()**

**'** Paul Beverley - Version 06.08.15

' Add a full point to the next line that doesn't have one

Selection.Collapse wdCollapseEnd

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[!.]^13"

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindStop

.MatchCase = False

.MatchWildcards = True

.Execute

End With

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.TypeText "."

Selection.MoveStart , -1

End Sub

**Sub GivenNameToInitials()**

**'** Paul Beverley - Version 27.03.14

' Reduce given names to initial only

givenNameFirst = True

addFP = True

Selection.Expand wdParagraph

parStart = Selection.Start

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]{4}"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Selection.Start = parStart

myText = Selection

If addFP = True Then FP = "." Else FP = ""

wds = Split(myText, " ")

lastWd = UBound(wds)

' First deal with the first author's given name

n = wds(1)

newone = Left(n, 1) & FP

If Right(n, 1) = "," Then newone = newone & ","

newText = wds(0) & " " & newone & " "

If givenNameFirst = True Then thisIsSurname = False

For i = 2 To lastWd - 1

n = wds(i)

If n = "and" Then

newText = newText & "and "

Else

If thisIsSurname = True Then

newText = newText & n & " "

Else

newone = Left(n, 1) & FP

If Right(n, 1) = "," Then newone = newone & ","

newText = newText & newone & " "

End If

thisIsSurname = Not (thisIsSurname)

End If

Next i

newText = newText & wds(lastWd)

Selection.TypeText newText

End Sub

**Sub GreekHighlighter()**

**'** Paul Beverley - Version 05.11.17

' Finds unicode Greek sections of text and highlights them

myForeign = "[" & ChrW(913) & "-" & ChrW(980) & "]"

myEnglishChars = "[A-Za-z]"

notTheseChars = vbCr & Chr(11) & " " & Chr(160)

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myForeign

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While Selection.Find.Found = True

myStart = Selection.Start

Selection.Collapse wdCollapseEnd

Selection.Find.Text = myEnglishChars

Selection.Find.Execute

myEnd = Selection.End

Selection.Collapse wdCollapseStart

Selection.MoveStartWhile cset:=notTheseChars, count:=wdBackward

Selection.Collapse wdCollapseStart

Selection.Start = myStart

Selection.Range.HighlightColorIndex = wdTurquoise

Selection.Start = myEnd

Selection.Find.Text = myForeign

Selection.Find.Execute

Loop

End Sub

**Sub HeadingLister()**

**'** Paul Beverley - Version 11.01.14

' Create a list of all coded headings

codesWanted = "PH CH A B C D"

myCodeEnds = "<>"

lCode = Left(myCodeEnds, 1)

rCode = Right(myCodeEnds, 1)

codesWanted = lCode & Trim(codesWanted) & rCode

codesWanted = Replace(codesWanted, " ", rCode & lCode)

Set rng = ActiveDocument.Content

rng.Copy

Documents.Add

Selection.Paste

Set rng = ActiveDocument.Content

rng.Revisions.AcceptAll

rng.HighlightColorIndex = wdYellow

For Each myTable In ActiveDocument.Tables

myTable.Delete

Next myTable

For Each myPar In ActiveDocument.Paragraphs

If Len(myPar.Range.Text) > 6 Then

myTest = Left(myPar.Range.Text, 6)

codePos = InStr(myTest, rCode)

If codePos > 0 Then myTest = Left(myTest, codePos)

If InStr(codesWanted, myTest) > 0 Then

myPar.Range.HighlightColorIndex = 0

End If

End If

Next myPar

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Text = "[^13]{2,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Beep

End Sub

**Sub InitialCapAlyse()**

**'** Paul Beverley - Version 17.09.18

' Highlight possible capitalisation inconsistency

'Not tried if it works

myColour1 = wdYellow

myColour2 = wdTurquoise

showTime = True

If ActiveDocument.Words.count / ActiveDocument.Paragraphs.count \_

< 10 Then GoTo analyse

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

Set rng = ActiveDocument.Content

Selection.HomeKey Unit:=wdStory

Documents.Add

Selection.Text = rng.Text

Selection.HomeKey Unit:=wdStory

For Each sn In ActiveDocument.Sentences

sn.Words(1).HighlightColorIndex = wdYellow

Next sn

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.Highlight = True

.MatchCase = False

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

Exit Sub

analyse:

timeStart = Timer

Selection.HomeKey Unit:=wdStory

Selection.TypeText vbCr

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^32{1,}"

.Wrap = wdFindContinue

.Replacement.Text = "^t"

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.Text = "^p^t"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

wordPointer = 1

testWord = ActiveDocument.Paragraphs(5).Range.Words(wordPointer)

If LCase(testWord) = UCase(testWord) Then wordPointer = 2

' Find the final actual item in the list

maxPara = ActiveDocument.Paragraphs.count

Do While Len(ActiveDocument.Paragraphs(maxPara).Range.Text) < 5

maxPara = maxPara - 1

Loop

previousWord = "dummy"

myColour = myColour1

For i = 1 To maxPara

thisWord = Trim(ActiveDocument.Paragraphs(i).Range.Words(wordPointer))

' myNumber = ActiveDocument.Paragraphs(i).Range.Words.Count

j = j + 1

If j = 30 Then

DoEvents ' Debug.Print thisWord

StatusBar = thisWord

j = 0

End If

gotOne = (LCase(thisWord) = LCase(previousWord))

If gotOne Then

' DoEvents ' Debug.Print thisWord, previousWord

paraText = ActiveDocument.Paragraphs(i).Range.Text

prevParaText = ActiveDocument.Paragraphs(i - 1).Range.Text

If InStr(paraText, "-") > 0 Then

testWord = ActiveDocument.Paragraphs(i).Range.Words(wordPointer + 2)

previousTestWord = ActiveDocument.Paragraphs(i - 1).Range.Words(wordPointer + 2)

gotOne = (LCase(testWord) = LCase(previousTestWord))

End If

If gotOne Then

ActiveDocument.Paragraphs(i).Range.HighlightColorIndex = myColour

ActiveDocument.Paragraphs(i - 1).Range.HighlightColorIndex = myColour

If myColour = myColour2 Then

myColour = myColour1

Else

myColour = myColour2

End If

End If

End If

previousWord = thisWord

Next i

totTime = Timer - timeStart

If showTime = True And totTime > 60 Then \_

MsgBox ((Int(10 \* totTime / 60) / 10) & \_

" minutes")

End Sub

**Sub InitialCapWord()**

**'** Paul Beverley - Version 18.09.15

' Initial capitalises the word at the cursor

Selection.Expand wdWord

If UCase(Selection) = Selection Then

Selection.Range.Case = wdLowerCase

Selection.Range.Characters(1).Case = wdUpperCase

Selection.Collapse wdCollapseEnd

Else

Selection.Range.Case = wdUpperCase

End If

End Sub

**Sub InsertFigureRefs()**

**'** Paul Beverley - Version 21.02.18

' Adds hyperlinked reference to all figures

Set rng = Selection.Range.Duplicate

Application.ScreenUpdating = False

myFigs = ActiveDocument.GetCrossReferenceItems(wdRefType)

numFigs = UBound(myFigs)

st = -9: en = 0

Do

st = st + 10: en = en + 10

allFigs = ""

For i = st To en

allFigs = allFigs & i & myFigs(i) & vbCr

If i = numFigs Then Exit For

Next i

allFigs = allFigs & vbCr & vbCr & "Reference number?"

thisNumber = InputBox(allFigs, "Add table reference")

Loop Until en > numFigs Or Val(thisNumber) > 0

Application.ScreenUpdating = True

rng.Select

If Val(thisNumber) = 0 Then

Beep

Else

Selection.InsertCrossReference ReferenceType:="Figure", ReferenceKind:= \_

wdOnlyLabelAndNumber, ReferenceItem:=thisNumber, InsertAsHyperlink:=True, \_

IncludePosition:=False, SeparateNumbers:=False, SeparatorString:=" "

End If

End Sub

**Sub InsertHeadingRef()**

**'** Paul Beverley - Version 18.02.18

' Adds a hyperlinked reference to a heading

myHeads = ActiveDocument.GetCrossReferenceItems(wdRefTypeHeading)

For i = 1 To UBound(myHeads) - 8

allHeads = allHeads & i & myHeads(i) & vbCr

Next i

allHeads = allHeads & vbCr & vbCr & "Reference number?"

thisNumber = InputBox(allHeads, "Add heading reference")

Selection.InsertCrossReference ReferenceType:="Heading", ReferenceKind:= \_

wdContentText, ReferenceItem:=thisNumber, InsertAsHyperlink:=True, \_

IncludePosition:=False, SeparateNumbers:=False, SeparatorString:=" "

End Sub

**Sub InsertTableRefs()**

**'** Paul Beverley - Version 21.02.18

' Adds hyperlinked references to all tables

Set rng = Selection.Range.Duplicate

Application.ScreenUpdating = False

numTables = ActiveDocument.Tables.count

st = -9: en = 0

Do

st = st + 10: en = en + 10

allTables = ""

For i = st To en

ActiveDocument.Tables(i).Select

Selection.Collapse wdCollapseStart

Selection.MoveUp , 1

Selection.Expand wdParagraph

If Len(Selection) > 5 Then allTables = allTables & Selection.Text

If i = numTables Then Exit For

Next i

allTables = allTables & vbCr & vbCr & "Reference number?"

thisNumber = InputBox(allTables, "Add table reference")

Loop Until en > numTables Or Val(thisNumber) > 0

Application.ScreenUpdating = True

rng.Select

If Val(thisNumber) = 0 Then

Beep

Else

Selection.InsertCrossReference ReferenceType:="table", ReferenceKind:= \_

wdOnlyLabelAndNumber, ReferenceItem:=thisNumber, InsertAsHyperlink:=True, \_

IncludePosition:=False, SeparateNumbers:=False, SeparatorString:=" "

End If

End Sub

**Sub ItalicOffWord()**

**'** Paul Beverley - Version 18.05.17

' Removes all italic and underline from selected text (or current word)

uLineOffToo = False

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If Len(Selection) > 1 Then

Selection.Range.Font.Italic = False

If uLineOffToo = True Then Selection.Range.Font.Underline = False

Else

Selection.Expand wdWord

Selection.Range.Font.Italic = False

If uLineOffToo = True Then Selection.Range.Font.Underline = False

Selection.Collapse wdCollapseEnd

End If

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub ListSurnameSwap()**

**'** Paul Beverley - Version 29.01.18

' Swaps forename first to surname first in selected text

colourSurname = True

startWord = 5

If Selection.Start = Selection.End Then Selection.WholeStory

Set rng = Selection.Range.Duplicate

For i = 1 To rng.Paragraphs.count

If Len(rng.Paragraphs(i)) > 2 Then

saarfais = Len(rng.Paragraphs(i))

srName = startWord

Do

gotsurname = True

srName = srName + 1

Set myPar = rng.Paragraphs(i)

mySurname = myPar.Range.Words(srName)

If Len(mySurname) = 1 Then gotsurname = False

If InStr(mySurname, ".") > 0 Then gotsurname = False

Loop Until gotsurname

mySurname = myPar.Range.Words(srName)

myPar.Range.Words(srName).Delete

myPar.Range.Words(startWord).InsertBefore Text:=Trim(mySurname) & ", "

If colourSurname Then myPar.Range.Words(startWord).Font.Color = wdColorBlue

End If

Next i

End Sub

**Sub LongParagraphCheck()**

**'** Paul Beverley - Version 25.03.20

' Colour long sentences

myCol1 = wdColorBlue

myCol2 = wdColorSeaGreen

minLen = 40

' nowTrack = ActiveDocument.TrackRevisions

' ActiveDocument.TrackRevisions = False

myCol = myCol1

For Each snt In ActiveDocument.Sentences

allWords = snt

wordNum = Len(allWords) - Len(Replace(allWords, " ", ""))

If wordNum > minLen Then

snt.Font.Color = myCol

' Swap between the two colours ready for next long sentence

If myCol = myCol1 Then

myCol = myCol2

Else

myCol = myCol1

End If

End If

Next snt

' ActiveDocument.TrackRevisions = nowTrack

End Sub

**Sub MakeOKwordsList()**

**'** Paul Beverley - Version 26.10.19

' Makes a list of all words not highlighted

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Set myList = ActiveDocument

For i = myList.Paragraphs.count To 1 Step -1

Set myPar = myList.Paragraphs(i).Range

myPar.End = myPar.End - 1

isHigh = myPar.HighlightColorIndex <> 0

If Len(myPar.Text) > 1 Then

col = myPar.Font.Color

Else

col = 0

End If

If isHigh Or col > 0 Then

myList.Paragraphs(i).Range.Delete

sdfgdsf = 0

End If

Next i

myList.Paragraphs(1) = "OKwords"

End Sub

**Sub MoveWordLeft()**

**'** Paul Beverley - Version 30.11.19

' Moves current word one word to the right

If Selection.Start = Selection.End Then

Selection.Expand wdWord

End If

gotRightSpace = (Right(Selection, 1) = " ")

If gotRightSpace Then

Selection.MoveEnd , -1

Selection.Cut

Selection.MoveEnd , 1

Else

Selection.Cut

Selection.MoveStart , -1

End If

Selection.Delete

Selection.MoveLeft wdWord, 1

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

If LCase(Selection) <> UCase(Selection) Then

Selection.Paste

Selection.TypeText Text:=" "

Else

Selection.TypeText Text:=" "

Selection.Paste

End If

Selection.MoveLeft , 1

Selection.Expand wdWord

End Sub

**Sub MoveWordRight()**

**'** Paul Beverley - Version 30.11.19

' Moves current word one word to the Right

If Selection.Start = Selection.End Then

Selection.Expand wdWord

End If

gotRightSpace = (Right(Selection, 1) = " ")

If gotRightSpace Then

Selection.MoveEnd , -1

Selection.Cut

Selection.MoveEnd , 1

Else

Selection.Cut

Selection.MoveStart , -1

End If

' myWord = Selection

Selection.Delete

Selection.Expand wdWord

Selection.Collapse wdCollapseEnd

If LCase(Selection) <> UCase(Selection) Then

Selection.Paste

Selection.TypeText Text:=" "

Else

Selection.TypeText Text:=" "

Selection.Paste

End If

Selection.MoveLeft , 1

Selection.Expand wdWord

End Sub

**Sub MultiFilePageCount()**

**'** Paul Beverley - Version 30.03.21

' Counts pages in a group of files

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 \_

And InStr(LCase(rng.Text), ".pdf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.count

Dialogs(wdDialogFileOpen).Show

If Documents.count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Or \_

InStr(LCase(myFile), ".pdf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Count ALL the files in this folder:" & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "Multifile Wordcount")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Count just the files listed here?", \_

vbQuestion + vbYesNoCancel, "Multifile Wordcount")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

For i = 1 To numFiles

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set myDoc = Application.Documents.Open(fileName:=thisFile)

StatusBar = allMyFiles(i)

Selection.EndKey Unit:=wdStory

myCount = Selection.Information(wdActiveEndAdjustedPageNumber)

myResult = myResult & myDoc.Name & vbTab & Trim(Str(myCount)) & vbCr

myResult = Replace(myResult, ".docx", "")

myResult = Replace(myResult, ".doc", "")

myResult = Replace(myResult, ".pdf", "")

myDoc.Close SaveChanges:=wdDoNotSaveChanges

DoEvents

Next i

Documents.Add

Selection.TypeText myResult

Selection.WholeStory

Set tbl = Selection.ConvertToTable(Separator:=wdSeparateByTabs)

tbl.Columns.AutoFit

Selection.HomeKey Unit:=wdStory

End Sub

**Sub NameToInitial()**

**'** Paul Beverley - Version 02.03.15

' Change given name to initial, then jump on to next

Selection.Words(1).Select

myInitial = Left(Selection, 1) & "."

lastChar = Right(Selection, 1)

If lastChar = " " Then

Selection.TypeText myInitial & " "

Else

Selection.TypeText myInitial

End If

Selection.MoveRight Unit:=wdWord, count:=1

Selection.Words(1).Select

asgfsd = Selection

If Selection = "and " Or UCase(Selection) = LCase(Selection) Then \_

Selection.MoveRight Unit:=wdWord, count:=1

Selection.Words(1).Select

If Selection = "and " Or UCase(Selection) = LCase(Selection) Then \_

Selection.MoveRight Unit:=wdWord, count:=1

Selection.Collapse wdCollapseStart

End Sub

**Sub NoteRelocater()**

**'** Paul Beverley - Version 01.02.18

' Takes footnotes out into the running text

totNotes = ActiveDocument.Footnotes.count

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "xxxx^p"

.Execute

End With

Selection.Collapse wdCollapseEnd

idx = 0

ntNum = 0

For Each myPar In ActiveDocument.Paragraphs

If Left(myPar.Range.Text, 4) = "xxxx" Then

ntNum = 0

Selection.Find.Execute

Selection.Collapse wdCollapseEnd

Selection.TypeText vbCr

Selection.Expand wdParagraph

Selection.Range.Style = wdStyleNormal

End If

If myPar.Range.Footnotes.count > 0 Then

' myPar.Range.Select

paraNTs = myPar.Range.Footnotes.count

For i = 1 To paraNTs

ntNum = ntNum + 1

idx = idx + 1

StatusBar = totNotes - idx

If idx > totNotes Then Exit For: Exit For

Set fn = ActiveDocument.Footnotes(idx)

Set cit = fn.Reference

cit.InsertBefore Text:="[" & ntNum & "]"

' Debug.Print fn.range.Text

fn.Range.Copy

Selection.TypeText "[" & ntNum & "] "

Selection.Paste

Selection.TypeText vbCr

Next i

End If

Next myPar

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\[([0-9]{1,})\]"

.Replacement.Text = "\1"

.Replacement.Font.Superscript = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Beep

End Sub

**Sub ParaSentenceLengthHighlighter()**

**'** Paul Beverley - Version 16.05.19

' Highlights paragraphs with too many sentences

For Each myPar In ActiveDocument.Paragraphs

If myPar.Range.Sentences.count > 3 Then \_

myPar.Range.HighlightColorIndex = wdYellow

If myPar.Range.Sentences.count > 5 Then \_

myPar.Range.HighlightColorIndex = wdBrightGreen

Next myPar

End Sub

**Sub ParaSplitJoin()**

**'** Paul Beverley - Version 04.12.24

' Splits the para after current sentence or joins to next para

addBlankLine = False

' addBlankLine = True

Set rng = Selection.Range.Duplicate

rng.Expand wdSentence

CR = vbCr

If addBlankLine = True Then CR = CR & CR

If Right(rng.Text, 1) = " " Then

rng.start = rng.End - 1

rng.Text = CR

Else

rng.start = rng.End - 2

If Left(rng.Text, 1) <> vbCr Then rng.MoveStart , 1

rng.Text = " "

End If

End Sub

**Sub PDFpager()**

**'** Paul Beverley - Version 14.05.12

' Highlights page numbers of text from a PDF

pageNumberStyle = wdStyleHeading1

textBefore = ">>>"

textAfter = "<<<"

numberAtBottom = False

endCode1 = " "

'endCode1 = "^p"

endCode2 = "^p"

Selection.Expand wdWord

findPage = Val(Selection) - 1

stepSize = 3

If numberAtBottom = True Then

codeBefore = "": codeAfter = Chr(12)

Else

codeBefore = Chr(12): codeAfter = ""

End If

If endCode1 = "^p" Then endCode1 = Chr(13)

If endCode2 = "^p" Then endCode2 = Chr(13)

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

rng.Start = Selection.Start - 1

If Asc(rng) <> Asc(Right(textBefore, 1)) Then

Selection.InsertBefore codeBefore & textBefore

Selection.InsertAfter textAfter & codeAfter

Selection.Expand wdParagraph

Selection.ParagraphFormat.LineSpacingRule = wdLineSpaceSingle

Selection.Style = ActiveDocument.Styles(pageNumberStyle)

End If

pagesMissing = ""

Do

hereNow = Selection.Start

rng.Start = 0

rng.End = hereNow

If findPage >= stepSize Then

myLimit = hereNow \* (findPage - stepSize) / findPage + 1

Else

myLimit = 1

End If

Do

thisIsIt = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Trim(Str(findPage))

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

If rng.Find.Found Then

rng.Select

rng2.Start = rng.Start - 1

rng2.Select

leftCode = Left(rng2, 1)

rng2.Start = rng.End

rng2.Select

rightCode = Left(rng2, 1)

If leftCode = endCode1 And rightCode = endCode2 Then thisIsIt = True

If leftCode = endCode2 And rightCode = endCode1 Then thisIsIt = True

End If

Loop Until thisIsIt Or rng.Start <= myLimit Or rng.Find.Found = False

If thisIsIt And rng.Start > myLimit Then

rng.Select

Selection.InsertBefore codeBefore & textBefore

Selection.InsertAfter textAfter & codeAfter

Selection.Expand wdParagraph

Selection.ParagraphFormat.LineSpacingRule = wdLineSpaceSingle

Selection.Style = ActiveDocument.Styles(pageNumberStyle)

Selection.MoveRight Unit:=wdCharacter, count:=1

Else

pagesMissing = pagesMissing & Trim(Str(findPage)) & ", "

Beep

' StatusBar = "Searching for: " & findPage - 1

End If

findPage = findPage - 1

Loop Until nowPage = 1 Or findPage = 0

Selection.HomeKey Unit:=wdStory

Selection.TypeText pagesMissing & vbCr & vbCr

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub AbstractAndKeywordCount()**

**'** Paul Beverley - Version 09.04.21

' Counts abstract word numbers and keyword items

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

numWds = rng.ComputeStatistics(wdStatisticWords)

If numWds = 0 Then

rng.MoveEnd , 1

numWds = rng.ComputeStatistics(wdStatisticWords)

End If

lenLess = Len(rng) - Len(Replace(rng, ",", ""))

If lenLess < 3 Then lenLess = Len(rng) - Len(Replace(rng, ";", ""))

numKeywords = lenLess + 1

MsgBox ("Words: " & Trim(Str(numWds)) & vbCr & vbCr & \_

"Keywords: " & Trim(Str(numKeywords)))

End Sub

**Sub AuthorSurnameSwap()**

**'** Paul Beverley - Version 19.05.21

' Moves author surname in a list to the beginning of the line

If Selection.Start = Selection.End Then

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.MoveLeft wdWord, 1, Extend:=wdExtend

Selection.Cut

Selection.MoveStart , -1

Selection.Delete

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Selection.Paste

Selection.TypeText Text:=", "

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Else

Selection.Cut

Selection.Expand wdParagraph

hasComma = (InStr(Selection, ",") > 0)

Selection.Collapse wdCollapseStart

Selection.Paste

If Right(Selection, 1) = " " Then Selection.MoveLeft , 1

Selection.TypeText Text:=", "

Selection.Expand wdParagraph

spPos = InStr(Selection, " ,")

If hasComma Then

If spPos > 0 Then Selection.MoveStart , spPos - 1

Selection.Collapse wdCollapseStart

Selection.MoveEnd , 1

Selection.Delete

End If

Selection.Expand wdParagraph

If Left(Right(Selection, 2), 1) = " " Then

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 2

Selection.MoveEnd , 1

Selection.Delete

Selection.MoveRight , 1

End If

Selection.Collapse wdCollapseEnd

End If

End Sub

**Sub CheckCapSentences()**

**'** Paul Beverley - Version 18.06.21

' Forces an initial capital on every sentence

For Each sn In ActiveDocument.Sentences

mySnt = sn

init = Left(mySnt, 1)

If UCase(init) <> init Then

sn.Characters(1).Text = UCase(init)

End If

DoEvents

Next sn

End Sub

**Sub CommentDeleteMoveNext()**

**'** Paul Beverley - Version 17.05.22

' Deletes current comment and moves to the next comment

Application.Run macroName:="RejectChangesSelected"

hereNow = Selection.Start

Selection.GoTo What:=wdGoToComment, Which:=wdGoToNext

If Selection.Start = hereNow Then Beep

End Sub

**Sub CommentJumpToText()**

**'** Paul Beverley - Version 22.06.22

' Moves the cursor from the comment back into the text

cursorAtEnd = True

cursorAtStart = False

Selection.Comments(1).Scope.Select

If cursorAtEnd = True Then Selection.Collapse wdCollapseEnd

If cursorAtStart = True Then Selection.Collapse wdCollapseStart

End Sub

**Sub CopySelectedForCompare()**

**'** Paul Beverley - Version 23.10.21

' Copies the selected text to a specified file

myFile = "aaTempFile.docx"

Set mainFile = ActiveDocument

If Selection.Start = Selection.End Then

Beep

myResponse = MsgBox("Work with me! Select some text first. :-)", \_

, "CopySelectedForCompare")

Exit Sub

End If

Selection.Copy

Documents.Open fileName:=myFile

Selection.WholeStory

Selection.Delete

Selection.Paste

mainFile.Activate

Beep

End Sub

**Sub DeleteBackToSentenceStart()**

**'** Paul Beverley - Version 23.04.21

' Selects from current word to end of sentence

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

Set rng = Selection.Range.Duplicate

rng.End = rng.End + 1

rng.Text = UCase(rng.Text)

rng.Expand wdSentence

Selection.Start = rng.Start

Selection.Delete

End Sub

**Sub EquationsAllItalic()**

**'** Paul Beverley - Version 02.11.20

' Makes all equations in the current selection italic

For Each myMath In Selection.Range.OMaths

myMath.Range.Italic = True

DoEvents

Next myMath

End Sub

**Sub ExceptionsListEdit()**

**'** Paul Beverley - Version 27.05.20

' Allows the user to edit, add to or replace the exceptions list

CR = vbCr: CR2 = CR & CR

allText = Selection.Text

If Selection.Start = Selection.End Then

Selection.TypeText Text:=pbExceptionsList

Else

myInput = InputBox("1 = Add to exceptions list" & CR & \_

"2 = Create new exceptions list", "ExceptionsListEdit", "1")

myNumber = Val(myInput)

sdfsgf = Replace(pbExceptionsList, vbCr, " ")

Select Case myNumber

Case 1: pbExceptionsList = pbExceptionsList & allText

Case 2: pbExceptionsList = allText

Case Else: Beep: Exit Sub

End Select

If Left(pbExceptionsList, 1) <> CR Then \_

pbExceptionsList = CR & pbExceptionsList

If Right(pbExceptionsList, 1) <> CR Then \_

pbExceptionsList = pbExceptionsList & CR

End If

End Sub

**Sub FootnoteTidy()**

**'** Paul Beverley - Version 30.07.21

' Checks each footnote

For Each ft In ActiveDocument.Footnotes

fnText = ft.Range.Text

If Left(fnText, 1) <> " " Then

ft.Range.InsertBefore " "

End If

Next ft

End Sub

**Sub FormatText()**

**'** Paul Beverley - Version 15.10.21

' Formats a word/selection of whole words

myFont = "Arial"

' myFont = ""

mySize = 14

' mySize = 0

myItalic = True

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

If myFont <> "" Then Selection.Font.Name = myFont

If mySize > 0 Then Selection.Font.Size = mySize

If myItalic = True Then Selection.Font.Italic = True

Selection.Collapse wdCollapseEnd

End Sub

**Sub FunnyDashToEnDash()**

**'** Paul Beverley - Version 06.08.21

' Changes all funny dashes to proper en dash

doHighlight = True

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8210)

.Wrap = wdFindContinue

.Forward = True

If doHighlight = True Then .Replacement.Highlight = True

.Replacement.Text = ChrW(8211)

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Beep

End Sub

**Sub GermanAccusative()**

**'** Paul Beverley - Version 18.06.21

' Forces the accusative -en ending to the current word

Selection.Expand wdWord

If Right(Selection, 1) = " " Then Selection.MoveEnd , -1

Selection.Collapse wdCollapseEnd

Selection.MoveStart , -1

If Selection <> "e" Then Selection.MoveStart , -1

If Left(Selection, 1) <> "e" Then Selection.Collapse wdCollapseEnd

Selection.TypeText Text:="en"

End Sub

**Sub HighlightAllTrackedText()**

**'** Paul Beverley - Version 15.04.22

' Highlights (and/or font colours) all tracked text

myColour = wdBrightGreen

myColour = wdYellow

'myColour = wdNoHighlight

myFcolour = wdColorBlack

'myFcolour = wdColorBlue

ActiveDocument.TrackRevisions = False

For Each rev In ActiveDocument.Range.Revisions

Set rng = rev.Range

rng.Start = rng.Start - 1

rng.End = rng.End + 1

If myColour > 0 Then rng.HighlightColorIndex = myColour

If myFcolour > 0 Then rng.Font.Color = myFcolour

DoEvents

i = i + 1

If i Mod 20 = 1 Then rng.Select

Next rev

If ActiveDocument.Footnotes.count > 0 Then

For Each rev In ActiveDocument.StoryRanges(wdFootnotesStory).Revisions

Set rng = rev.Range

rng.Start = rng.Start - 1

rng.End = rng.End + 1

If myColour > 0 Then rng.HighlightColorIndex = myColour

If myFcolour > 0 Then rng.Font.Color = myFcolour

DoEvents

i = i + 1

If i Mod 20 = 1 Then rng.Select

Next rev

End If

If ActiveDocument.Endnotes.count > 0 Then

For Each rev In ActiveDocument.StoryRanges(wdEndnotesStory).Revisions

Set rng = rev.Range

rng.Start = rng.Start - 1

rng.End = rng.End + 1

If myColour > 0 Then rng.HighlightColorIndex = myColour

If myFcolour > 0 Then rng.Font.Color = myFcolour

DoEvents

i = i + 1

If i Mod 20 = 1 Then rng.Select

Next rev

End If

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub HighlightOddQuotations()**

**'** Paul Beverley - Version 25.05.20

' Marks all quotes + displayed text

' Do you want displayed text marked?

markDisplayed = True

' Add coding to existing displayed quotes?

addCodes = True

addCodes = False

preCode = "<DIS>"

Postcode = "</DIS>"

codeOnNextLine = False

' Minimum length of quotes (words)

minLength = 3

' Minimum indent of quotes (cm)

minIndent = 1.05

' Colour the font of quotations

colourFont = True

colourFont = False

myColour = wdColorBlue

' Colour of the possible plural possessive problems

possessiveColour = wdColorRed

' Possible possessive errors only shown if within

' this many paragraphs

maxParasPossessive = 5

' Add a highlight

highlightText = True

myHighlight = wdYellow

' Strike it through

strSingle = False

strDouble = False

' What kind of quote?

doDoubleQuotes = True

doSingleQuotes = True

' Do you want the notes checked (if there are any)?

doFootnotes = True

doEndnotes = True

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

nowColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myHighlight

soCalledWords = "---,.;:!?…" & ChrW(8220) & ChrW(8221)

' Clear old codes, in case you're running it a second time

If addCodes = True Then

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = preCode

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Postcode

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

If markDisplayed = True Then

For Each myPara In ActiveDocument.Paragraphs

If myPara.Range.ParagraphFormat.LeftIndent > \_

CentimetersToPoints(minIndent) Then

myPara.Range.Font.Emboss = True

If addCodes = True Then

myPara.Range.InsertBefore Text:=preCode

myPara.Range.InsertAfter Text:=Postcode

End If

End If

Next

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = preCode & Postcode

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

If codeOnNextLine = False Then

With rng.Find

.Text = "^p" & Postcode

.Replacement.Text = Postcode & "^p"

.Execute Replace:=wdReplaceAll

End With

End If

End If

gottaFunny = False

funnyCount = 0

For i = 1 To 3

If i = 1 And (ActiveDocument.Footnotes.count = 0 Or \_

doFootnotes = False) Then i = 2

If i = 2 And (ActiveDocument.Endnotes.count = 0 Or \_

doEndnotes = False) Then i = 3

Select Case i

Case 1: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

' Single quotes

If doSingleQuotes = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8216) & "\*" & ChrW(8217)

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Emboss = True

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

' Check for any apostrophe-letter that is half struck-through

If doSingleQuotes = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8217) & "^$"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

' If this is half struckthrough

If rng.Font.Emboss > 999 Then

' Extend to the following close single quote

rng.MoveEndUntil cset:=ChrW(8217), count:=wdForward

rng.MoveEnd wdCharacter, 1

rng.Font.Emboss = True

rng.Start = rng.End - 1

rng.Collapse wdCollapseStart

Else

rng.Collapse wdCollapseEnd

End If

rng.Find.Execute

DoEvents

Loop

' Now check for plural possessives: s'

Select Case i

Case 1: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

' Add a dummy open quote at the end of the text

rng.InsertAfter ChrW(8216)

theVeryEnd = rng.End

' Check from each of the s-apostrophes to the next open quote

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "s" & ChrW(8217) & "\*" & ChrW(8216)

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.End = rng.Start + 1

isQuote = rng.Font.Emboss

rng.End = endNow

rng.MoveStart wdCharacter, 2

endQuoteWas = rng.Start

myText = rng.Text

If InStr(myText, ChrW(8217)) > 0 And isQuote = True Then

' How many apostrophe/close quotes are there?

aposNum = Len(myText) - Len(Replace(myText, ChrW(8217), ""))

myPointer = endQuoteWas

endQuoteNow = myPointer

For j = 1 To aposNum

aposPos = InStr(myText, ChrW(8217))

nextChar = Mid(myText, aposPos + 1, 1)

myPointer = myPointer + aposPos

' Is nextChar non-alpha? If so, this is the \*real\* end of the quote

If LCase(nextChar) = UCase(nextChar) Then

endQuoteNow = myPointer

End If

myText = Mid(myText, aposPos + 1)

Next

If endQuoteNow > endQuoteWas Then

endNow = rng.End

rng.End = endQuoteNow

rng.Font.Emboss = True

myChar = Right(rng.Text, 2)

thisLot = rng.Text

numParas = Len(thisLot) - Len(Replace(thisLot, Chr(13), ""))

If Asc(myChar) = Asc("s") And numParas < maxParasPossessive Then

gottaFunny = True

rng.Font.Emboss = False

rng.Font.Color = possessiveColour

funnyCount = funnyCount + 1

End If

End If

rng.End = myPointer

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

rng.Start = theVeryEnd - 2

rng.End = theVeryEnd - 1

rng.Delete

DoEvents

End If

' Double quotes

If doDoubleQuotes = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8220) & "\*" & ChrW(8221)

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Emboss = True

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End If

' Remove embossing from short quotes

If minLength > 0 Then

Select Case i

Case 1: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Emboss = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

' Find the true word count

myText = rng.Text

wdsOff = 1

For j = 1 To Len(myText)

If InStr(soCalledWords, Mid(myText, j, 1)) > 0 Then \_

wdsOff = wdsOff + 1

Next j

If j > 3 Then

' To correct Word's weird wordcount'hello' = count OK

' 'hello?' error by one word, and 'hello??' = count OK

charOne = Mid(myText, j - 3, 1)

charTwo = Mid(myText, j - 2, 1)

oneIsAlpha = (LCase(charOne) <> UCase(charOne))

twoIsNotAlpha = (LCase(charTwo) = UCase(charTwo))

If oneIsAlpha And twoIsNotAlpha Then wdsOff = wdsOff + 1

If rng.Words.count - wdsOff < minLength - 1 Then

rng.Font.Emboss = False

rng.Collapse wdCollapseEnd

End If

End If

rng.Find.Execute

DoEvents

Loop

End If

' Apply whatever effect is wanted on the embossed text

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Font.Emboss = True

.Text = ""

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Font.Emboss = False

If highlightText = True Then .Replacement.Highlight = True

If strSingle = True Then .Replacement.Font.StrikeThrough = True

If strDouble = True Then .Replacement.Font.DoubleStrikeThrough = True

If colourFont = True Then .Replacement.Font.Color = myColour

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next i

' Warn the user of possible plural possessives

If gottaFunny = True Then

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Font.Color = possessiveColour

.MatchWildcards = False

.Execute

End With

Selection.End = Selection.Start - 1

Selection.Find.Execute

MsgBox "Please check possible plural possessives (" \_

& Trim(Str(funnyCount)) & ")"

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8220) & "[!^13]" & ChrW(8221)

.Highlight = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = False

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Options.DefaultHighlightColorIndex = nowColour

ActiveDocument.TrackRevisions = myTrack

Beep

End Sub

**Sub HighlightToStyles()**

**'** Paul Beverley - Version 03.08.21

' Finds highlighted text and does things to each colour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

For Each ch In rng.Characters

Select Case ch.HighlightColorIndex

Case wdBrightGreen

ch.Bold = True

Case wdYellow

ch.Italic = True

Case wdTurquoise

ch.Style = ActiveDocument.Styles("HTML sample")

DoEvents

End Select

Next ch

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Beep

End Sub

**Sub InitialCapAllCapsWords()**

**'** Paul Beverley - Version 03.06.21

' Initial caps the all-caps words in a document

For Each myWord In ActiveDocument.Words

wd = myWord.Text

If Len(wd) > 2 And UCase(wd) = wd And \_

LCase(wd) <> wd Then

newWord = Left(wd, 1) & LCase(Mid(wd, 2))

If newWord <> wd Then myWord.Text = newWord

Debug.Print myWord.Text & "|" & newWord & "|"

End If

DoEvents

Next myWord

End Sub

**Sub ItalicSpeechToSingleQuotes()**

**'** Paul Beverley - Version 04.05.22

' Finds italic text, Romanises it and adds single quotes

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindStop

.Font.Italic = True

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.InsertBefore Text:=ChrW(8216)

rng.Font.Italic = False

If Right(rng.Text, 1) = vbCr Then rng.MoveEnd , -1

rng.InsertAfter Text:=ChrW(8217)

i = i + 1

If i Mod 10 = 0 Then rng.Select

rng.Start = endNow + 1

rng.End = endNow + 1

rng.Find.Execute

DoEvents

Loop

End Sub

**Sub LanguageToggle()**

**'** Paul Beverley - Version 01.11.21

' Toggles the language setting of (part selected) text

myLanguage1 = wdEnglishUK

addItalic1 = False

myLanguage2 = wdFrench

addItalic2 = True

myPriority = myLanguage2

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

nowLanguage = Selection.LanguageID

Debug.Print nowLanguage

Select Case nowLanguage

Case myLanguage1: Selection.LanguageID = myLanguage2

If addItalic2 = True Then Selection.Font.Italic = True

Case myLanguage2: Selection.LanguageID = myLanguage1

If addItalic1 = True Then Selection.Font.Italic = True

Case Else: Selection.LanguageID = myPriority

End Select

End Sub

**Sub LongParagraphHighlighterChars()**

**'** Paul Beverley - Version 23.06.21

' Highlights all paragraphs more than a certain length

charsLength = 40

myCharsColour = wdBrightGreen

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

For Each myPara In ActiveDocument.Paragraphs

If myPara.Range.Characters.count > charsLength Then

myPara.Range.HighlightColorIndex = myCharsColour

myPara.Range.Select

End If

DoEvents

Next

ActiveDocument.TrackRevisions = nowTrack

Beep

End Sub

**Sub LongQuoteNext()**

**'** Paul Beverley - Version 01.08.20

' Jumps to the next long quotation

minWords = 40

ignoreSapostrophe = True

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

If InStr(ChrW(8216) & ChrW(8220), Left(rng, 1)) > 0 Then \_

rng.Start = rng.Start + 1

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & ChrW(8216) & ChrW(8220) & "]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

DoEvents

End With

Do While rng.Find.Found = True

mySingle = (AscW(rng) = 8216)

rng.End = ActiveDocument.Content.End

Set rng2 = rng.Words(minWords + 1)

rng2.Start = rng.Start

myTest = rng2

If mySingle = True Then

myEnd = ChrW(8217)

myTest = Replace(myTest, myEnd & "s", "")

If ignoreSapostrophe = True Then \_

myTest = Replace(myTest, "s" & myEnd, "")

myTest = Replace(myTest, myEnd & "t", "")

myTest = Replace(myTest, myEnd & "v", "")

myTest = Replace(myTest, myEnd & "d", "")

myTest = Replace(myTest, myEnd & "l", "")

myTest = Replace(myTest, myEnd & "r", "")

Debug.Print myTest

Else

myEnd = ChrW(8221)

End If

If InStr(myTest, myEnd) = 0 Then

rng.End = ActiveDocument.Content.End

myTest = Replace(rng.Text, myEnd & "s", "xx")

If ignoreSapostrophe = True Then \_

myTest = Replace(myTest, "s" & myEnd, "xx")

myTest = Replace(myTest, myEnd & "t", "xx")

myTest = Replace(myTest, myEnd & "v", "xx")

myTest = Replace(myTest, myEnd & "d", "xx")

myTest = Replace(myTest, myEnd & "l", "xx")

myTest = Replace(myTest, myEnd & "r", "xx")

Debug.Print Left(myTest, 40)

myEndQuote = InStr(myTest, myEnd)

rng2.End = rng2.Start + myEndQuote

rng2.Select

Exit Sub

End If

rng.Start = rng.Start + 1

rng.Find.Execute

Loop

Beep

End Sub

**Sub MenuItemMaker()**

**'** Paul Beverley - Version 13.04.20

' Creates a histogram of paragraph length

hashstart = InStr(ActiveDocument.Content, "#####")

hashEnd = InStr(ActiveDocument.Content, "#####" & vbCr)

CR = vbCr: CR2 = CR & CR

If Selection.Start < hashstart Then

Set rng = ActiveDocument.Range(0, hashstart - 2)

m = rng.Text

Debug.Print m

m = Replace(m, "a = a & """, "")

m = Replace(m, "a = """"" & CR, "")

m = Replace(m, """" & CR, "" & vbCr)

m = Replace(m, "|", "")

Selection.Start = hashEnd + 6

Selection.Collapse wdCollapseStart

Selection.TypeText Text:=m

Else

Set rng = ActiveDocument.Range(hashEnd + 5, \_

ActiveDocument.Content.End)

m = rng.Text

Debug.Print m & CR2

m = Replace(m, CR, "|""" & CR & "a = a & """)

Debug.Print m

m = Mid(m, 3, Len(m) - 23)

m = "a = """"" & m

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:=m & CR2

End If

End Sub

**Sub MultifileXML()**

**'** Paul Beverley - Version 01.09.21

' Save a folder full of files as XMLs

showTrackChanges = True

XMLFilesInFolder = True

XMLFolderName = "XML"

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "Multifile Text Collection")

docCount = Documents.count

Dialogs(wdDialogFileOpen).Show

If Documents.count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

If XMLFilesInFolder = True Then

mySubFolder = "\" & XMLFolderName

Else

mySubFolder = ""

End If

myResponse = MsgBox("Save XMLs of ALL the files in this folder: " & mySubFolder & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "Multifile XML Writer")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

If XMLFilesInFolder = True Then

myResponse = MsgBox("Save XMLs of just the files listed here, in this folder: " & \_

mySubFolder & myFolder & " ?", \_

vbQuestion + vbYesNoCancel, "Multifile XML Writer")

If myResponse <> vbYes Then Exit Sub

End If

For i = 1 To numFiles

thisName = allMyFiles(i)

myFileType = Mid(thisName, InStr(thisName, "."))

justName = Replace(thisName, myFileType, "")

thisFile = myFolder & myDelimiter & thisName

Set myDoc = Application.Documents.Open(fileName:=thisFile)

StatusBar = allMyFiles(i)

If XMLFilesInFolder = True Then

justName = XMLFolderName & "\" & justName & ".xml"

End If

DoEvents

ActiveDocument.SaveAs2 fileName:=justName, FileFormat:= \_

wdFormatFlatXML, LockComments:=False, Password:="", Addtorecentfiles:= \_

True, WritePassword:="", ReadOnlyRecommended:=False, EmbedTrueTypeFonts:= \_

False, SaveNativePictureFormat:=False, SaveFormsData:=False, \_

SaveAsAOCELetter:=False, CompatibilityMode:=12

myDoc.Close SaveChanges:=wdDoNotSaveChanges

Next

Selection.HomeKey Unit:=wdStory

MsgBox (numFiles & " files saved as XML")

End Sub

**Sub NamesReverse()**

**'** Paul Beverley - Version 13.06.22

' Switches two names and adds (or removes) the comma

Selection.Expand wdWord

Selection.MoveEnd wdWord, 1

If InStr(Selection, ",") = 0 Then

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Cut

Selection.MoveRight wdWord, 1

Selection.MoveStart , -1

ch = Asc(Selection)

If ch = 32 Then

Selection.TypeText Text:=", "

Selection.Paste

Else

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=", "

Selection.Paste

Selection.MoveStart , -1

Selection.Delete

End If

Else

' Swap and remove comma

Selection.Cut

Selection.MoveEnd wdWord, 1

If Right(Selection, 1) = " " Then

Selection.Collapse wdCollapseEnd

Selection.Paste

Selection.MoveLeft , 1

Selection.MoveStart , -1

Selection.Delete

Else

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=" "

Selection.Paste

Selection.MoveStart , -2

Selection.Delete

End If

End If

End Sub

**Sub NavPaneLoad()**

**'** Paul Beverley - Version 10.04.21

' Copies selected text into the F&R box

Application.CommandBars ("Navigation")

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

myText = Selection

myText = Replace(myText, vbCr, "^p")

myText = Replace(myText, vbTab, "^t")

If Asc(myText) <> 32 Then myText = Trim(myText)

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myText

.Forward = True

.Replacement.Text = myText

.MatchWildcards = False

.MatchWholeWord = False

.MatchCase = True

End With

Selection.End = Selection.Start

On Error Resume Next

Application.Run MacroName:="EditReplace"

End Sub

**Sub PageCountBySection()**

**'** Paul Beverley - Version 03.04.21

' Counts the pages between section headings

myStyle = "Heading 1"

Set testDoc = ActiveDocument

Documents.Add

Set rng = testDoc.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Style = ActiveDocument.Styles(myStyle)

.Format = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

pgWas = 1

myChapter = ""

myNewChapter = Replace(rng.Text, vbCr, "")

Do While rng.Find.Found = True

myNewChapter = Replace(rng.Text, vbCr, "")

pgNow = rng.Information(wdActiveEndPageNumber)

rng.Collapse wdCollapseEnd

numPages = pgNow - pgWas

Debug.Print Left(myChapter, 15), pgWas, pgNow

If myChapter > "" Then

myChapter = myChapter & vbTab & Trim(Str(numPages)) & vbCr

Selection.TypeText Text:=myChapter

End If

pgWas = pgNow

myChapter = myNewChapter

DoEvents

rng.Find.Execute

Loop

rng.Collapse wdCollapseEnd

pgNow = rng.Information(wdActiveEndPageNumber)

numPages = pgNow - pgWas + 2

myChapter = myChapter & vbTab & Trim(Str(numPages)) & vbCr

Selection.TypeText Text:=myChapter

pgWas = pgNow

DoEvents

End Sub

**Sub ParaDelete()**

**'** Paul Beverley - Version 29.10.21

' Deletes the current paragraph

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Delete

End Sub

**Sub PDFrunningHeadHighlight()**

**'** Paul Beverley - Version 09.04.21

' Locates running heads and highlights/enlarges page number

myFontSize = 40

' myColour = wdYellow

myColour = 0

'myFontColour = 0

myFontColour = wdColorRed

minLength = 8

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13[A-Z0-9. \-\(\)" & ChrW(8211) & \_

ChrW(8212) & "]{5,}^13"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

If LCase(rng) <> rng And Len(rng) > minLength Then

i = 1

Do

i = i + 1

txt = Left(rng, i)

Loop Until LCase(txt) <> txt

i = i - 1

Set numRng = ActiveDocument.Range(rng.Start + 1, rng.Start + i)

If Len(numRng) > 1 Then

numRng.Font.Size = myFontSize

If myColour > 0 Then numRng.HighlightColorIndex = myColour

If myFontColour > 0 Then numRng.Font.Color = myFontColour

End If

i = 1

Do

i = i + 1

txt = Right(rng, i)

Loop Until LCase(txt) <> txt

i = i - 1

Set numRng = ActiveDocument.Range(rng.End - i, rng.End - 1)

If Len(numRng) > 1 Then

numRng.Font.Size = myFontSize

If myColour > 0 Then numRng.HighlightColorIndex = myColour

If myFontColour > 0 Then numRng.Font.Color = myFontColour

End If

j = j + 1

If j Mod 20 = 0 Then

rng.Select

Selection.Collapse wdCollapseEnd

End If

End If

rng.Start = endNow

DoEvents

rng.Find.Execute

Loop

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub PhrasesInSentences()**

**'** Paul Beverley - Version 25.10.20

' Lists all sentences containing a give phrase

myPhrase = Trim(Selection)

CR2 = vbCr & vbCr

If Len(myPhrase) < 5 Then

myResponse = MsgBox("Find phrase:" & CR2 & myPhrase & CR2 & "Sure?", vbQuestion \_

+ vbYesNoCancel, "PhrasesInSentences")

If myResponse <> vbYes Then Exit Sub

End If

Set rng = ActiveDocument.Content

Documents.Add

num = 0

gogo = False

For i = 1 To rng.Sentences.count

mySent = rng.Sentences(i).Text

If InStr(mySent, myPhrase) > 0 Then

num = num + 1

Selection.TypeText Text:=mySent & CR2

End If

DoEvents

If num > 49 And gogo = False Then

myResponse = MsgBox("Found 50 sentences!" & CR2 & "Continue?", vbQuestion \_

+ vbYesNoCancel, "PhrasesInSentences")

If myResponse <> vbYes Then Exit Sub

gogo = True

End If

Next i

MsgBox ("Found: " & Str(num) & " times")

End Sub

**Sub PluraliseDE()**

**'** Paul Beverley - Version 23.05.21

' Toggles the ending plural/singular

Selection.Expand wdWord

Selection.Start = Selection.End - 4

myText = Selection.Text

If InStr(myText, "em") > 0 Then

Selection.Text = Replace(myText, "em", "en")

Else

Selection.Text = Replace(myText, "en", "em")

End If

If Selection.Text = myText Then Beep

Selection.Collapse wdCollapseStart

End Sub

**Sub QuotationMarkErrorFinder()**

**'** Paul Beverley - Version 28.05.20

' Moves to the next mismatched double quotes

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = """"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

i = i + 1: If i Mod 30 = 0 Then rng.Select

If rng.Text = ChrW(8220) Then

' An open double quote

rngStart = rng.Start

DoEvents

rng.MoveEndUntil cset:=ChrW(8221) & ChrW(8220) & Chr(13)

rng.Collapse wdCollapseEnd

rng.End = rng.Start + 1

DoEvents

If rng.Text <> ChrW(8221) Then

rng.Start = rngStart

rng.Select

Beep

Exit Sub

End If

Else

' A close double quote

DoEvents

rng.MoveStartUntil cset:=ChrW(8221) & ChrW(8220) & Chr(13), count:=wdBackward

DoEvents

rng.Select

Beep

Exit Sub

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Beep

rng.Select

End Sub

**Sub RedTagger()**

**'** Paul Beverley - Version 28.10.20

' Colours all tags red

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\<[a-zA-Z]@\>"

' Or square bracket version:

' .Text = "\[[a-zA-Z]@\]"

.Replacement.Text = ""

.Replacement.Font.Color = wdColorRed

.Forward = True

.Wrap = wdFindContinue

.Format = True

.MatchCase = False

.MatchWholeWord = False

.MatchAllWordForms = False

.MatchSoundsLike = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub ReviewPaneToggle()**

**'** Paul Beverley - Version 03.03.22

' Opens and closes the track change review pane

useVertical = True

If ActiveDocument.ActiveWindow.Panes.count > 1 Then

Application.Run macroName:="Reviewingpane"

Else

If useVertical = True Then

Application.Run macroName:="Reviewingpanevertical"

Else

Application.Run macroName:="Reviewingpanehorizontal"

End If

End If

End Sub

**Sub SectionNumberChecker()**

**'** Paul Beverley - Version 16.02.11

' Check the sequence of section numbers

maxTabPosition = 15

marker = Chr(9) ' tab character

marker = " "

Selection.Expand wdParagraph

myText = Selection

tabPos = InStr(myText, marker) - 1

wasNum = Left(myText, tabPos)

' this should now hold just the heading number, e.g. F2.4.5

Selection.Start = Selection.End

Dim posNow(4) As Long

For i = 0 To 3

posNow(i) = i

Next i

Do

Do

i = Int(4 \* Rnd())

posNow(i) = Selection.Start

myJump = 0

If posNow(0) = posNow(1) Then i = -1

If posNow(0) = posNow(2) Then i = -1

If posNow(0) = posNow(3) Then i = -1

If posNow(1) = posNow(2) Then i = -1

If posNow(1) = posNow(3) Then i = -1

If posNow(2) = posNow(3) Then i = -1

If i = -1 Then Selection.MoveRight Unit:=wdCell, count:=3

Selection.Expand wdParagraph

myText = Selection

Selection.Start = Selection.End

If Selection.Start > ActiveDocument.Range.End - 3 Then

Selection.HomeKey Unit:=wdStory

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

End If

Loop Until Len(myText) > 3

tabPos = InStr(myText, Chr(9)) - 1

If tabPos > 2 And tabPos < maxTabPosition Then

nowNum = Left(myText, tabPos)

' this should now hold just the heading number, e.g. F2.4.5

nowLen = Len(nowNum)

wasLen = Len(wasNum)

If nowLen >= wasLen Then

longer = True

i = wasLen + 1

Else

longer = False

i = nowLen + 1

End If

Do

i = i - 1

commonBit = Left(nowNum, i)

Loop Until commonBit = Left(wasNum, i)

wasPart = Replace(wasNum, commonBit, "")

nowPart = Replace(nowNum, commonBit, "")

gotanError = True

If longer = True Then

If Right(commonBit, 1) = "." Then

If Val(nowPart) - Val(wasPart) = 1 Then gotanError = False

Else

If wasPart = "" And nowPart = ".1" Then gotanError = False

End If

Else

wasPart = Left(wasPart, InStr(wasPart, ".") - 1)

If Val(nowPart) - Val(wasPart) = 1 Then gotanError = False

End If

If Right(commonBit, 1) = "." Or nowLen = wasLen Then

If Val(nowPart) - Val(wasPart) = 1 Then gotanError = False

End If

wasNum = nowNum

End If

Loop Until (Selection.Start > ActiveDocument.Range.End - 3) Or gotanError = True

If gotanError = True Then

Selection.MoveLeft Unit:=wdCharacter, count:=2

Selection.Expand wdParagraph

Beep

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^t"

.Forward = True

.Replacement.Text = " "

.MatchWildcards = False

End With

End If

End Sub

**Sub SelectToSentenceEnd()**

**'** Paul Beverley - Version 23.04.21

' Selects from current word to end of sentence

Selection.Expand wdWord

Set rng = Selection.Range.Duplicate

rng.Expand wdSentence

If InStr(" " & vbCr, Right(rng.Text, 1)) > 0 Then

Selection.End = rng.End - 1

Else

Selection.End = rng.End

End If

If InStr(" " & vbCr, Right(Selection.Text, 1)) > 0 Then

Selection.MoveEnd , -1

End If

' Selection.Delete

End Sub

**Sub ShapeShowHide()**

**'** Paul Beverley - Version 29.08.17

' Makes all shapes distinguishable

useVariagated = False

doFill = True

doGlow = True

'On Error GoTo myReport

numTB = ActiveDocument.Shapes.count

Set myPlace = Selection.Range.Duplicate

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If myPlace.Start = 0 And myPlace.End = ActiveDocument.Content.End Then

myPlace.End = 0

' Remove shape showing

For i = 1 To numTB

thisType = ActiveDocument.Shapes(i).Type

If ActiveDocument.Shapes(i).TextFrame.HasText Then

ActiveDocument.Shapes(i).Select

Selection.Range.Font.Color = wdColorAutomatic

Else

If thisType <> 13 And doGlow = True Then

ActiveDocument.Shapes(i).Glow.Radius = 0

End If

ActiveDocument.Shapes(i).Fill.Visible = False

End If

Next i

Else

' Add shape showing

For i = 1 To numTB

thisType = ActiveDocument.Shapes(i).Type

' If thisType = 9 Then

' ActiveDocument.Shapes(i).Select

' Selection.Collapse wdCollapseStart

' Exit Sub

' End If

If ActiveDocument.Shapes(i).TextFrame.HasText Then

ActiveDocument.Shapes(i).Select

x = 1

If useVariagated = True Then

For Each ch In Selection.Characters

If x = 1 Then

ch.Font.Color = wdColorBlue

ch.Font.Bold = True

ch.Font.Size = 16

x = 0

Else

ch.Font.Color = wdColorRed

x = 1

End If

Next ch

Else

Selection.Font.Color = wdColorRed

Selection.Font.Bold = True

Selection.Font.Size = 12

End If

Else

If thisType <> 9 And thisType <> 13 And doFill = True Then

With ActiveDocument.Shapes(i).Fill

.ForeColor.RGB = RGB(55, 55, 200)

.Visible = True

.Solid

End With

End If

If thisType = 9 And doGlow = True Then

myAutoType = ActiveDocument.Shapes(i).AutoShapeType

If myAutoType <> -2 Then

ActiveDocument.Shapes(i).Glow.Radius = 2

End If

End If

End If

Next i

End If

ActiveDocument.TrackRevisions = myTrack

myPlace.Select

Beep

Exit Sub

myReport:

On Error GoTo 0

ActiveDocument.Shapes(i).Select

Selection.Collapse wdCollapseEnd

End Sub

**Sub SigloSmallCaps()**

**'** Paul Beverley - Version 24.02.22

' Finds "siglo" and small-caps the IVX following

doFootnotes = Selection.Information(wdInFootnote)

doEndnotes = Selection.Information(wdInEndnote)

Set rng = ActiveDocument.Content

If doFootnotes Then Set rng = \_

ActiveDocument.StoryRanges(wdFootnotesStory)

If doEndnotes Then Set rng = \_

ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[Ss]igl[os ]{2,3}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

' move to the following word

rng.MoveStart wdWord, 1

rng.MoveEnd wdWord, 1

' Check that the word only contains letters V, I and X

myText = Replace(rng.Text, "V", "")

myText = Replace(myText, "I", "")

myText = Replace(myText, "X", "")

If UCase(myText) = LCase(myText) Then

rng.Text = LCase(rng)

rng.Font.SmallCaps = True

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

rng.Select

Beep

End Sub

**Sub SmallCapsToProperNoun()**

**'** Paul Beverley - Version 11.02.18

' Changes every small caps word into initial cap + lowercase

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.SmallCaps = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

' If you want to count them...

myCount = myCount + 1

rng.Collapse wdCollapseStart

rng.Expand wdWord

newWord = rng.Text

rng.Font.SmallCaps = False

newWord = LCase(newWord)

newWord = UCase(Left(newWord, 1)) & Mid(newWord, 2)

rng.Text = newWord

rng.Expand wdWord

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

MsgBox "Changed: " & myCount

End Sub

**Sub SpacesInThousands()**

**'** Paul Beverley - Version 29.11.16

' Changes four-figure numbers from comma to space.

' Start from the top of the document

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^#,^#^#^#"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While Selection.Find.Found = True

' Note where the end of the found item is

endNow = Selection.End

' Make your required change

oldText = Selection.Text

newText = Replace(oldText, ",", " ")

' or if you prefer a fixed space?

' newText = Replace(oldText, ",", ChrW(160))

' Type in the replacement text

Selection.TypeText newText

' Be sure you're past the previous occurrence

Selection.End = endNow

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

End Sub

**Sub SpeciesAbbreviator()**

**'** Paul Beverley - Version 12.03.22

' Adds genus and species to the text, pulling data off a file

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Font.Italic = True

.Text = "[A-Z][a-z]{1,}"

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = True

.Execute

End With

myGenusList = "!"

Do While rng.Find.Found = True

rng.Select

myGenus = rng.Text

If InStr(myGenusList, myGenus & "!") > 0 Then

rng.Text = Left(myGenus, 1) & "."

Else

myGenusList = myGenusList & myGenus & "!"

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Debug.Print myGenusList

Loop

End Sub

**Sub SpellCheckerPlus()**

**'** Paul Beverley - Version 27.05.20

' Spellchecks from cursor, using multiple languages

OKwords = " etc "

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

Selection.Collapse wdCollapseStart

langText = Languages(Selection.LanguageID).NameLocal

CR = vbCr: CR2 = CR & CR

OKwords = Replace(" " & OKwords & " ", " ", CR)

Set rngAll = ActiveDocument.Content

For Each wd In rng.Words

DoEvents

w = Trim(wd)

If Application.CheckSpelling(wd, \_

MainDictionary:=langText) = False \_

And InStr(pbExceptionsList, CR & w & CR) = 0 Then

wd.Select

Set suggList = Application.GetSpellingSuggestions(wd, \_

MainDictionary:=langText)

myPrompt = "{{ " & w & " }} \* "

If suggList.count > 0 Then

newWord = suggList.Item(1).Name

myPrompt = myPrompt & "Alt: >> " & newWord & " <<"

Else

myPrompt = myPrompt & "No alternative"

End If

myInput = InputBox(myPrompt & CR2 & \_

"1 = Replace one" & CR & "2 = Replace all" & CR & \_

"3 = Add to exceptions list" & CR & \_

"0 = Exit", "SpellCheckPlus")

Select Case myInput

Case "": 'Give up

Selection.Collapse wdCollapseEnd

Exit Sub

Case "0": 'Give up

Selection.Collapse wdCollapseEnd

Exit Sub

Case "1": ' Replace once

If Right(wd, 1) = " " Then newWord = newWord & " "

wd.Text = newWord

Case "2": ' Replace all

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = w

.Wrap = wdFindContinue

.Replacement.Text = newWord

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Case "3": ' Replace all

pbExceptionsList = pbExceptionsList & w & CR

Case Else: ' Ignore it

End Select

Debug.Print Replace(pbExceptionsList, CR, " ")

End If

Next wd

Beep

End Sub

**Sub SrikeAllHighlightText()**

**'** Paul Beverley - Version 25.10.20

' Adds strikethrough to all highlighted text

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

' If you want to count them...

' Note where the end of the found item is

endNow = rng.End

rng.Font.StrikeThrough = True

' Be sure you're past the previous occurrence

rng.Start = endNow

rng.End = endNow

' Go and find the next occurrence (if there is one)

rng.Find.Execute

Loop

End Sub

**Sub StyleFollowingHeadings()**

**'** Paul Beverley - Version 13.04.18

' Sets the style of the paragraph after each heading

styleFollowing = "NoIndent"

afterTheseStyles = "Heading 1, Heading 2, Heading 3"

For Each myPar In ActiveDocument.Paragraphs

If doNextPara Then myPar.Range.Style = styleFollowing

st = myPar.Range.Style

doNextPara = InStr(afterTheseStyles, st) > 0

Next myPar

End Sub

**Sub SuperscriptHighlightSmart()**

**'** Paul Beverley - Version 25.09.15

' Highlight all superscript numbers according to type

ignoreThese = "cm,m,"

myColourFoot = wdBrightGreen

myColourSuper = wdYellow

myColourQuery = wdGray25

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColourFoot

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^2"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Replacement.Highlight = True

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = myColourSuper

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]{1,}"

.Font.Superscript = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myStart = rng.Start

myEnd = rng.End

rng.Collapse wdCollapseStart

rng.MoveStart , -1

Do

rng.MoveStart , -1

fstChar = rng.Characters(1)

DoEvents ' Debug.Print rng

Loop Until UCase(fstChar) = LCase(fstChar)

rng.MoveStart , 1

testWord = rng.Text

rng.Start = myStart

rng.End = myEnd

If InStr(ignoreThese, testWord) = 0 Then

rng.HighlightColorIndex = myColourSuper

Else

rng.HighlightColorIndex = myColourQuery

End If

rng.Collapse wdCollapseEnd

' Go and find the next occurence (if there is one)

rng.Find.Execute

Loop

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub SurnamesAllCaps()**

**'** Paul Beverley - Version 18.09.15

' Capitalise all the surnames in a refs list

andInCaps = False

paraNum = ActiveDocument.Range(0, Selection.Paragraphs(1).Range.End).Paragraphs.count

For para = paraNum To ActiveDocument.Paragraphs.count

Set p = ActiveDocument.Paragraphs(para).Range

If Len(p.Text) < 4 Then

p.Select

Selection.Collapse wdCollapseEnd

Beep

Exit Sub

End If

For i = 1 To p.Words.count

doCapIt = True

w = p.Words(i).Text

If Val(w) > 100 Then Exit For

If Trim(w) = "and" And andInCaps = False Then doCapIt = False

If doCapIt = True Then p.Words(i).Case = wdUpperCase

Next i

Next para

Selection.Collapse wdCollapseEnd

End Sub

**Sub SymbolShowNumber()**

**'** Paul Beverley - Version 28.10.13

' Types out the 'unicode' of the symbol font character

Selection.MoveEnd , 1

symbolCode = Replace(Hex(Dialogs(wdDialogInsertSymbol).CharNum), "FFFF", "")

Selection.Collapse wdCollapseEnd

Selection.TypeText symbolCode & ",;"

Selection.MoveRight , 1

End Sub

**Sub TablesMarkAllEmptyCells()**

**'** Paul Beverley - Version 09.06.22

' Adds some dummy text into every empty cell

dummyText = "Hello!"

myHighlight = wdYellow

' myHighlight = wdNoHighlight

Set rng = ActiveDocument.Content

For Each myTable In rng.Tables

For Each myCell In myTable.Range.Cells

If Len(myCell.Range.Text) < 3 Then

myCell.Range.InsertBefore Text:=dummyText

If myHighlight > 0 Then myCell.Range.HighlightColorIndex = myHighlight

End If

DoEvents

Next myCell

Next myTable

Beep

End Sub

**Sub TenseChanger()**

**'** Paul Beverley - Version 18.01.20

' Finds one of the words from a list, then changes to its alternate

myWords = "am,are,is,have,has,can,"

myWords = myWords & ",does,do,goes,go,"

myRange = 200

myWords = "," & myWords & ","

For i = 1 To myRange

Selection.MoveRight wdWord, 1

Selection.Expand wdWord

Selection.MoveEndWhile cset:=ChrW(8217) & " '", count:=wdBackward

If InStr(myWords, "," & Selection & ",") > 0 Then Exit For

DoEvents

Next i

If i > myRange Then

Beep

Else

Application.Run macroName:="MultiSwitch"

End If

End Sub

**Sub TimecodeEditor()**

**'** Paul Beverley - Version 22.10.14

' Increments/decrements timecode in a file

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Selection.End = Selection.End + 5

myText = Selection

wasStart = Selection.Start

wasMins = Val(Left(myText, 2))

wasSecs = Val(Right(myText, 2))

userInput = InputBox("+/- seconds?", "TimecodeEditor")

extraSecs = Val(userInput)

If extraSecs = 0 Then

myResponse = MsgBox("Please type + or -, and a number of seconds", \_

vbExclamation, "TimecodeEditor")

Exit Sub

End If

' Calculate new timecode

myNewSecs = wasSecs + extraSecs

secsNew = myNewSecs

minsNew = wasMins

If myNewSecs < 0 Then

secsNew = 60 + myNewSecs Mod 60

minsNew = wasMins + Int(myNewSecs / 60)

End If

If myNewSecs >= 59 Then

secsNew = myNewSecs Mod 60

minsNew = wasMins + Int(myNewSecs / 60)

End If

minsText = Right("0" & Trim(Str(minsNew)), 2)

secsText = Right("0" & Trim(Str(secsNew)), 2)

newTime = minsText & "." & secsText

myResponse = MsgBox("You typed " & userInput & vbCr & \_

"New timecode = " & newTime & " OK?", \_

vbQuestion + vbYesNo, "TimecodeEditor")

If myResponse <> vbYes Then

Beep

Exit Sub

End If

Selection.TypeText newTime

' Go and find the first occurrence

Set rng = ActiveDocument.Content

rng.Start = Selection.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p^#^#.^#^#"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While rng.Find.Found = True

' Note where the end of the found item is

endNow = rng.End

rng.Start = rng.Start + 1

myText = rng.Text

wasMins = Val(Left(myText, 2))

wasSecs = Val(Right(myText, 2))

' Calculate new timecode

myNewSecs = wasSecs + extraSecs

secsNew = myNewSecs

minsNew = wasMins

If myNewSecs < 0 Then

secsNew = 60 + myNewSecs Mod 60

minsNew = wasMins + Int(myNewSecs / 60)

End If

If myNewSecs >= 59 Then

secsNew = myNewSecs Mod 60

minsNew = wasMins + Int(myNewSecs / 60)

End If

minsText = Right("0" & Trim(Str(minsNew)), 2)

secsText = Right("0" & Trim(Str(secsNew)), 2)

newTime = minsText & "." & secsText

rng.Text = newTime

' Be sure you're past the previous occurrence

rng.Start = endNow + 5

rng.Find.Execute

Loop

Beep

End Sub

**Sub TimestampRemover()**

**'** Paul Beverley - Version 29.08.21

' Deletes all timestamps of the form: nn:nn:nn<sp>

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<([0-9]{2}):([0-9]{2}):([0-9]{2})^32"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Beep

End Sub

**Sub TrackChangeReport()**

**'** Paul Beverley - Version 21.05.24

' Creates a file of sentences containing (at least n-number of) TCs

minTracksPerSentence = 3

' Increase this number if you only want to list

' the more highly tracked paragraphs

beforeAndAfter = False

' beforeAndAfter = True

' Set to True if you want to display the original sentence

' as well as the edited sentence

mySpace = vbCr & vbCr

ActiveDocument.TrackRevisions = False

Set sourceText = ActiveDocument

Documents.Add

Set myReport = ActiveDocument

Set rptRng = ActiveDocument.Content

rptRng.Collapse wdCollapseEnd

sntEnd = 0

For Each snt In sourceText.Sentences

numTracks = snt.Revisions.Count

If numTracks >= minTracksPerSentence Then

' Set rng = snt.Duplicate

If Right(snt, 2) = vbCr & vbCr Then

snt.MoveEnd , -1

End If

snt.Copy

rptRng.Paste

If beforeAndAfter = True Then

Set rng = rptRng.Duplicate

rng.Revisions.RejectAll

rptRng.Collapse wdCollapseEnd

rptRng.Paste

End If

rptRng.Collapse wdCollapseEnd

rptRng.InsertAfter Text:=mySpace

rptRng.Collapse wdCollapseEnd

End If

DoEvents

Next snt

myReport.Activate

Selection.HomeKey Unit:=wdStory

myFileName = sourceText.Name

If Left(myFileName, 3) <> "Doc" Then

myFileName = Left(myFileName, InStr(sourceText.Name, ".") - 1)

End If

Selection.TypeText myFileName & mySpace

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

Selection.MoveLeft , 2

End Sub

**Sub TracksInlineToggle()**

**'** Paul Beverley - Version 07.03.22

' Toggles track change deletions between bubbles and inline

' At the top of the VBA macros file insert:

' Private pbBalloons As Boolean

With ActiveWindow.View.RevisionsFilter

.Markup = wdRevisionsMarkupAll

.View = wdRevisionsViewFinal

End With

If pbBalloons = False Then

Application.Run macroName:="UseBalloons"

pbBalloons = True

Else

Application.Run macroName:="NoInsertionDeletionBalloons"

pbBalloons = False

End If

End Sub

**Sub URLlinksVisible()**

**'** Paul Beverley - Version 01.05.21

' Unlinks all the URLs in the selection or the whole file

foreText = " ("

afterText = ")"

'foreText = " "

'afterText = " "

makeTextBold = False

remove1 = "http://"

remove2 = "https://"

remove1 = ""

remove2 = ""

If Selection.Start <> Selection.End Then

Set rng = Selection.Range.Duplicate

Else

Set rng = ActiveDocument.Content

End If

For i = rng.Fields.count To 1 Step -1

Set fld = rng.Fields(i)

myType = fld.Type

If myType = 88 Then

fld.Select

myURLtext = fld.Code

myDisplayText = fld.Result

myURLtext = Replace(myURLtext, "HYPERLINK """, "")

qtPos = InStr(myURLtext, """")

myURLtext = Trim(Left(myURLtext, qtPos - 1))

myURLtext = Replace(myURLtext, remove1, "")

myURLtext = Replace(myURLtext, remove2, "")

myDisplayText = Replace(myDisplayText, remove1, "")

myDisplayText = Replace(myDisplayText, remove2, "")

Debug.Print myDisplayText

Debug.Print myURLtext & vbCr

If myDisplayText <> myURLtext Then

Selection.InsertBefore Text:=myDisplayText & foreText

myStart = Selection.Start

fld.Result.Text = myURLtext

Selection.Collapse wdCollapseEnd

Selection.InsertAfter Text:=afterText

Selection.Range.Font.ColorIndex = wdAuto

Selection.Range.Underline = False

End If

Selection.Collapse wdCollapseEnd

End If

Next i

End Sub

**Sub WhatFolder()**

**'** Paul Beverley - Version 01.09.23

' Loads the clipboard with the address of folder of this file

myPath = ActiveDocument.Path

Selection.InsertAfter Text:=myPath

Selection.Range.Copy

WordBasic.EditUndo

End Sub

**Sub WordsMoveLeft()**

**'** Paul Beverley - Version 15.04.21

' Moves the partly selected words one word left

endNow = Selection.End

Selection.Collapse wdCollapseStart

Selection.MoveRight , 1

Selection.Expand wdWord

startNew = Selection.Start

Selection.End = endNow

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.Expand wdWord

Selection.Start = startNew

Selection.Cut

Selection.MoveLeft wdWord, 1

startHere = Selection.Start

Selection.Paste

Selection.Start = startHere

End Sub

**Sub WordsMoveRight()**

**'** Paul Beverley - Version 15.04.21

' Moves the partly selected words one word right

endNow = Selection.End

Selection.Collapse wdCollapseStart

Selection.MoveRight , 1

Selection.Expand wdWord

startNew = Selection.Start

Selection.End = endNow

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.Expand wdWord

Selection.Start = startNew

Selection.Cut

Selection.MoveRight wdWord, 1

startHere = Selection.Start

Selection.Paste

Selection.Start = startHere

End Sub

**Sub WordsMoveRight()**

**'** Paul Beverley - Version 15.04.21

' Moves the partly selected words one word right

endNow = Selection.End

Selection.Collapse wdCollapseStart

Selection.MoveRight , 1

Selection.Expand wdWord

startNew = Selection.Start

Selection.End = endNow

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.Expand wdWord

Selection.Start = startNew

Selection.Cut

Selection.MoveRight wdWord, 1

startHere = Selection.Start

Selection.Paste

Selection.Start = startHere

End Sub

**Sub ZeroIndentFirstLine()**

**'** Paul Beverley - Version 29.03.15

' Zero indent on paragraph after heading

paraNum = ActiveDocument.Range(0, \_

Selection.Paragraphs(1).Range.End).Paragraphs.count

For i = paraNum To ActiveDocument.Paragraphs.count - 3

Set rng1 = ActiveDocument.Paragraphs(i).Range

Set rng2 = ActiveDocument.Paragraphs(i + 1).Range

Set rng3 = ActiveDocument.Paragraphs(i + 2).Range

If rng1.Font.Bold = True And rng3.Font.Bold = False Then

rng1.Select

If Len(rng2.Text) > 3 Then

Beep

Exit Sub

End If

rng3.ParagraphFormat.FirstLineIndent = 0

End If

Next i

Beep

End Sub

**Sub SelectBetweenQuotes()**

**'** Paul Beverley - Version 17.07.23

' Selects text between double quotes and optionally does a Fetch

doFetchIt = True

myFetch = "GoogleFetch"

myRange = 100

Selection.MoveStartUntil cset:=ChrW(8220) & """", count:=wdBackward

If Len(Selection) > myRange Then

Beep

Exit Sub

End If

Selection.MoveEndUntil cset:=ChrW(8221) & """", count:=wdForward

If Len(Selection) > myRange Then

Beep

Exit Sub

End If

Selection.MoveEnd , -1

If doFetchIt = True Then Application.Run macroName:=myFetch

End Sub

**Sub DisplayedQuoteToInline()**

**'** Paul Beverley - Version 17.07.23

' Turns a displayed quote to an inline quote

useUSpunct = True

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Collapse wdCollapseStart

rng.MoveStart , -1

rng.Text = " " & ChrW(8220)

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

rng.MoveStart , -1

rng.Text = " "

rng.Collapse wdCollapseStart

rng.MoveStart , -1

If InStr(",.", rng.Text) > 0 And useUSpunct = True Then

rng.InsertBefore Text:=ChrW(8221)

Else

rng.InsertAfter Text:=ChrW(8221)

End If

End Sub

**Sub** **TablesStrikeThroughAll()**

' Paul Beverley - Version 19.07.23

' Adds a strike-through to all table text

For Each myTable In ActiveDocument.Tables

myTable.Range.Font.StrikeThrough = True

Next myTable

End Sub

**Sub ReferenceErrorHighlight()**

**'** Paul Beverley - Version 19.07.23

' Highlights references that aren't found

For Each fld In ActiveDocument.Fields

If fld.Type = 3 Then

myResult = fld.Result

myResult = Mid(myResult, 2)

If myResult = "0" Then

fld.Select

Selection.MoveStart , -8

Selection.MoveEnd , 1

Selection.Range.HighlightColorIndex = wdBrightGreen

Selection.Collapse wdCollapseEnd

End If

End If

DoEvents

Next fld

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub LinkProblemHighlighter()**

**'** Paul Beverley - Version 19.07.23

' Highlights links that don't go anywhere

For Each fld In ActiveDocument.Fields

If fld.Type > 0 Then

If InStr(fld.Code, "about:blank") > 0 Then

fld.Select

Selection.Range.HighlightColorIndex = wdBrightGreen

End If

End If

DoEvents

Next fld

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub DisplayedQuotesShortHighlight()**

**'** Paul Beverley - Version 17.07.23

' Highlights all displayed quotes that are too short

maxWords = 50

myColour = wdYellow

For Each myPar In ActiveDocument.Paragraphs

If myPar.Range.Words.count < maxWords Then

If myPar.Range.ParagraphFormat.LeftIndent > 0 Then

myPar.Range.HighlightColorIndex = myColour

End If

End If

DoEvents

Next myPar

End Sub

**Sub MoveToEnd()**

**'** Paul Beverley - Version 22.07.23

' Moves the selected text to the end of the sentence

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Selection.Range.Characters.Last) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Expand wdWord

Selection.Start = rng.Start

rng.End = Selection.End

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", rng.Characters.Last) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.End = rng.End

End If

Set rng = Selection.Range.Duplicate

rng.Select

Selection.MoveStart , -1

Selection.Cut

Selection.Expand wdSentence

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.Paste

End Sub

**Sub NoteCitationsOutsidePunctuationGlobal()**

**'** Paul Beverley - Version 28.07.23

' Moves all note numbers to outside the punctuation

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "(^2)([.,:;\!\?])"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "\2\1"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub LinkProblemHighlight()**

**'** Paul Beverley - Version 19.07.23

' Highlights links that don't go anywhere

For Each fld In ActiveDocument.Fields

If fld.Type > 0 Then

If InStr(fld.Code, "about:blank") > 0 Then

fld.Select

Selection.Range.HighlightColorIndex = wdBrightGreen

End If

If InStr(fld.Code, "wiki") > 0 Then

fld.Select

Selection.Range.HighlightColorIndex = wdYellow

End If

End If

DoEvents

Next fld

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub ApostropheAdd()**

**'** Paul Beverley - Version 08.12.22

' Adds an apostrophe, highlighted

myColour = wdBrightGreen

Selection.InsertAfter Text:=ChrW(8217)

Selection.Range.HighlightColorIndex = myColour

Selection.Collapse wdCollapseEnd

End Sub

**Sub BackgroundPatternStyleChecker()**

**'** Paul Beverley - Version 26.01.23

' Searches for text style with background pattern applied

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

For Each myPara In rng.Paragraphs

myPara.Range.Select

If myPara.Range.Style.ParagraphFormat.Shading.BackgroundPatternColor \_

<> wdColorAutomatic Then

Beep

myPara.Range.Select

Exit Sub

End If

DoEvents

Next myPara

End Sub

**Sub BinomialAdder()**

**'** Paul Beverley - Version 04.04.22

' Adds Latin binomials into a document

nameFile = "bird\_names"

captionStyle = "Caption"

gotNamesFile = False

For Each myDoc In Documents

myName = myDoc.Name

If InStr(myName, nameFile) > 0 Then

Set rngN = myDoc.Content

gotNamesFile = True

End If

Next myDoc

If gotNamesFile = False Then

Beep

myResponse = MsgBox("Please open file: " & nameFile, \_

vbExclamation + vbOK, "BinomialAdder")

Exit Sub

End If

allText = LCase(rngN.Text)

For Each myPar In rngN.Paragraphs

myText = myPar.Range.Text

If Len(myText) < 2 Then Exit Sub

commaPos = InStr(myText, ",")

If commaPos = 0 Then

Beep

MsgBox ("Missing comma")

Exit Sub

End If

engName = LCase(Left(myText, commaPos - 1))

latinName = Replace(Mid(myText, commaPos + 1), vbCr, "")

' Debug.Print engName

' StatusBar = " " & engName

DoEvents

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = engName

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchCase = False

.MatchWildcards = False

.Execute

DoEvents

End With

If rng.Find.Found = False Then GoTo goNextBird

doneThisOnce = False

Do While rng.Find.Found

Set rng2 = rng.Duplicate

rng2.MoveStart wdWord, -1

thisName = Trim(LCase(rng2))

If InStr(allText, thisName) > 0 Then GoTo goNext

rng2.MoveStart wdWord, -1

thisName = Trim(LCase(rng2))

If InStr(allText, thisName) > 0 Then GoTo goNext

' Found this bird

rng.Select

Selection.MoveEnd , 1

If Right(Selection, 1) <> "s" Then Selection.MoveEnd , -1

spPos = InStr(latinName, " ")

latinNameAbbr = Left(latinName, 1) & "." & Mid(latinName, spPos)

Debug.Print latinNameAbbr

If rng.Information(wdWithInTable) Then

addText = latinName

Else

If doneThisOnce = False Then

' The first time in the document

addText = latinName

doneThisOnce = True

Else

If rng.Style = captionStyle Then

Set rng2 = rng.Duplicate

rng2.Expand wdParagraph

rng2.End = rng.Start

If InStr(LCase(rng2), engName) = 0 Then

addText = latinName

Else

addText = latinNameAbbr

End If

Else

addText = latinNameAbbr

End If

End If

End If

Selection.Collapse wdCollapseEnd

Selection.InsertAfter Text:=" " & addText

Selection.Font.Bold = True

goNext:

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

goNextBird:

Next myPar

End Sub

**Sub BoldThisWord()**

**'** Paul Beverley - Version 11.08.22

' Makes the word at the cursor bold

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Font.Bold = True

Selection.Collapse wdCollapseEnd

End Sub

**Sub ChicagoNoteReferenceAlyse()**

**'** Paul Beverley - Version 31.12.21

' Helps to check note-based reference citations

myColour = wdGray25

myColour = wdYellow

If InStr(ActiveDocument.Paragraphs(1), "References") > 0 Then GoTo mySurname

myScreenOff = True

myResponse = MsgBox("Analyse Chicago references?", vbQuestion \_

+ vbOK, "ChicagoNoteReferenceAlyse")

If myResponse <> vbOK Then Exit Sub

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

Set myDoc = ActiveDocument

Documents.Add

Set rng = ActiveDocument.Content

If myDoc.Endnotes.count > 0 Then

Set rngOld = myDoc.StoryRanges(wdEndnotesStory)

rng.FormattedText = rngOld.FormattedText

End If

If myDoc.Footnotes.count > 0 Then

Set rngOld = myDoc.StoryRanges(wdFootnotesStory)

rng.FormattedText = rngOld.FormattedText

End If

Set rng = ActiveDocument.Content

rng.InsertBefore Text:="References list" & vbCr

rng.ParagraphFormat.LineSpacingRule = wdLineSpaceSingle

rng.ParagraphFormat.SpaceBefore = 12

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^02^32"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Highlight the "surname"

For Each myPar In ActiveDocument.Paragraphs

prevWord = myPar.Range.Words(1)

For i = 2 To myPar.Range.Words.count

wd = myPar.Range.Words(i).Text

Debug.Print wd

If Left(wd, 1) = "," Or Left(wd, 1) = "and" Then

myPar.Range.Words(i - 1).HighlightColorIndex = myColour

Exit For

End If

DoEvents

Next i

Next myPar

Application.ScreenUpdating = True

Exit Sub

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "ibid"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the end of the found item is

endNow = rng.End

fsdfsdf = rng.Font.Italic

rng.Select

' Be sure you're past the previous occurrence

rng.Start = endNow

rng.End = endNow

' Go and find the next occurrence (if there is one)

rng.Find.Execute

DoEvents

Loop

mySurname:

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

Set rng = Selection.Range.Duplicate

For i = 1 To rng.Words.count

Next i

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub CleanUp()**

**'** Paul Beverley - Version 10.06.20

' Cleans up a file by F&R

For i = 1 To 3

changeIt = False

If i = 1 Then

Set rng = ActiveDocument.Content

changeIt = True

If i = 2 And ActiveDocument.Footnotes.count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

changeIt = True

End If

If i = 3 And ActiveDocument.Endnotes.count > 0 Then \_

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

changeIt = True

End If

If changeIt = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " "

.Wrap = wdFindContinue

.Replacement.Text = " "

.Forward = True

.MatchCase = True

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

.Text = " & "

.Replacement.Text = " and "

.Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

.Text = "#"

.Replacement.Text = "^&"

.Execute Replace:=wdReplaceAll

.MatchWildcards = True

.Text = "<etc>([!.])"

.Replacement.Text = "etc.\1"

.Execute Replace:=wdReplaceAll

.MatchWildcards = False

.Text = " ie "

.Replacement.Text = " i.e. "

.Execute Replace:=wdReplaceAll

.Text = " eg "

.Replacement.Text = " e.g. "

.Execute Replace:=wdReplaceAll

.Text = " / "

.Replacement.Text = "/"

.Execute Replace:=wdReplaceAll

.Text = "/ "

.Replacement.Text = "/"

.Execute Replace:=wdReplaceAll

.Text = " /"

.Replacement.Text = "/"

.Execute Replace:=wdReplaceAll

.Text = " - "

.Replacement.Text = " ^= "

.Execute Replace:=wdReplaceAll

' switch highlighting off

.Replacement.ClearFormatting

.Text = "'"

.Replacement.Text = "^039"

.Execute Replace:=wdReplaceAll

.Text = """"

.Replacement.Text = "^034"

.Execute Replace:=wdReplaceAll

.Text = "-ly"

.Replacement.Text = "ly"

.Execute Replace:=wdReplaceAll

End With

End If

Next i

End Sub

**Sub ClipCopy()**

**'** Paul Beverley - Version 06.02.21

' Copies the selected text into a ClipStore file

myHighlight = 0

myHighlight = wdYellow

myClipFile = "zClipStore.docx"

' On Windows, it will need to be something like:

myFolder = "C:\Documents and Settings\Paul\My Documents\Macro stuff\"

' On a Mac, it will need to be something like:

myFolder = "/Users/Paul/My Documents/Macro stuff/"

maxClips = 12

CR = vbCr: CR2 = CR & CR

' Backstore items

Dim bStart(99) As Integer

Dim bEnd(99) As Integer

Dim bName(99) As String

' Main clip items

Dim cTitle(99) As Integer

Dim cStart(99) As Integer

Dim cEnd(99) As Integer

Dim cName(99) As String

' Remember start selection

Set rngWas = Selection.Range.Duplicate

Set startDoc = ActiveDocument

myFilename = startDoc.Name

dotPos = InStr(myFilename, ".")

If dotPos > 0 Then myFilename = Left(myFilename, dotPos - 1)

' Look for an open ClipStore file

On Error GoTo ReportIt

gottaStoreFile = False

For i = 1 To Application.Windows.count

Set thisFile = Application.Windows(i).Document

If InStr(thisFile.Name, myClipFile) > 0 Then

Set clipDoc = Application.Windows(i).Document

gottaStoreFile = True

Exit For

End If

Next i

' If it's not open, load it

If gottaStoreFile = False Then

Documents.Open myFolder & myClipFile

Set clipDoc = ActiveDocument

End If

On Error GoTo 0

cursorInClipFile = (startDoc = clipDoc)

' Interpret the contents of the ClipStore

inBackStore = True

bNum = 0

cNum = 0

For i = 1 To clipDoc.Paragraphs.count

Set pa = clipDoc.Paragraphs(i).Range

tx = Replace(pa.Text, CR, "")

If Left(tx, 3) = "max" Then maxClips = Val(Right(tx, 2))

If Left(tx, 3) = "for" Then textOnly = (Right(tx, 1) = "-")

If Left(tx, 4) = "999" Then

inBackStore = False

bEnd(bNum) = pa.Start - 1

endBackupClips = pa.Start - 1

End If

If Left(tx, 3) = "000" Then

bEnd(bNum) = pa.Start - 1

startBackupClips = pa.End

End If

If Left(tx, 1) = "\_" Then

If inBackStore = True Then

' store parameters for backstore items

bNum = bNum + 1

bStart(bNum) = pa.End

If bNum > 1 Then bEnd(bNum - 1) = pa.Start - 1

bNumMax = bNum

Else

' store parameters for clipstore items

cNum = Val(Right(pa.Text, 3))

cTitle(cNum) = pa.Start

cStart(cNum) = pa.End

End If

End If

cEnd(cNum) = pa.End

Next i

bEnd(0) = startBackupClips - 1

cEnd(cNum) = clipDoc.Content.End

newClipNum = (cNum Mod maxClips) + 1

newClipNumText = Right(Trim(Str(100 + newClipNum)), 2)

If cursorInClipFile = True And rngWas.End > endBackupClips Then

Beep

Exit Sub

End If

' To add text from target file to ClipStore, ask...

' Has this new clip number already been used?

Set rng = clipDoc.Content

oldClipPos = InStr(rng.Text, "\_" & newClipNumText & CR)

If oldClipPos > 0 Then

Set rng = clipDoc.Range(cTitle(newClipNum), cEnd(newClipNum))

rng.Delete

End If

Set rng = clipDoc.Content

rng.Collapse wdCollapseEnd

' Add this clip to numbered clip area ...

If cursorInClipFile = False Then

' ... either from the target file

rng.InsertAfter Text:="\_" & myFilename & "\_" & newClipNumText & vbCr

rng.HighlightColorIndex = myHighlight

rng.Collapse wdCollapseEnd

If Len(rngWas) > 1 Then

rngWas.Copy

rng.Paste

Else

rng.Paste

End If

rng.InsertAfter Text:=CR

Else

' ... or from ClipStore's own backstore

csrPos = rngWas.Start

For i = 1 To bNumMax

If bEnd(i - 1) > csrPos Then Exit For

Next i

bNum = i - 1

Set rngClip = ActiveDocument.Range(bEnd(bNum - 1) + 1, bEnd(bNum) + 1)

rngClip.Copy

rng.Paste

numPos = InStr(rng, "\_xx")

If numPos > 0 Then

numPos = rng.Start + numPos - 1

Set rng = clipDoc.Range(numPos, numPos + 3)

rng.Text = "\_" & newClipNumText

End If

End If

rng.Select

Selection.EndKey Unit:=wdStory

startDoc.Activate

Exit Sub

ReportIt:

If Err.Number = 5174 Then

Err.Clear

Beep

myClipFile = Replace(myClipFile, ".docx", "")

myResponse = MsgBox("Can't find your clipboard file: """ \_

& myClipFile & """", vbOKOnly, "ClipStore")

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub LongWordHighlighter()**

**'** Paul Beverley - Version 09.08.23

' Highlights all words more than a certain length

med = 8

high = 12

mega = 16

Set rng = Selection.Range.Duplicate

If Len(rng) < 2 Then rng.Expand wdParagraph

For Each myWord In rng.Words

If myWord.Characters.count > mega Then

myWord.HighlightColorIndex = wdBrightGreen

Else

If myWord.Characters.count > med Then \_

myWord.HighlightColorIndex = wdGray25

If myWord.Characters.count > high Then \_

myWord.HighlightColorIndex = wdYellow

End If

Next

End Sub

**Sub MultiSyllableWordsHighlight()**

**'** Paul Beverley - Version 10.08.23

' Highlights words of 3, 4 or 5+ syllables

' Set the range to the current paragraph

Set rng = Selection.Range.Duplicate

If Len(rng) < 2 Then rng.Expand wdParagraph

For Each wd In rng.Words

If wd.Font.StrikeThrough = False Then

' Loop through each character in the word

syllCount = 0

myWord = Replace(LCase(wd.Text), " ", "")

myWord = Replace(myWord, ChrW(8217), "")

myWord = Replace(myWord, "'", "")

For i = 1 To Len(myWord)

' Check if the character is a vowel

If InStr("aeiou", Mid(myWord, i, 1)) > 0 Then

' Increment syllable count

If wasVowel = False Then syllCount = syllCount + 1

wasVowel = True

Else

wasVowel = False

End If

Next i

' Adjust syllable count based on common patterns

If syllCount > 1 And InStr("ed es", Right(myWord, 2)) > 0 Then \_

syllCount = syllCount - 1

If syllCount > 1 And InStr("e", Right(myWord, 1)) > 0 Then \_

syllCount = syllCount - 1

' Colour according to number of syllables

If syllCount > 4 Then

wd.HighlightColorIndex = wdBrightGreen

Else

If syllCount > 2 Then \_

wd.HighlightColorIndex = wdGray25

If syllCount > 3 Then \_

wd.HighlightColorIndex = wdYellow

End If

End If

DoEvents

Next wd

Beep

End Sub

**Sub IncludeBracketInSentence()**

**'** Paul Beverley - Version 11.08.23

' Moves the full point to after the bracketed reference

Selection.Expand wdWord

Selection.Collapse wdCollapseEnd

Selection.MoveEnd , 1

Selection.Cut

Selection.Expand wdParagraph

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 1

Selection.Paste

End Sub

**Sub LanguageSwitch()**

**'** Paul Beverley - Version 16.08.23

' Switches the language between two or three alternates

Lan1 = wdEnglishUS

Lan2 = wdEnglishUK

Lan3 = wdEnglishCanadian

' for two languages only, use:

' Lan3 = 0

' If you want all the styles to be language-set as well,

' make it true:

setStyleLanguage = False

'setStyleLanguage = True

nowLanguage = Selection.Range.LanguageID

If Lan3 <> 0 Then

Select Case nowLanguage

Case Lan1: myLanguage = Lan2

Case Lan2: myLanguage = Lan3

Case Lan3: myLanguage = Lan1

Case Else: myLanguage = Lan1

End Select

Else

Select Case nowLanguage

Case Lan1: myLanguage = Lan2

Case Lan2: myLanguage = Lan1

Case Else: myLanguage = Lan1

End Select

End If

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

ActiveDocument.Content.LanguageID = myLanguage

For Each aStory In ActiveDocument.StoryRanges

Do

aStory.LanguageID = myLanguage

If aStory.NextStoryRange Is Nothing Then

MoreStoryRanges = False

Else

MoreStoryRanges = True

Set aStory = aStory.NextStoryRange

End If

Loop While MoreStoryRanges

DoEvents

Next aStory

If setStyleLanguage = True Then

For i = 1 To ActiveDocument.Styles.count

Set sty = ActiveDocument.Styles(i)

If sty.Type = wdStyleTypeParagraph Then \_

sty.LanguageID = myLanguage

DoEvents

Next i

End If

If ActiveDocument.Shapes.count > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

shp.TextFrame.TextRange.LanguageID = myLanguage

End If

End If

DoEvents

Next

End If

ActiveDocument.SpellingChecked = False

ActiveDocument.GrammarChecked = False

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub GoogleFetchEgypt()**

' Paul Beverley - Version 30.08.23

' Launches selected text on Google

mySite = "http://www.google.com.eg/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub GoogleFetchQuotesEgypt()**

' Paul Beverley - Version 30.08.23

' Launches selected text - with quotes - on Google

mySite = "http://www.google.com.eg/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, vbCr, "")

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = "%22" & mySubject & "%22"

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub TabbedParasToIndent()**

**'** Paul Beverley - Version 30.08.23

' Finds tabbed paras, deletes them and adds first-line indent instead

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p^t"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

' If you want to count them...

myCount = myCount + 1

rng.MoveStart , 1

rng.Delete

If myCount Mod 20 = 0 Then rng.Select

rng.ParagraphFormat.FirstLineIndent = InchesToPoints(0.5)

rng.Find.Execute

DoEvents

Loop

Beep

MsgBox "Changed: " & myCount

End Sub

**Sub TemplatesFolderAddressMac()**

**'** Paul Beverley - Version 10.02.21

' Loads the clipboard with the address of your Templates folder

myPath = Application.NormalTemplate.Path

myPath = Replace(myPath, "Macintosh HD:", "/")

myPath = Replace(myPath, ":", "/")

Documents.Add

Selection.TypeText Text:=myPath

Selection.WholeStory

Selection.Copy

Selection.Collapse wdCollapseEnd

ActiveDocument.Close SaveChanges:=False

End Sub

**Sub SelectionGrowRight()**

**'** Paul Beverley - Version 02.09.23

' Extends the selection word by word to the left

Set rng = Selection.Range.Duplicate

If rng.Start = rng.End Then

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

rng.Select

Exit Sub

End If

Do

endNow = rng.End

rng.MoveEnd wdWord, 1

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

' If the selection hasn't increased, move two words

If rng.End = endNow Then

rng.MoveEnd wdWord, 2

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

End If

If rng.End = ActiveDocument.Content.End Then

Beep

Exit Sub

End If

lastChar = rng.Characters.Last

DoEvents

Loop Until LCase(lastChar) <> UCase(lastChar)

rng.Select

End Sub

**Sub SelectionGrowLeft()**

**'** Paul Beverley - Version 02.09.23

' Extends the selection word by word to the left

Set rng = Selection.Range.Duplicate

If rng.Start = rng.End Then

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

rng.Select

Exit Sub

End If

If rng.Start = 0 Then

Beep

Exit Sub

End If

Do

rng.MoveStart wdWord, -1

preChar = rng.Characters.First

DoEvents

Loop Until LCase(preChar) <> UCase(preChar)

rng.Select

End Sub

**Sub AllCapWordsCollect()**

**'** Paul Beverley - Version 15.09.23

' Collects all the words in all caps style

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-zA-Z]{1,}"

.Font.AllCaps = True

.Wrap = wdFindStop

.MatchWildcards = True

.Execute

End With

Documents.Add

Set rng2 = ActiveDocument.Content

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

If myCount Mod 20 = 0 Then rng.Select

rng2.InsertAfter Text:=rng & " "

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Beep

MsgBox "Found: " & myCount

End Sub

**Sub CommentsResolveSelected()**

**'** Paul Beverley - Version 19.09.23

' Resolves all the comments in the selected area of text

numComments = ActiveDocument.Comments.Count

If Selection.Start = Selection.End Then

Beep

myResponse = MsgBox("Resolve ALL the comments in the whole document?!", \_

vbQuestion + vbYesNo, "CommentsResolveSelected")

If myResponse <> vbYes Then Exit Sub

cmtFirst = 1: cmtLast = numComments

Else

Set rng = Selection.Range.Duplicate

rng.Start = 0

cmtLast = rng.Comments.Count

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Start = 0

cmtFirst = rng.Comments.Count + 1

End If

For i = cmtFirst To cmtLast

ActiveDocument.Comments(i).Done = True

DoEvents

Next i

Selection.Collapse wdCollapseEnd

End Sub

**Sub SaveAllUnsavedDocs()**

**'** Paul Beverley - Version 26.10.23

' Saves all unsaved documents

' On a Mac, you will need something like:

myFolder = "/Users/Paul/My Documents/Macro\_stuff/"

' On Windows, you will need something like:

myFolder = "C:\Documents and Settings\Paul\Macro\_stuff\"

For Each myDoc In Documents

myName = myDoc.FullName

If InStr(myName, "\") = 0 And InStr(myName, "/") = 0 Then

newName = Trim(myDoc.Content.Paragraphs(1))

newName = Replace(newName, vbCr, "")

myDate = Replace(Left(Date, 3), "/", "")

myFullFilename = myFolder & newName & "\_" & myDate

Debug.Print myFullFilename

myDoc.SaveAs fileName:=myFullFilename

End If

DoEvents

Next myDoc

End Sub

**Sub FigureCaptionSplit()**

**'** Paul Beverley - Version 27.10.23

' Splits a caption into two lines, the second in italic

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.End = rng.End - 1

' In case the number is a link

rng.Text = rng.Text

tabPos = InStr(rng, vbTab)

If tabPos > 0 Then

rng.Start = rng.Start + tabPos - 1

rng.End = rng.Start + 1

Else

For i = 1 To rng.Characters.Count

If Val(rng.Characters(i)) > 0 Then

numStart = i

Exit For

End If

DoEvents

Next i

For i = numStart To rng.Characters.Count

nowPos = i

If InStr("0123456789.", rng.Characters(i)) = 0 Then

Exit For

End If

DoEvents

Next i

rng.Start = rng.Start + nowPos - 1

rng.End = rng.Start + 1

End If

rng.Delete

rng.InsertAfter vbCr

rng.MoveStart , 1

rng.Expand wdParagraph

rng.Font.Italic = True

rng.Collapse wdCollapseStart

rng.Select

End Sub

**Sub NTbackupDatedSimple()**

**'** Paul Beverley - Version 20.07.24

' Backs up the Normal template with date and time

backupFolder = "Backup"

nmlName = ThisDocument.Name

If nmlName <> "Normal.dotm" Then

Beep

CR2 = vbCr & vbCr

myPmt = "Possible issue with Normal file (" & nmlName & ")" & CR2

myPmt = myPmt & "Have you copied the macro text somewhere safe?" & CR2

myResponse = MsgBox(myPmt & "Continue with backup?", \_

vbQuestion + vbYesNoCancel, "NTbackupDatedSimple")

If myResponse <> vbYes Then

MsgBox ("For safety, copy all the VBA text and save it to a new Word file. OK?")

Exit Sub

End If

End If

NTname = NormalTemplate.FullName

' Mac or PC? Use correct delimiter

If InStr(NTname, "\") = 0 Then

backupFolder = backupFolder & "/"

Else

backupFolder = backupFolder & "\"

End If

' Add date & time

mySuffix = " " & Format(Now, "\_YY-MM-DD-hh-mm")

mySuffix = Replace(mySuffix, " \_", " 20")

newName = Replace(NTname, "Normal", backupFolder & "Normal" & mySuffix)

Debug.Print newName

ThisDocument.SaveAs newName

Beep

MsgBox ("Backup completed.")

End Sub

**Sub FigureCaptionSplitAll()**

**'** Paul Beverley - Version 30.10.23

' Splits all captions into two lines, the second in italic

allMyCaps = "Fig Figure Table Box "

maxSentences = 3

maxWords = 50

minWords = 2

Beep

myResponse = MsgBox("Split ALL captions? Really?!" \_

, vbQuestion + vbYesNoCancel, "FigureCaptionSplitAll")

If myResponse <> vbYes Then Exit Sub

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Collapse wdCollapseStart

rng.End = ActiveDocument.Content.End

Do

thisCap = Trim(rng.Words(1))

doThisOne = (InStr(allMyCaps, thisCap) > 0)

Debug.Print thisCap, doThisOne

If doThisOne Then

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

Debug.Print rng.Words.Count

If rng.Sentences.Count > maxSentences Then doThisOne = False

If rng.Words.Count > maxWords Then doThisOne = False

If rng.Words.Count < minWords + 2 Then doThisOne = False

End If

If doThisOne Then

rng.End = rng.End - 1

' In case the number is a link

rng.Text = rng.Text

tabPos = InStr(rng, vbTab)

If tabPos > 0 Then

rng.Start = rng.Start + tabPos - 1

rng.End = rng.Start + 1

Else

For j = 1 To rng.Characters.Count

If Val(rng.Characters(j)) > 0 Then

numStart = j

Exit For

End If

DoEvents

Next j

For j = numStart To rng.Characters.Count

nowPos = j

If InStr("0123456789.", rng.Characters(j)) = 0 Then

Exit For

End If

DoEvents

Next j

rng.Start = rng.Start + nowPos - 1

rng.End = rng.Start + 1

End If

rng.Delete

rng.InsertAfter vbCr

rng.MoveStart , 1

rng.Expand wdParagraph

rng.Font.Italic = True

rng.Select

rng.Collapse wdCollapseEnd

Else

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

End If

DoEvents

rng.End = ActiveDocument.Content.End

Loop Until rng.Paragraphs.Count < 2

Beep

End Sub

**Sub TitleCapAllItalic()**

**'** Paul Beverley - Version 13.11.23

' Changes all italic titles to title capitals.

' Start from the top of the document

Selection.HomeKey Unit:=wdStory

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

Do While Selection.Find.Found = True

' Note where the end of the found item is

endNow = Selection.End

' Make your required change

Call CapperMax

' Be sure you're past the previous occurrence

Selection.End = endNow

Selection.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

Selection.Find.Execute

Loop

End Sub

**Sub ChapterFileLinker()**

' Paul Beverley - Version 16.11.23

' Rejoins all the chapters of a book

bookName = "allTheBook"

Dim allMyFiles(200) As String

Set rng = ActiveDocument.Content

myDelay = 10

myExtent = 250

If rng.End - rng.Start > myExtent Then rng.End = rng.Start + myExtent

CR = vbCr

CR2 = CR & CR

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = \_

MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "ChapterFileLinker")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & CR

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

If Application.PathSeparator <> "\" Then

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

End If

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath & CR

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

myResponse = MsgBox("Collect formatted text " & moreText & \_

"from ALL the files in" & " directory:" & dirName & \_

" ?", vbQuestion + vbYesNoCancel, "ChapterFileLinker")

If myResponse <> vbYes Then Exit Sub

Else

myResponse = MsgBox("Collect formatted text " & moreText & \_

"from the files listed here?", vbQuestion + vbYesNoCancel, \_

"ChapterFileLinker")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles(numFiles) = thisFile

End If

End If

End If

Next myPara

' Now pick up formatted text from all the files

firstFile = myFolder & myDelimiter & allMyFiles(1)

Set bookDoc = Application.Documents.Open(fileName:=firstFile)

ActiveDocument.TrackRevisions = False

myResponse = MsgBox("Save combined file as: " & bookName & \_

"?", vbQuestion + vbYesNoCancel, "ChapterFileLinker")

ActiveDocument.SaveAs myFolder & myDelimiter & bookName

For j = 1 To myDelay

DoEvents

Next j

For i = 2 To numFiles

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=Chr(12)

thisFile = myFolder & myDelimiter & allMyFiles(i)

Set thisChapter = Application.Documents.Open(fileName:=thisFile)

thisChapter.TrackRevisions = False

thisChapter.Content.Copy

StatusBar = allMyFiles(i)

For j = 1 To myDelay

DoEvents

Next j

bookDoc.Activate

Selection.EndKey Unit:=wdStory

Selection.Paste

For j = 1 To myDelay

DoEvents

Next j

thisChapter.Close SaveChanges:=wdDoNotSaveChanges

Next i

Beep

bookDoc.Activate

ActiveDocument.Save

myResponse = MsgBox("Click OK here, and then WAIT until the flashing cursor returns." \_

& CR2 & "Not doing so can cause Word to crash." & CR2 & \_

"If unsure, try just pressing the cursor right key.", \_

vbOKOnly, "ChapterFileLinker")

End Sub

**Sub FieldCodesVisible()**

**'** Paul Beverley - Version 24.10.23

' Switches between fields being visible and not visible

Set rng = Selection.Range.Duplicate

ActiveWindow.View.ShowFieldCodes = Not ActiveWindow.View.ShowFieldCodes

rng.Select

ActiveDocument.ActiveWindow.LargeScroll up:=1

rng.Select

ActiveDocument.ActiveWindow.SmallScroll up:=1

End Sub

**Sub MarkupDisplaySwitch()**

**'** Paul Beverley - Version 23.11.23

' Cycles through track change display levels

' N.B. You need to put a line at the top of the VBA area:

' Private pbMarkupMode As Integer

' (but without the ' at the beginning of the line, of course)

includeNoMarkup = True

viewInBalloons = True

Set rng = Selection.Range.Duplicate

If pbMarkupMode = 2 And includeNoMarkup = False Then

pbMarkupMode = 3

End If

Debug.Print pbMarkupMode

Select Case pbMarkupMode

Case 1: markupNext = wdRevisionsMarkupAll

pbMarkupMode = 2

Debug.Print "All"

StatusBar = "All"

Case 2: markupNext = wdRevisionsMarkupNone

pbMarkupMode = 3

StatusBar = "No markup"

Debug.Print "No markup"

Case 3: markupNext = wdRevisionsMarkupSimple

Debug.Print "Simple"

StatusBar = "Simple"

pbMarkupMode = 1

Case Else

markupNext = wdRevisionsMarkupAll

pbMarkupMode = 2

StatusBar = "All"

Debug.Print "All"

End Select

With ActiveWindow.View.RevisionsFilter

.Markup = markupNext

.View = wdRevisionsViewFinal

End With

If viewInBalloons = True Then

ActiveWindow.View.MarkupMode = wdBalloonRevisions

Else

ActiveWindow.View.MarkupMode = wdInLineRevisions

End If

rng.Select

End Sub

**Sub MatchParentheses()**

**'** Paul Beverley - Version 24.11.23

' Check whether brackets match up

stopEachTime = False

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Set rngToGo = Selection.Range.Duplicate

rngToGo.End = ActiveDocument.Content.End

For Each myPara In rngToGo.Paragraphs

myText = myPara.Range.Text

If Len(myText) > 4 Then

myLen = Len(myText)

myOpenNum = myLen - Len(Replace(myText, "(", ""))

myCloseNum = myLen - Len(Replace(myText, ")", ""))

' Ignore "1)", or "a)" type lists

Set rng = myPara.Range

Debug.Print rng.Words(1) & rng.Words(2)

rng.End = rng.Start + 3

If InStr(rng.Text, ")") > 0 Then myCloseNum = myCloseNum - 1

If myOpenNum <> myCloseNum Then

If stopEachTime = True Then

Beep

rng.Expand wdParagraph

parenPos = InStr(rng.Text, "(")

If parenPos = 0 Then parenPos = InStr(rng.Text, ")")

If parenPos < 3 Then

rng.Start = rng.Start + 3

parenPos = InStr(rng.Text, "(")

If parenPos = 0 Then parenPos = InStr(rng.Text, ")")

End If

rng.Start = rng.Start + parenPos - 1

rng.End = rng.Start + 1

rng.Select

ActiveDocument.ActiveWindow.LargeScroll down:=1

rng.Select

ActiveDocument.ActiveWindow.SmallScroll down:=1

rng.Select

myResponse = MsgBox("Continue?", \_

vbQuestion + vbYesNo, "MatchParentheses")

If myResponse <> vbYes Then Exit Sub

Else

myPara.Range.Font.Underline = True

myCount = myCount + 1

StatusBar = "Found: " & myCount

DoEvents ' Debug.Print "Found: " & myCount

End If

End If

DoEvents

End If

Next

StatusBar = ""

Selection.HomeKey Unit:=wdStory

If myCount = 0 Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

MsgBox ("All clear!")

Else

MsgBox ("Number of suspect paragraphs: " & Trim(myCount))

End If

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Replacement.Text = ""

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

End Sub

**Sub DPTMFetch()**

**'** Paul Beverley - Version 21.11.23

' Launches selected text on DPTM - Busca el texto seleccionado en el Diccionario de términos médicos

alsoCopySubject = True

mySite = "https://dptm.es/dptm/?k="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

If alsoCopySubject = True Then Selection.Copy

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

Debug.Print mySite & mySubject

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

End Sub

**Sub RandomTextTypeMac()**

**'** Paul Beverley - Version 24.11.23

' Types out paragraphs of pseudo-Latin or English text (Mac version)

autoCorrect.ReplaceText = True

myText = InputBox("How many paras?" & vbCr & \_

"L = lorem, R = random", "MacroUpdater")

myText = LCase(Trim(myText))

If InStr(myText, "r") > 0 Then

myType = "=rand(num)"

Else

myType = "=lorem(num)"

End If

myText = Replace(myText, "r", "")

myText = Replace(myText, "l", "")

myType = Replace(myType, "num", myText)

Debug.Print myType

Selection.InsertAfter myType

Selection.Collapse wdCollapseEnd

End Sub

**Sub RandomTextType()**

**'** Paul Beverley - Version 24.11.23

' Types out paragraphs of pseudo-Latin or English text

autoCorrect.ReplaceText = True

myText = InputBox("How many paras?" & vbCr & \_

"L = lorem, R = random", "MacroUpdater")

myText = LCase(Trim(myText))

If InStr(myText, "r") > 0 Then

myType = "=rand(num)"

Else

myType = "=lorem(num)"

End If

myText = Replace(myText, "r", "")

myText = Replace(myText, "l", "")

myType = Replace(myType, "num", myText)

Debug.Print myType

Selection.InsertAfter myType

Selection.Collapse wdCollapseEnd

SendKeys "{ENTER}"

SendKeys "{NUMLOCK}"

End Sub

**Sub SearchHierarchically()**

**'** Paul Beverley - Version 24.11.23

' Shell macro: Searches hierarchically for a character

myChar = "z"

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

For Each myPar In rng.Paragraphs

If InStr(myPar.Range.Text, myChar) > 0 Then

For Each wd In myPar.Range.Words

If InStr(wd.Text, myChar) > 0 Then

For Each ch In wd.Characters

If ch = myChar Then

ch.Select

Beep

Exit Sub

End If

DoEvents

Next ch

End If

DoEvents

Next wd

End If

DoEvents

Next myPar

End Sub

**Sub PublisherSwitch()**

**'** Paul Beverley - Version 14.11.23

' Swaps "Location: Publisher Name" to "Publisher Name, Location"

Set rng = Selection.Range.Duplicate

If rng.Start <> rng.End Then GoTo otherBit

rng.Expand wdWord

myStart = rng.Start

rng.Expand wdParagraph

rng.Start = myStart

colonPos = InStr(rng, ": ")

rng.End = rng.Start + colonPos + 1

myPlace = ", " & Replace(rng, ": ", "")

rng.Delete

myStart = rng.Start

rng.Expand wdParagraph

rng.Start = myStart

rng.MoveEnd , -2

rng.Collapse wdCollapseEnd

rng.InsertAfter myPlace

rng.Collapse wdCollapseEnd

rng.Select

Exit Sub

otherBit:

Set rng2 = rng.Duplicate

rng2.Collapse wdCollapseEnd

rng2.Expand wdWord

rng2.Collapse wdCollapseEnd

myEnd = rng.End

rng.Collapse wdCollapseStart

rng.Expand wdWord

rng.Collapse wdCollapseStart

rng.End = myEnd

rng.Select

colonPos = InStr(rng, ": ")

rng.End = rng.Start + colonPos + 1

myPlace = ", " & Replace(rng.Text, ": ", "")

rng.Delete

rng2.InsertAfter myPlace

rng2.Collapse wdCollapseEnd

rng2.Select

End Sub

**Sub SwapNamesFullPoints()**

**'** Paul Beverley - Version 13.11.23

' Swap name and initials and add full points

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

rng.Collapse wdCollapseStart

myStart = rng.Start

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\([0-9]{4}\)"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

rng.Collapse wdCollapseStart

rng.Start = myStart

myEnd = rng.End

gotTwo = (InStr(rng, " and ") > 0)

myInits = Trim(rng.Words(3))

rng.Words(2) = " "

rng.Words(2) = ""

rng.Collapse wdCollapseStart

newInits = ""

For i = 1 To Len(myInits)

newInits = newInits & Mid(myInits, i, 1) & "."

DoEvents

Next i

rng.InsertBefore newInits & " "

If gotTwo Then

rng.End = myEnd

andPos = InStr(rng, " and ")

rng.MoveStart , andPos + 4

rng.Expand wdWord

rng.Collapse wdCollapseStart

rng.End = myEnd

rng.MoveEnd wdWord, 4

myInits = Trim(rng.Words(3))

rng.Words(2) = " "

rng.Words(2) = ""

rng.Collapse wdCollapseStart

newInits = ""

For i = 1 To Len(myInits)

newInits = newInits & Mid(myInits, i, 1) & "."

DoEvents

Next i

rng.InsertBefore newInits & " "

End If

DoEvents

rng.Collapse wdCollapseEnd

rng.Select

End Sub

**Sub ParagraphEndChecker()**

**'** Paul Beverley - Version 22.12.23

' Highlights the end of all possibly punctuation-less paragraphs

myMainHighlight = wdBrightGreen

mySoftColour = wdColorBlue

' or for no coloration inside tables

' mySoftColour = wdColorAutomatic

myOKChars = ".!?:…"

addLightColour = True

' addLightColour = False

myLightColour = wdGray25

' To not highlight all headings,

' ignore short "paragraphs" = headings.

minWords = 10

underlineLineFeeds = True

myScreenOff = True

Dim i As Long

Dim j As Long

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

If addLightColour = True Then

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myLightColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ".^p"

.Wrap = wdFindContinue

.Replacement.Highlight = True

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ".^p"

.Wrap = wdFindContinue

.Replacement.Highlight = True

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

End If

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ".^p"

.Wrap = wdFindContinue

.Replacement.Highlight = True

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

End If

Options.DefaultHighlightColorIndex = oldColour

End If

sps = "": For i = 1 To 20: sps = sps & " ": Next i

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If Selection.Start <> Selection.End Then

myResponse = MsgBox("Remove existing highlights?", vbQuestion \_

+ vbYesNoCancel, "ParagraphEndChecker")

If myResponse <> vbYes Then

Application.ScreenUpdating = True

Exit Sub

End If

Set rng = ActiveDocument.Content

For Each myPar In ActiveDocument.Paragraphs

rng.Start = myPar.Range.End - 3

rng.End = myPar.Range.End - 1

rng.Font.Color = wdColorAutomatic

rng.HighlightColorIndex = wdNoHighlight

DoEvents

Next myPar

Else

For myPass = 1 To 3

If myPass = 1 And ActiveDocument.Endnotes.Count = 0 Then \_

myPass = 2

If myPass = 2 And ActiveDocument.Footnotes.Count = 0 Then \_

myPass = 3

Select Case myPass

Case 1: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

i = rng.Paragraphs.Count

For Each myPar In rng.Paragraphs

Set testRng = myPar.Range

testRng.End = testRng.End - 1

If testRng.Font.Bold = False Then

If Right(testRng.Text, 1) = " " Then \_

testRng.End = testRng.End - 1

If Right(testRng.Text, 1) = " " Then \_

testRng.End = testRng.End - 1

If testRng.Words.Count > minWords + 2 Then

myLen = Len(testRng)

myTest = Right(testRng.Text, 3)

gottaPunc = False

For x = 3 To 1 Step -1

If InStr(myOKChars, Mid(myTest, x, 1)) > 0 Then

gottaPunc = True

puncPosn = x

Exit For

End If

DoEvents

Next x

testRng.Start = testRng.End - 2

If gottaPunc = False Then

If testRng.Information(wdWithInTable) = True Then

testRng.Font.Color = mySoftColour

Else

testRng.HighlightColorIndex = myMainHighlight

End If

Else

If addLightColour = True And gottaPunc = True Then

Set rng2 = rng.Duplicate

rng2.Start = rng2.End - 1

rng2.HighlightColorIndex = myLightColour

End If

End If

End If

i = i - 1

If i Mod 10 = 0 Then StatusBar = sps & i

DoEvents

End If

Next myPar

Next myPass

End If

If underlineLineFeeds = True Then

Options.DefaultHighlightColorIndex = myLightColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^?^?^11^?^?"

.Wrap = wdFindContinue

.Replacement.Font.Underline = True

.Replacement.Highlight = True

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

End With

Options.DefaultHighlightColorIndex = oldColour

End If

ActiveDocument.TrackRevisions = myTrack

Beep

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub ParticipleSentenceStartShow()**

**'** Paul Beverley - Version 04.01.24

' Underlines + highlights the participles that start sentences

myHighColour = wdYellow

' or myHighColour = 0

' for no highlight

myUnderline = True

For Each mySent In ActiveDocument.Sentences

myWd = Trim(mySent.Words(1))

If Len(myWd) > 4 Then

If Right(myWd, 3) = "ing" Then

If myHighColour > 0 Then \_

mySent.Words(1).HighlightColorIndex = myHighColour

If myUnderline = True Then \_

mySent.Words(1).Font.Underline = True

mySent.Words(1).Select

DoEvents

End If

End If

Next

End Sub

**Sub CommentCollectTabulated()**

**'** Paul Beverley - Version 15.01.24

' Collects comments into a table

sortOnColumn = 0

splitComment = True

doLandscape = True

maxHeading = "Heading 1"

s = ChrW(160)

s = s & s & s & s & s

myspaces = s & s & s & s & s & s & s & s

' If Word throws up and error 4605 about pasting

' increase the delay value to, say, 1000

myDelay = 500

myScreenOff = True

totCmnts = ActiveDocument.Comments.Count

Selection.HomeKey Unit:=wdStory

Selection.EndKey Unit:=wdStory

If totCmnts = 0 Then

MsgBox "There are no comments in this file!"

Exit Sub

End If

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

ReDim pageNo(totCmnts) As String

ReDim lineNo(totCmnts) As String

For i = 1 To totCmnts

Selection.GoTo What:=wdGoToComment, Count:=i

pageNo(i) = Trim(Str(Selection.Range.Information(wdActiveEndPageNumber)))

' pageNo(i) = Trim(Str(Selection.Range.Information(wdActiveEndAdjustedPageNumber)))

lineNo(i) = Trim(Str(Selection.Range.Information(wdFirstCharacterLineNumber)))

For j = 1 To myDelay

DoEvents

Next j

Next i

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.Copy

Documents.Add

For j = 1 To myDelay

DoEvents

Next j

Selection.Paste

totCmnts = ActiveDocument.Comments.Count

Dim cmnt As Word.Comment

ReDim lineNum(totCmnts) As Integer

ReDim pageNum(totCmnts) As Integer

maxHeadNo = Val(Right(maxHeading, 1))

hdStart = Left(maxHeading, InStr(maxHeading, " ") - 1)

For i = 1 To totCmnts

Selection.GoTo What:=wdGoToComment, Count:=i

pn = pageNo(i)

ln = lineNo(i)

Set cmnt = ActiveDocument.Comments(i)

inits = cmnt.Initial

scp = cmnt.Scope

itemNo = Trim(Str(i))

If splitComment = True And InStr(cmnt.Range, "|") = False Then

cmnt.Range.InsertBefore Text:="|"

End If

' Selection.GoTo what:=wdGoToComment, Count:=i

Set rng = ActiveDocument.Range(0, Selection.End)

paraNum = rng.Paragraphs.Count

For j = paraNum - 1 To 1 Step -1

myStyle = ActiveDocument.Paragraphs(j).Range.Style

If InStr(myStyle, hdStart) > 0 Then

hdLevel = Val(Right(myStyle, 1))

If hdLevel < maxHeadNo + 1 Then

hd = ActiveDocument.Paragraphs(j).Range.Text

Exit For

End If

End If

DoEvents

Next j

Debug.Print "Comment: ", totCmnts - i

StatusBar = "Comments to go: " & Str(totCmnts - i)

' BEFORE TEXT HERE <<<<<<<<<<<<<<<<<<<<<<<<

' Full version

' cmnt.Range.InsertBefore Text:=inits & itemNo & "|" & "p." & pn & \_

" l." & ln & "|" & scp & "|"

' Paul's own short version

cmnt.Range.InsertBefore Text:="yNLyp." & pn & "|" & scp

' cmnt.range.InsertBefore Text:="yNLyp." & pn & "|" & scp

' AFTER TEXT HERE <<<<<<<<<<<<<<<<<<<<<<<<<

cmnt.Range.InsertAfter Text:=""

Next i

ActiveDocument.StoryRanges(wdCommentsStory).Copy

' Replace whole text with just the (augmented) comments

Selection.WholeStory

For j = 1 To myDelay

DoEvents

Next j

Selection.Paste

Selection.WholeStory

' Remove hidden page references

Selection.Fields.Unlink

If doLandscape = True Then Selection.PageSetup.Orientation = wdOrientLandscape

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.Text = "^p"

.Replacement.Text = "zCRz"

.Wrap = wdFindContinue

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

For i = 1 To 2

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

' Go and find the first occurrence

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zCRz"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

Do While rng.Find.Found = True

rng2.Start = rng.End

rng2.End = rng.End + 5

If rng2 = "yNLy" Then rng.Delete

rng.Collapse wdCollapseEnd

' Go and find the next occurrence (if there is one)

rng.Find.Execute

Loop

Next i

' Remove the very first new line marker

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "yNLy"

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

' Change other new line markers into new lines

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "yNLy"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Replacement.Highlight = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.EndKey Unit:=wdStory

Selection.MoveStart , -4

If Selection = "zCRz" Then Selection.Delete

Selection.EndKey Unit:=wdStory

Selection.MoveStart , -4

If Selection = "zCRz" Then Selection.Delete

Selection.HomeKey Unit:=wdStory

' HEADING LINE HERE<<<<<<<<<<<<<<<<<<<<<<<<

Selection.TypeText Text:="|Context|Comment/query|Author" & ChrW(160) & \_

"response" & myspaces & vbCr

ActiveDocument.Paragraphs(1).Range.Bold = True

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "|"

.Wrap = wdFindContinue

.Replacement.Text = "^t"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Author queries " & ChrW(8211) & \_

" Chapter " & vbCr & vbCr

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

ActiveDocument.Paragraphs(3).Range.Select

Selection.Collapse wdCollapseEnd

Selection.End = ActiveDocument.Range.End

If sortOnColumn > 0 Then Selection.Sort ExcludeHeader:=False, \_

FieldNumber:="Column " & Trim(Str(sortOnColumn)), \_

SortFieldType:=wdSortFieldAlphanumeric, \_

SortOrder:=wdSortOrderAscending, \_

Separator:=wdSortSeparateByTabs, SortColumn:=False, \_

CaseSensitive:=True

ActiveDocument.Paragraphs(2).Range.Select

Selection.Collapse wdCollapseEnd

Selection.End = ActiveDocument.Range.End

Selection.ConvertToTable Separator:=wdSeparateByTabs

Selection.Tables(1).AutoFitBehavior (wdAutoFitContent)

' Restore the new lines that are inside the comments

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "zCRz"

.Replacement.Text = "^p"

.Wrap = wdFindContinue

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Beep

ActiveDocument.TrackRevisions = myTrack

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub FindResetParameters()**

**'** Paul Beverley - Version 15.01.24

' Does a dummy find to set all the parameters to default

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "youwon'tfindthis!"

.Replacement.Text = ""

.Forward = True

.Wrap = wdFindContinue

.Format = False

.MatchCase = False

.MatchWholeWord = False

.MatchWildcards = False

.MatchSoundsLike = False

.MatchAllWordForms = False

.Execute

.Text = ""

.Execute

End With

End Sub

**Sub HighlightOffIncNotes()**

**'** Paul Beverley - Version 24.01.24

' Removes all highlights and/or colour from text and notes

doColourToo = True

nowTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

ActiveDocument.Content.HighlightColorIndex = wdNoHighlight

If doColourToo = True Then

ActiveDocument.Content.Font.Color = wdColorAutomatic

End If

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

rng.HighlightColorIndex = wdNoHighlight

If doColourToo = True Then rng.Font.Color = wdColorAutomatic

End If

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

rng.HighlightColorIndex = wdNoHighlight

If doColourToo = True Then rng.Font.Color = wdColorAutomatic

End If

ActiveDocument.TrackRevisions = nowTrack

End Sub

**Sub PlurAlyse()**

' Paul Beverley - Version 17.06.23

' Creates a frequency list of single/plural pairs

minLength = 3

Dim myResult As String

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox("Run PlurAlyse ?", vbQuestion \_

+ vbYesNoCancel, "PlurAlyse")

If myResponse <> vbYes Then Exit Sub

End If

Set testDoc = ActiveDocument

strttime = Timer

Set rng = ActiveDocument.Content

Documents.Add

Selection.Text = rng.Text

Selection.HomeKey Unit:=wdStory

Application.ScreenUpdating = False

On Error GoTo ReportIt

Set rng = ActiveDocument.Content

rng.Text = " " & vbCr & vbCr & LCase(rng.Text)

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

Set rng2 = ActiveDocument.Content

wasLength = rng2.End

myWild = "<[a-zA-Z]@ies>"

For i = 3 To 1 Step -1

If i = 2 Then myWild = "<[a-zA-Z]@es>"

If i = 1 Then myWild = "<[a-zA-Z]@s>"

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myWild

.Font.StrikeThrough = False

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

End With

Do While rng.Find.Found = True

endWas = rng.End

myLong = rng.Text

rng.MoveEnd , -i

If i = 3 Then

myShort = rng.Text & "y"

Else

myShort = rng.Text

End If

Selection.MoveRight , 1

' Does the singular occur?...

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myShort

.MatchWholeWord = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

If rng2.Find.Found = True And Len(rng2.Text) > (minLength - 1) Then

rng2.Collapse wdCollapseEnd

' ... if so, then count occurrences

DoEvents

Application.ScreenUpdating = True

ActiveDocument.Paragraphs(1).Range.Text = myShort & vbCr

ActiveDocument.Paragraphs(1).Range.Select

Selection.Collapse wdCollapseStart

DoEvents

Application.ScreenUpdating = False

ActiveDocument.Paragraphs(1).Range.Text = vbCr

DoEvents

wasLength = ActiveDocument.Content.End

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myShort

.Wrap = wdFindContinue

.MatchWholeWord = True

.Replacement.Text = "^&!"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

numSingular = ActiveDocument.Content.End - wasLength

WordBasic.EditUndo

myResult = myResult & myShort & vbTab & Trim(Str(numSingular))

DoEvents

wasLength = ActiveDocument.Content.End

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myLong

.Wrap = wdFindContinue

.MatchWholeWord = True

.Replacement.Text = "^&!"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

numPlural = ActiveDocument.Content.End - wasLength

WordBasic.EditUndo

myResult = myResult & vbTab & myLong & vbTab & \_

Trim(Str(numPlural)) & vbCr

End If

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myLong

.Wrap = wdFindContinue

.MatchWholeWord = True

.Replacement.Text = "^&"

.Replacement.Font.StrikeThrough = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

rng.End = endWas

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Next i

Selection.WholeStory

Selection.Style = ActiveDocument.Styles(wdStyleNormal)

Selection.TypeText Text:=myResult

Set rng = ActiveDocument.Content

rng.Sort

Selection.HomeKey Unit:=wdStory

Selection.MoveEnd , 2

Selection.TypeText "Plurals use"

startTable = Selection.End + 1

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

Selection.Start = startTable

Selection.End = ActiveDocument.Range.End

Selection.ConvertToTable Separator:=wdSeparateByTabs

timNow = Timer

Application.ScreenUpdating = True

If doingSeveralMacros = False Then

timGone = timNow - strttime

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

m = Int(timGone / 60)

s = Int(timGone) - m \* 60

timeAll = "Time: " & Trim(Str(m)) & " m " & \_

Trim(Str(s)) & " s"

Selection.HomeKey Unit:=wdStory

numPairs = ActiveDocument.Tables(1).Rows.count

MsgBox "Items: " & Trim(Str(numPairs)) & vbCr & vbCr \_

& timeAll

Else

FUT.Activate

End If

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub IndexListItems()**

**'** Paul Beverley - Version 01.02.24

' Finds page numbers of all the words/phrases given in a list

doWholeWordsOnly = False

doMatchCase = False

firstGap = " " & ChrW(8211) & " "

myLink = ", "

oldColour = Options.DefaultHighlightColorIndex

Set myDoc = ActiveDocument

gottaList = False

For Each myWnd In Application.Windows

thisName = myWnd.Document.Name

If InStr(LCase(thisName), "list") > 0 Then

myWnd.Document.Activate

myResponse = MsgBox("Is this your list?" & vbCr & vbCr \_

& ">>> " & thisName & " <<<", \_

vbQuestion + vbYesNoCancel, "IndexListItems")

If myResponse = vbCancel Then

Beep

Exit Sub

End If

If myResponse = vbYes Then

gottaList = True

Exit For

End If

End If

Next myWnd

If gottaList = False Then

Beep

MsgBox "Can't find a word list."

Exit Sub

End If

If myDoc.FullName = myWnd.Document.FullName Then

Beep

MsgBox "Please place the cursor in the text to be" & vbCr & "indexed and rerun the macro."

Exit Sub

End If

Set myList = myWnd.Document

Documents.Add

Set rngIndex = ActiveDocument.Content

rngIndex.Text = myList.Content.Text

With rngIndex.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^11"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^13[0-9]{1,}^13^13"

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = " . . . [0-9]{1,}"

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

For i = 1 To rngIndex.Paragraphs.Count

Set myItem = rngIndex.Paragraphs(i).Range

myText = Replace(myItem.Text, vbCr, "")

myFind = myText

If Left(myText, 1) = "#" Then

Beep

ActiveDocument.TrackRevisions = nowTrack

Exit Sub

End If

If Len(myText) > 1 And Left(myText, 1) <> "|" Then

thisHighlightColour = myList.Paragraphs(i).Range.Characters(1).HighlightColorIndex

Options.DefaultHighlightColorIndex = thisHighlightColour

thisTextColour = myList.Paragraphs(i).Range.Characters(1).Font.Color

' if item is coloured/highlighted do that in the document

If thisHighlightColour <> 0 Or thisTextColour <> 0 Then

Set rngDoc = myDoc.Content

With rngDoc.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = myText

.Replacement.Text = "^&"

If thisHighlightColour > 0 Then .Replacement.Highlight = True

If thisTextColour > 0 Then .Replacement.Font.Color = thisTextColour

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = doWholeWordsOnly

.Execute Replace:=wdReplaceAll

End With

If myDoc.Footnotes.Count > 0 Then

Set rngNts = myDoc.StoryRanges(wdFootnotesStory)

With rngNts.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = myText

.Replacement.Text = "^&"

If thisHighlightColour <> 0 Then .Replacement.Highlight = True

If thisTextColour <> 0 Then .Replacement.Font.Color = thisTextColour

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = doWholeWordsOnly

.Execute Replace:=wdReplaceAll

End With

End If

If myDoc.Endnotes.Count > 0 Then

Set rngNts = myDoc.StoryRanges(wdEndnotesStory)

With rngNts.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = myText

.Replacement.Text = "^&"

If thisHighlightColour > 0 Then .Replacement.Highlight = True

If thisTextColour > 0 Then .Replacement.Font.Color = thisTextColour

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = doWholeWordsOnly

.Execute Replace:=wdReplaceAll

End With

End If

End If

Set rngDoc = myDoc.Content

With rngDoc.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = myFind

.Replacement.Text = "^&"

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = doWholeWordsOnly

.Execute

myText = myText & firstGap

Do While .Found = True

myPg = Trim(Str(rngDoc.Information(wdActiveEndAdjustedPageNumber)))

myText = myText & myPg & myLink

rngDoc.Collapse wdCollapseEnd

rngDoc.Find.Execute

DoEvents

Loop

End With

If myDoc.Footnotes.Count > 0 Then

Set rngNts = myDoc.StoryRanges(wdFootnotesStory)

With rngNts.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = myFind

.Replacement.Text = "^&"

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = doWholeWordsOnly

.Execute

If .Found = True Then myText = myText & " Notes on pp: "

Do While .Found = True

myPg = Trim(Str(rngNts.Information(wdActiveEndAdjustedPageNumber)))

myText = myText & myPg & myLink

rngNts.Collapse wdCollapseEnd

rngNts.Find.Execute

DoEvents

Loop

End With

End If

If myDoc.Endnotes.Count > 0 Then

Set rngNts = myDoc.StoryRanges(wdEndnotesStory)

With rngNts.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = myFind

.Replacement.Text = "^&"

.MatchCase = doMatchCase

.MatchWildcards = False

.MatchWholeWord = doWholeWordsOnly

.Execute

If .Found = True Then myText = myText & " Notes on pp: "

Do While .Found = True

myPg = Trim(Str(rngNts.Information(wdActiveEndAdjustedPageNumber)))

myText = myText & myPg & myLink

rngNts.Collapse wdCollapseEnd

rngNts.Find.Execute

DoEvents

Loop

End With

End If

myText = myText & vbCr

myText = Replace(myText, myLink & vbCr, vbCr)

myItem.Text = myText

End If

DoEvents

If i Mod 3 = 0 Then

rngIndex.Paragraphs(i).Range.Select

Selection.Collapse wdCollapseEnd

End If

Next i

Options.DefaultHighlightColorIndex = oldColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = "(<[0-9]{1,}), \1"

.Replacement.Text = "\1"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Execute Replace:=wdReplaceAll

.Execute Replace:=wdReplaceAll

.Execute Replace:=wdReplaceAll

.Text = ", "

.Replacement.Text = " "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub PunctuationToApostrophe()**

**'** Paul Beverley - Version 01.02.24

' Changes next quote mark to an apostrophe

trackIt = True

newChar = ChrW(8217)

searchChars = Chr(34) & Chr(39) & ChrW(8216) \_

& ChrW(8220) & ChrW(8221) & ChrW(8249) & ChrW(8250) \_

& ChrW(8222) & ChrW(8218) & ChrW(171) & ChrW(187) & ChrW(96) \_

& ChrW(8242) & ChrW(8243)

Set rng = Selection.Range.Duplicate

For i = 1 To 1000

rng.MoveEnd , 1

If InStr(searchChars, Right(rng, 1)) > 0 Then

rng.Start = rng.End - 1

gotChar = True

Exit For

End If

Next i

If gotChar = False Then

Beep

Exit Sub

End If

ch = Right(rng, 1)

Selection.Collapse wdCollapseStart

myTrack = ActiveDocument.TrackRevisions

If trackIt = False Then ActiveDocument.TrackRevisions = False

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ch

.Replacement.Text = ""

.Execute

End With

Selection.TypeText Text:=newChar

Selection.Collapse wdCollapseEnd

ActiveDocument.TrackRevisions = myTrack

End Sub

**Sub HighlightTrackedTextAuthor()**

**'** Paul Beverley - Version 02.02.24

' Highlights all tracked text by a specific author

' Put some or all of the author's name here

auName = "John"

myColour = wdYellow

ActiveDocument.TrackRevisions = False

For Each rev In ActiveDocument.Range.Revisions

Set rng = rev.Range

rng.Start = rng.Start - 1

rng.End = rng.End + 1

Debug.Print rev.Author

If InStr(rev.Author, auName) > 0 Then

rng.HighlightColorIndex = myColour

End If

DoEvents

i = i + 1

If i Mod 6 = 1 Then

rng.Select

rng.Collapse wdCollapseEnd

End If

Next rev

If ActiveDocument.Footnotes.Count > 0 Then

For Each rev In ActiveDocument.StoryRanges(wdFootnotesStory).Revisions

Set rng = rev.Range

rng.Start = rng.Start - 1

rng.End = rng.End + 1

Debug.Print rev.Author

If InStr(rev.Author, auName) > 0 Then

rng.HighlightColorIndex = myColour

End If

DoEvents

i = i + 1

If i Mod 6 = 1 Then

rng.Select

rng.Collapse wdCollapseEnd

End If

Next rev

End If

Exit Sub

If ActiveDocument.Endnotes.Count > 0 Then

For Each rev In ActiveDocument.StoryRanges(wdEndnotesStory).Revisions

rng.Start = rng.Start - 1

rng.End = rng.End + 1

Debug.Print rev.Author

If InStr(rev.Author, auName) > 0 Then

rng.HighlightColorIndex = myColour

End If

DoEvents

i = i + 1

If i Mod 6 = 1 Then

rng.Select

rng.Collapse wdCollapseEnd

End If

Next rev

End If

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub InitialPullBack()**

**'** Paul Beverley - Version 03.02.24

' Pulls author's trailing initial(s) (e.g. B. or C.D.) in front of surname

Set rng = Selection.Range.Duplicate

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z.\-]{2,}"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute

DoEvents

End With

myInits = rng & " "

Set rngNext = rng.Duplicate

rngNext.Collapse wdCollapseEnd

rngNext.MoveStart , 3

rng.MoveStart , -2

rng.Delete

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

rng.InsertBefore Text:=myInits

rngNext.Select

End Sub

**Sub BoldItalicEtcToStyle()**

**'** Paul Beverley - Version 18.02.24

' Changes bold, italic, etc to character styles

doHighlight = True

myHiColour = wdYellow

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myHiColour

Set rng = ActiveDocument.Content

With rng.Find

' Strike through all foot/endnote citations

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^2"

.Wrap = wdFindContinue

.Replacement.Font.StrikeThrough = True

.Replacement.Text = "^&"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

' Make the style changes

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Font.Superscript = True

.Font.StrikeThrough = False

.Forward = True

.Replacement.Font.Superscript = False

.Replacement.Style = ActiveDocument.Styles("Superscript")

.Replacement.Text = "^&"

.MatchWildcards = False

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.StrikeThrough = False

.Font.Subscript = True

.Replacement.Font.Subscript = False

.Replacement.Style = ActiveDocument.Styles("Subscript")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.StrikeThrough = False

.Font.Italic = True

.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Style = ActiveDocument.Styles("Italic")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Bold = True

.Font.StrikeThrough = False

.Font.Italic = False

.Replacement.Font.Bold = False

.Replacement.Style = ActiveDocument.Styles("Bold")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Underline = True

.Font.StrikeThrough = False

.Replacement.Font.Underline = False

.Replacement.Style = ActiveDocument.Styles("Underline")

.Replacement.Highlight = True

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Bold = True

.Font.Italic = True

.Font.StrikeThrough = False

.Replacement.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Style = ActiveDocument.Styles("Bold Italic")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

rng.Font.StrikeThrough = False

If ActiveDocument.Footnotes.Count > 0 Then

For Each nt In ActiveDocument.Footnotes

' Strike through all footnote numbers

Set rng = nt.Range

rng.Collapse wdCollapseStart

rng.MoveStart , -2

rng.Font.StrikeThrough = True

rng.Collapse wdCollapseEnd

Next nt

For Each nt In ActiveDocument.Footnotes

' Now make the style changes

With nt.Range.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Font.Superscript = True

.Font.StrikeThrough = False

.Forward = True

.Replacement.Font.Superscript = False

.Replacement.Style = ActiveDocument.Styles("Superscript")

.Replacement.Text = "^&"

.MatchWildcards = False

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.StrikeThrough = False

.Font.Subscript = True

.Replacement.Font.Subscript = False

.Replacement.Style = ActiveDocument.Styles("Subscript")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.StrikeThrough = False

.Font.Italic = True

.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Style = ActiveDocument.Styles("Italic")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Bold = True

.Font.StrikeThrough = False

.Font.Italic = False

.Replacement.Font.Bold = False

.Replacement.Style = ActiveDocument.Styles("Bold")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Underline = True

.Font.StrikeThrough = False

.Replacement.Font.Underline = False

.Replacement.Style = ActiveDocument.Styles("Underline")

.Replacement.Highlight = True

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Bold = True

.Font.Italic = True

.Font.StrikeThrough = False

.Replacement.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Style = ActiveDocument.Styles("Bold Italic")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Next nt

End If

If ActiveDocument.Endnotes.Count > 0 Then

For Each nt In ActiveDocument.Endnotes

' Strike through all footnote numbers

Set rng = nt.Range

rng.Collapse wdCollapseStart

rng.MoveStart , -2

rng.Font.StrikeThrough = True

rng.Collapse wdCollapseEnd

Next nt

For Each nt In ActiveDocument.Endnotes

' Now make the style changes

With nt.Range.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Font.Superscript = True

.Font.StrikeThrough = False

.Forward = True

.Replacement.Font.Superscript = False

.Replacement.Style = ActiveDocument.Styles("Superscript")

.Replacement.Text = "^&"

.MatchWildcards = False

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.StrikeThrough = False

.Font.Subscript = True

.Replacement.Font.Subscript = False

.Replacement.Style = ActiveDocument.Styles("Subscript")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.StrikeThrough = False

.Font.Italic = True

.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Style = ActiveDocument.Styles("Italic")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Bold = True

.Font.StrikeThrough = False

.Font.Italic = False

.Replacement.Font.Bold = False

.Replacement.Style = ActiveDocument.Styles("Bold")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Underline = True

.Font.StrikeThrough = False

.Replacement.Font.Underline = False

.Replacement.Style = ActiveDocument.Styles("Underline")

.Replacement.Highlight = True

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Bold = True

.Font.Italic = True

.Font.StrikeThrough = False

.Replacement.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Style = ActiveDocument.Styles("Bold Italic")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Next nt

End If

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

rng.Font.StrikeThrough = False

End If

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

rng.Font.StrikeThrough = False

End If

Options.DefaultHighlightColorIndex = oldColour

Beep

End Sub

**Sub CopyToSwitchList()**

**'** Paul Beverley - Version 04.12.23

' Copies selected text into a switch list file for MultiSwitch

keyWord = "switchlist"

' copyWholePara = True

copyWholePara = False

myHighlightColour = wdColorBlack

' To add highlight, use:

' myHighlightColour = wdYellow

alwaysCopyAtEnd = False

Set thisDoc = ActiveDocument

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.MoveEnd , -1

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

rng.Select

End If

Set sourceText = Selection.Range

For Each myDoc In Application.Documents

thisName = myDoc.Name

nm = LCase(thisName)

gottaList = False

If InStr(nm, LCase(keyWord)) > 0 Then

myDoc.Activate

gottaList = True

Exit For

End If

Next myDoc

If gottaList = False Then

Beep

MsgBox "Can't find a list."

Exit Sub

End If

If alwaysCopyAtEnd = True Then

Selection.EndKey Unit:=wdStory

Else

hereNow = Selection.Start

Selection.Expand wdParagraph

If Selection.Start = hereNow Then Selection.Collapse wdCollapseStart

If Len(Selection) = 1 Then

Selection.Collapse wdCollapseStart

Else

Selection.Collapse wdCollapseEnd

End If

End If

Selection.InsertAfter sourceText.Text & vbCr & vbCr & vbCr

Selection.Collapse wdCollapseEnd

Selection.MoveUp , 2

If myHighlightColour > 0 Then

sourceText.HighlightColorIndex = myHighlightColour

Selection.Collapse wdCollapseEnd

End If

End Sub

**Sub FindFetch()**

**'** Paul Beverley - Version 08.08.22

' Shell macro for making your own macro:

' Finds something specific and does things to each one

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Fetch"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

myC = ""

Do While rng.Find.Found = True

rng.Expand wdWord

myC = myC & Trim(rng.Text) & vbCr

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Documents.Add

Selection.TypeText Text:=myC

MsgBox "Changed: " & myCount

End Sub

**Sub FindHashAndStyleNext()**

**'** Paul Beverley - Version 19.01.23

' Finds a hash and changes the style of the following paragraph

myStyle = "Heading 1"

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "#"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

If myCount Mod 20 = 0 Then rng.Select

rng.Style = myStyle

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

MsgBox "Changed: " & myCount

End Sub

**Sub FindMissingPageNumbers()**

**'** Paul Beverley - Version 28.09.23

' Clears trailing spaces, then highlights refs without page numbers

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[ ]{1,}^13"

.Replacement.Text = "^p"

.Forward = True

.Wrap = wdFindContinue

.Format = True

.MatchCase = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[0-9]{1,}:[0-9]{1,}^13"

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchCase = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

End Sub

**Sub IndexPageNumberSwap()**

**'** Paul Beverley - Version 21.11.23

' Swaps the page number at the cursor with the next number, e.g. 3,6,4,8,10

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

Set rng2 = Selection.Range.Duplicate

rng2.Expand wdParagraph

rng.End = rng2.End

myFirstNum = rng.Words(1)

rng.Words(1) = rng.Words(3)

rng.Words(3) = myFirstNum

End Sub

**Sub IndexPageOrderChecker()**

**'** Paul Beverley - Version 21.11.23

' Seeks paragraphs that have number list in wrong order

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

Do While Len(rng) > 5

myLine = Replace(rng.Text, " ", "")

myNum = Split(rng, ",")

prevNum = 0

For i = 0 To UBound(myNum)

Debug.Print myNum(i)

nowNum = Val(myNum(i))

If nowNum < prevNum Then

Beep

rng.Select

Exit Sub

End If

prevNum = nowNum

Next i

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

DoEvents

Loop

Beep

rng.Collapse wdCollapseEnd

rng.Select

End Sub

**Sub LinkAdd()**

**'** Paul Beverley - Version 25.01.24

' Selects the current word(s) and adds a link from your input

Set rng = Selection.Range.Duplicate

myEnd = rng.End

rng.Collapse wdCollapseStart

rng.Expand wdWord

myStart = rng.Start

rng.Select

rng.End = myEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

rng.MoveEnd , 1

rng.Select

If Right(rng, 1) <> "/" Then rng.MoveEnd , -1

rng.Select

Selection.Start = myStart

myPrompt = "Type or paste in your URL"

myText = InputBox(myPrompt, "LinkAdd", Selection.Text)

ActiveDocument.Hyperlinks.Add Selection.Range, myText

End Sub

**Sub LinkDelete()**

**'** Paul Beverley - Version 25.01.24

' Deletes (unlinks) the current link

Set rng = ActiveDocument.Range(0, Selection.End)

linkNum = rng.Hyperlinks.Count

ActiveDocument.Hyperlinks(linkNum).Delete

End Sub

**Sub LinkCopy()**

**'** Paul Beverley - Version 25.01.24

' Copies the URL of the current link

Set rng = ActiveDocument.Range(0, Selection.End)

linkNum = rng.Hyperlinks.Count

Set rng = ActiveDocument.Hyperlinks(linkNum).Range

Selection.InsertAfter Text:=ActiveDocument.Hyperlinks(linkNum).Address

Selection.Copy

WordBasic.EditUndo

rng.Select

Selection.Collapse wdCollapseEnd

End Sub

**Sub ListAllCitations()**

**'** Paul Beverley - Version 28.12.23

' Alphabetically lists all fig/table/box/flowchart citations

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.Text = LCase(rngOld.Text)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Text = "<fig[eru :.\-0-9]{2,}"

.Replacement.Font.Underline = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "<flow[achrt :.\-0-9]{2,}"

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "<box[ :.\-0-9]{2,}"

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "<table[ :.\-0-9]{2,}"

.Execute Replace:=wdReplaceAll

DoEvents

.Text = ""

.Replacement.Text = "^p"

.Font.Underline = False

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Text = "([a-z])[:.\-]"

.Replacement.Text = "\1 "

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Text = " "

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[. ]{1,}^13"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

rng.Sort

rng.Font.Underline = False

rng.InsertAfter Text:=vbCr

rng.End = 2

rng.Delete

Selection.HomeKey Unit:=wdStory

End Sub

**Sub SpacedFontFinder()**

**'** Paul Beverley - Version 19.01.23

' Finds and highlights/underlines any text with odd spacing

myHiColour = wdYellow

' You can do it per paragraph, which is quicker...

For Each pa In ActiveDocument.Content.Paragraphs

If pa.Range.Font.Spacing <> 0 Then

pa.Range.Underline = True

End If

DoEvents

Next pa

' Or per word, in case some parts of a para are spaced

' and some not

For Each wd In ActiveDocument.Content.Words

If wd.Font.Spacing <> 0 Then

wd.HighlightColorIndex = myHiColour

End If

DoEvents

Next wd

End Sub

**Sub TrackChangeCountWords()**

**'** Paul Beverley - Version 18.01.24

' Checks the track changes to count words/characters added/subtracted

CR = vbCr

CR2 = CR & CR

For Each rev In ActiveDocument.Revisions

Select Case rev.Type

Case wdRevisionInsert

myAddChars = myAddChars + Len(rev.Range.Text)

myAddWords = myAddWords + rev.Range.Words.Count

Case wdRevisionDelete

myDeletesChars = myDeletesChars + Len(rev.Range.Text)

myDeletesWords = myDeletesWords + rev.Range.Words.Count

End Select

i = i + 1

If i Mod 10 = 0 Then DoEvents

Next rev

myResult = "Insertions" & CR

myResult = myResult & " Words: " & myAddWords & CR

myResult = myResult & " Characters: " & myAddChars & CR2

myResult = myResult & "Deletions" & CR

myResult = myResult & " Words: " & myDeletesWords & CR

myResult = myResult & " Characters: " & myDeletesChars & CR

MsgBox myResult

End Sub

**Sub NTaddressCopy()**

**'** Paul Beverley - Version 08.02.24

' Reports the address of the folder holding the Normal template

Documents.Add

Selection.InsertAfter NormalTemplate.Path

Selection.Copy

End Sub

**Sub OEDFetch()**

**'** Paul Beverley - Version 10.02.24

' Launches selected text on OED dictionary site

alsoCopySubject = False

mySite = "https://www.oed.com/search/dictionary/?scope=Entries&q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

If alsoCopySubject = True Then Selection.Copy

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

Debug.Print mySite & mySubject

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub PunctuationTrackAcceptAll()**

**'** Paul Beverley - Version 16.02.24

' Accepts all track changes that involve punctuation

For Each rv In ActiveDocument.Revisions

myText = rv.Range.Text

If rv.Type < 3 Then

If InStr(myText, ".") + InStr(myText, ",") + \_

InStr(myText, ";") + InStr(myText, ":") + \_

InStr(myText, ChrW(8216)) + InStr(myText, ChrW(8217)) + \_

InStr(myText, ChrW(8220)) + InStr(myText, ChrW(8221)) + \_

InStr(myText, "!") + InStr(myText, "?") > 0 Then

i = i + 1

If i Mod 20 = 0 Then rv.Range.Select

rv.Accept

DoEvents

End If

End If

Next rv

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub CopyToListAlphabetic()**

**'** Paul Beverley - Version 08.12.24

' Copies selected text into an alphabetic list file

beepIfAlreadyListed = True

' To locate list file (not case sensitive)

' keyWord = "queries"

' keyWord = "list"

keyWord = "sheet"

' Exact capitalisation(i.e. case sensitive)

followHeading = "Word list"

wordsToAvoid = "switch"

' wordsToAvoid = "FRedit,switch"

' copyWholePara = True

copyWholePara = False

includeFormatting = False

' includeFormatting = True

goBackToSource = True

Dim sourceText As Range, tgt As Range

Set thisDoc = ActiveDocument

wds = Split("," & LCase(wordsToAvoid), ",")

If Selection.start = Selection.End Then

If Selection = vbCr Then Selection.MoveLeft , 1

If copyWholePara = True Then

Selection.Expand wdParagraph

Else

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

rng.MoveEnd wdWord, 1

chkWd = rng.Words.Last

If chkWd = "-" Then

rng.MoveEnd wdWord, 2

chkWd = rng.Words.Last

If chkWd = "-" Then

rng.MoveEnd wdWord, 1

Else

rng.MoveEnd wdWord, -1

End If

Else

rng.MoveEnd wdWord, -1

End If

DoEvents

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

rng.Select

End If

Else

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.MoveEnd , -1

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.start = Selection.start

rng.Select

End If

Set sourceText = Selection.Range.Duplicate

myText = sourceText.Text

Selection.Collapse wdCollapseEnd

gottaList = False

For Each myDoc In Application.Documents

thisName = myDoc.Name

nm = LCase(thisName)

gottaList = False

If InStr(nm, LCase(keyWord)) > 0 Then gottaList = True

For i = 1 To UBound(wds)

If InStr(nm, wds(i)) > 0 Then gottaList = False

Next i

If gottaList = True Then Exit For

Next myDoc

If gottaList = False Then

Beep

MsgBox "Can't find a list."

Exit Sub

End If

' Decide where to put the item

Set rng = myDoc.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = followHeading

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchCase = True

.MatchWildcards = False

.Execute

DoEvents

End With

rng.Expand wdParagraph

listStart = rng.End

If rng.Find.Found = False Then

Beep

MsgBox "Can't find heading: """ & followHeading & """"

rng.start = 0

rng.End = 0

listStart = 0

Selection.HomeKey Unit:=wdStory

End If

rng.Collapse wdCollapseStart

' Check if it's already in the list

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p" & myText & "^p"

.Wrap = wdFindStop

.Execute

DoEvents

End With

If rng.Find.Found = True Then

If beepIfAlreadyListed = True Then Beep

If goBackToSource = False Then myDoc.Activate

Exit Sub

End If

rng.start = listStart

rng.End = myDoc.Content.End

If LCase(rng) <> UCase(rng) Then

For i = 1 To rng.Paragraphs.Count

myNextItem = Replace(rng.Paragraphs(i).Range.Text, vbCr, "")

If LCase(myNextItem) > LCase(myText) Or \_

LCase(myNextItem) = UCase(myNextItem) Then

Set rng = rng.Paragraphs(i).Range.Duplicate

Exit For

End If

Next i

End If

rng.Collapse wdCollapseStart

' Paste it in

If includeFormatting = True Then

rng.FormattedText = sourceText.FormattedText

If InStr(myText, vbCr) = 0 Then

rng.InsertAfter vbCr

End If

Else

rng.InsertAfter myText

If InStr(myText, vbCr) = 0 Then rng.InsertAfter vbCr

End If

rng.Revisions.AcceptAll

If goBackToSource = False Then myDoc.Activate

End Sub

**Sub SpellingCorrect()**

**'** Paul Beverley - Version 01.11.24

' Corrects spelling and wrong capitalisation

elistNameContains = "elist"

useEqualAsDelete = True

Selection.Collapse wdCollapseEnd

wasStart = Selection.start

Set rng = Selection.Range.Duplicate

myScope = 2

rng.MoveStart wdWord, -2

equalPos = InStr(rng, "=")

If equalPos > 0 And useEqualAsDelete = True Then

rng.End = rng.start + equalPos + 2

rng.start = rng.End - 2

rng.Delete

rng.Select

Beep

Exit Sub

End If

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

If LCase(rng) = UCase(rng) Then

If Len(rng) > 2 Then

' it's a number, give up

Beep

rng.Select

Exit Sub

Else

rng.Collapse wdCollapseStart

rng.MoveEnd , -2

rng.Expand wdWord

End If

End If

' No non-alpha characters at the end of the word

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

' Check capitalisation, e.g. BEverley

If Len(rng) > 2 Then

initChar = rng.Characters(1)

scndChar = rng.Characters(2)

lastChar = rng.Characters.Last

doBeep = True

If UCase(initChar) = initChar And UCase(scndChar) = scndChar \_

And UCase(lastChar) <> lastChar Then

doBeep = False

rng.Characters(2) = LCase(scndChar)

End If

End If

If Right(rng, 1) = vbCr Then rng.MoveEnd , -1

myWord = rng

' Check spelling language

' (check only first character, in case of split language)

langName = Languages(rng.Characters(1).LanguageID).NameLocal

' Check spelling

spellOK = Application.CheckSpelling(myWord, MainDictionary:=langName)

If Not (spellOK) Then

For Each myDoc In Documents

If InStr(LCase(myDoc.Name), elistNameContains) > 0 Then

If InStr(myDoc.Content, vbCr & myWord & vbCr) > 0 Then

spellOK = True

Exit For

End If

End If

DoEvents

Next myDoc

End If

If Not (spellOK) Then

Set suggList = Application.GetSpellingSuggestions(myWord, \_

MainDictionary:=langName)

If suggList.Count > 0 Then

newWord = suggList.Item(1).Name

If suggList.Count > 1 And LCase(newWord) = myWord \_

Then newWord = suggList.Item(2).Name

Else

newWord = myWord

End If

If newWord <> myWord Then

rng.Text = newWord

Else

spellOK = Application.CheckSpelling(myWord, \_

MainDictionary:=langName)

If spellOK = False Then

If newWord = myWord Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 1

Beep

If langName = "English (United States)" \_

And Application.CheckSpelling(myWord, \_

MainDictionary:="English (United Kingdom)") Then

MsgBox ("Word's spellchecker is playing sillies!")

Exit Sub

Else

MsgBox ("Word offers no suggestion")

End If

End If

End If

End If

Else

If doBeep Then Beep

End If

rng.start = wasStart - 2

rng.Expand wdWord

apoPos = InStr(rng, "'")

If apoPos > 0 Then rng.Characters(apoPos) = ChrW(8217)

rng.Collapse wdCollapseEnd

rng.Select

End Sub

**Sub CommentSuggestText()**

**'** Paul Beverley - Version 26.02.24

' Copies selected text to make a comment

preText = "Suggestion: "

doBold = True

If Selection.Start = Selection.End Then

Selection.Expand wdSentence

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

Selection.Copy

Dim myComment As Comment

Set myComment = Selection.Comments.Add(Range:=Selection.Range)

If doBold = True Then Selection.Font.Bold = True

Selection.TypeText Text:=preText & vbCr & vbCr

Selection.Font.Bold = False

ActiveWindow.ActivePane.Close

myComment.Edit

Selection.Collapse wdCollapseEnd

Selection.MoveLeft , 2

Selection.PasteAndFormat (wdFormatOriginalFormatting)

Selection.MoveRight , 2

End Sub

**Sub SectionHeadsInComments()**

**'** Paul Beverley - Version 26.02.24

' Adds the heading style and text of each heading into a comment

Dim cmt As Comment

For i = 1 To ActiveDocument.Paragraphs.Count

myStyle = ActiveDocument.Paragraphs(i).Range.Style

If Left(myStyle, 1) = "H" Then

Set rng = ActiveDocument.Paragraphs(i).Range

rng.End = rng.End - 1

myText = Replace(myStyle, "eading ", "") & ": " \_

& rng.Text

rng.Start = rng.End - 1

Set cmt = Selection.Comments.Add(Range:=rng)

cmt.Range.Text = myText

End If

DoEvents

Next i

End Sub

**Sub CommentAddJumpBack()**

**'** Paul Beverley - Version 26.02.24

' Creates a comment or jumps back to the text

doRoundUpSelection = True

cursorAtEnd = True

cursorAtStart = False

If Selection.Range.Information(wdInCommentPane) = True Then

startScope = ActiveDocument.Comments(1).Scope.Start

endScope = ActiveDocument.Comments(1).Scope.End

ActiveDocument.Range(startScope, endScope).Select

Exit Sub

hereNow = Selection.Start

cmtPos = 0

For i = 1 To ActiveDocument.Comments.Count - 1

endCmt = ActiveDocument.Comments(i).Range.End

If hereNow > cmtPos And hereNow < endCmt Then Exit For

cmtPos = endCmt

DoEvents

Next i

startScope = ActiveDocument.Comments(i).Scope.Start

endScope = ActiveDocument.Comments(i).Scope.End

ActiveDocument.Range(startScope, endScope).Select

If cursorAtEnd = True Then Selection.Collapse wdCollapseEnd

If cursorAtStart = True Then Selection.Collapse wdCollapseStart

Exit Sub

End If

' Select word or extend selection to whole words

If Selection.Start = Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

Else

If doRoundUpSelection = True Then

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

End If

End If

rng.HighlightColorIndex = wdYellow

Dim cmt As Comment

Set cmt = Selection.Comments.Add(Range:=rng)

cmt.Edit

Selection.TypeText Text:=myText

Selection.MoveLeft , 1

Selection.MoveRight , 1

If useCommentPane = False Then

ActiveWindow.ActivePane.Close

Else

Application.ActiveWindow.View.Zoom.Percentage = paneZoom

End If

End Sub

**Sub PreferredSpellingsAlyse()**

**'** Paul Beverley - Version 29.02.24

' Finds words similar to those in a preferred-spelling list

keyWord = "list"

wordsToAvoid = "switch"

' wordsToAvoid = "FRedit,switch"

Set sourceText = ActiveDocument

myResponse = MsgBox("Have you installed a version of" & vbCr & vbCr & \_

"ProperNounAlyse FEBRUARY 2024 or later?", vbQuestion \_

+ vbYesNoCancel, "PreferredSpellingsAlyse")

If myResponse <> vbYes Then Exit Sub

' Create a new document, copying the old

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.Text = LCase(rngOld.Text)

Set copyText = ActiveDocument

CR = vbCr: CR2 = CR & CR

' Find the list

wds = Split("," & LCase(wordsToAvoid), ",")

gottaList = False

For Each myDoc In Application.Documents

thisName = myDoc.Name

nm = LCase(thisName)

gottaList = False

If InStr(nm, LCase(keyWord)) > 0 Then gottaList = True

For i = 1 To UBound(wds)

If InStr(nm, wds(i)) > 0 Then gottaList = False

Next i

If gottaList = True Then

Exit For

End If

Next myDoc

If gottaList = False Then

Beep

MsgBox "Can't find a list."

Exit Sub

End If

myExtras = Trim(myDoc.Content.Text)

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=CR & LCase(myExtras) & CR

Selection.HomeKey Unit:=wdStory

allText = ""

For Each wd In copyText.Words

init = UCase(wd.Characters(1))

If Len(wd.Text) > 2 Then

w = Trim(wd)

allText = allText & " " & UCase(Left(w, 1)) & Mid(w, 2)

End If

i = i + 1

If i Mod 200 = 0 Then

DoEvents

wd.Select

End If

Next wd

Selection.WholeStory

Selection.Text = allText

Application.Run macroName:="ProperNounAlyse"

myExtras = ""

' Massage the 'proper noun' frequency list

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "Proper noun queries"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "Preferred spelling queries"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\*"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = " 99 ="

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^p"

.Replacement.Text = "^t= Z ^pqcqc "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^t"

.Replacement.Text = "^t "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "qcqc([!^t]@)^t([a-zA-Z0-9. ]@)^13"

.Replacement.Text = "xcz\1dfdf \2\1^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "qcqc([!^t^13]@)^t([a-zA-Z0-9. ]@)^t(\*)^13"

.Replacement.Text = "fgh\1abcd \2efgh \3\1^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

' create an array of preferred spellings

For Each w In myDoc.Content.Words

schWd = UCase(Left(w, 1)) & Mid(w, 2)

If Len(w) > 1 Then myExtras = myExtras & "," & schWd

Next w

myWd = Split(myExtras, ",")

' Check each preferred spelling to see if it appears

' in the list of 'proper nouns'

For i = 1 To UBound(myWd)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Trim(myWd(i)) & " . . . [0-9]{1,}"

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

End With

If rng.Find.Found = True Then

wdText = rng.Text

rngStart = rng.Start

rng.Expand wdParagraph

findStart = rng.Start

findEnd = rng.End

allLine = rng.Text

lft = InStr(rng, wdText)

preText = Left(allLine, lft)

postText = Mid(allLine, lft + Len(wdText))

postText = Replace(postText, vbCr, "")

' Debug.Print "|" & preText & "|"

' Debug.Print "|" & postText & "|"

' Debug.Print preText & "[!^13]@" & postText

rng.Collapse wdCollapseStart

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = preText & "[!^13]@" & postText

.Wrap = wdFindStop

.Forward = False

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

End With

'KKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKK

' rng.Select

' Debug.Print rng.Text

' Debug.Print rng.Find.Found

myDistanceUp = 9999

myFindSimilarUp = ""

If rng.Find.Found = True Then

myDistanceUp = findStart - rng.End

myFindSimilarUp = Right(preText, 1) & Replace(rng, preText, "")

myFindSimilarUp = Replace(myFindSimilarUp, postText, "")

End If

rng.Start = findEnd

rng.End = findEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = preText & "[!^13]@" & postText

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

End With

'LLLLLLLLLLLLLLLLLLLLLLLLLLLLL

' rng.Select

' Debug.Print rng.Find.Found

' Debug.Print rng.Text

myFindSimilarDown = ""

myDistanceDown = 9999

' rng.Select

If rng.Find.Found = True Then

myDistanceDown = rng.Start - findEnd

myFindSimilarDown = Right(preText, 1) & Replace(rng, preText, "")

myFindSimilarDown = Replace(myFindSimilarDown, postText, "")

End If

' Debug.Print myDistanceDown, myDistanceUp

If myFindSimilarDown = "" Then myNewWord = myFindSimilarUp

' Debug.Print myFindSimilarDown, myFindSimilarUp

If myFindSimilarUp = "" Then

myNewWord = myFindSimilarDown

Else

If myDistanceDown > myDistanceUp Then

myNewWord = myFindSimilarUp

Else

myNewWord = myFindSimilarDown

End If

End If

If myNewWord = "" Then myNewWord = "Dummy . . ."

spcPos = InStr(myNewWord, " . . .")

Debug.Print spPos

justWord = Left(myNewWord, spcPos - 1)

Debug.Print justWord & "|"

' Add new words to the list

If InStr(myExtras, "," & justWord & ",") = 0 Then

allFinds = allFinds & CR & myNewWord & \_

vbTab & vbTab & "(" & LCase(myWd(i)) & ")"

dfvzxdc = 0

End If

End If

Next i

Selection.EndKey Unit:=wdStory

Selection.InsertAfter Text:=CR

Selection.Start = 0

Selection.TypeText Text:=LCase(allFinds) & CR2

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "dummy . . \*^13"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

rng.Sort

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Preferred spellings queries" & CR

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading1

' Close unwanted files

newName = ActiveDocument.Name

closeNumber = Replace(newName, "Document", "")

closeNumber = Trim(Str(Val(closeNumber) - 1))

closeName = "Document" & closeNumber

Documents(closeName).Close SaveChanges:=False

copyText.Close SaveChanges:=False

Beep

End Sub

**Sub SimilarWordsAlyse()**

**'** Paul Beverley - Version 29.02.24

' Analyses similar words

minLengthCheck = 5

includeAcronyms = True

ignoreWords = "the this there those their they then these that"

similarChars = "bb,b; b,p; sch,sh; ch,sh; c,k; ph,f; ss,z; s,z;" & \_

" mp,m; ll,l; nn,n; nd,n; nt,n;"

' With non-English languages, you might need to make this False

ignorePlurals = True

myScreenOff = True

Set rng = Selection.Range.Duplicate

rng.End = rng.Start + 1

myLanguage = Languages(rng.LanguageID).NameLocal

Set FUT = ActiveDocument

' doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") + \_

InStr(FUT.Name, "Document") > 0)

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox(" SimilarWordsAlyse" & vbCr & vbCr & \_

"Analyse this document?", vbQuestion \_

+ vbYesNoCancel, "SimilarWordsAlyse")

If myResponse <> vbYes Then Exit Sub

End If

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

myDummy = ChrW(222)

For i = 1 To 100

spcs = " " & spcs

Next i

dummyText = ChrW(229) & "zzzx "

For i = 65 To 90

dummyText = dummyText & ChrW(i) & "zzzz "

Next i

checkFinalLetters = True

' checkFinalLetters = False

' Grey on word only

thisHighlight = wdGray25

doMissingLetter = True

' doMissingLetter = False

' Bold And blue

switchTest = True

' switchTest = False

' double strikethrough

doSimilarLetters = True

' doSimilarLetters = False

' various highlight colours + underline

doVowelTest = True

' doVowelTest = False

' various highlight colours + italic

' These last two tests cycle through these colours:

maxCol = 6

ReDim myCol(maxCol) As Integer

myCol(1) = wdYellow

myCol(2) = wdBrightGreen

myCol(3) = wdTurquoise

myCol(4) = wdRed

myCol(5) = wdPink

myCol(6) = wdGray25

colcode = 0

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = wdGray25

leadDots = " . . . "

title1 = "Similar words list"

title2 = "Similar words queries"

CR = vbCr: CR2 = CR & CR

convCharsUC = "AAAAAAA.EEEEIIII..OOOOO.OUUUU" & \_

"...aaaaaaa.eeeeiiiio.ooooo.ouuuu......"

convCharsLC = LCase(convCharsUC)

timeStart = Timer

' collect notes text, if any

endText = ""

footText = ""

If ActiveDocument.Endnotes.Count > 0 Then

endText = ActiveDocument.StoryRanges(wdEndnotesStory).Text

End If

If ActiveDocument.Footnotes.Count > 0 Then

footText = ActiveDocument.StoryRanges(wdFootnotesStory).Text

End If

' collect text in all the textboxes (if any)

sh = ActiveDocument.Shapes.Count

If sh > 0 Then

ReDim shText(sh)

i = 0

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

i = i + 1

shText(i) = shp.TextFrame.TextRange.Text

End If

End If

Next

shCount = i

End If

' Create various documents

Set rng = ActiveDocument.Content

Documents.Add

Set firstDoc = ActiveDocument

Set fnl = ActiveDocument.Content

Documents.Add

Set tempDoc = ActiveDocument

Set tmp = ActiveDocument.Content

Documents.Add

Set allText = ActiveDocument

Selection.TypeText dummyText & vbCr

Selection.FormattedText = rng.FormattedText

Selection.Collapse wdCollapseEnd

' Add notes + shape text

Selection.TypeText endText & CR & footText & CR

If shCount > 0 Then

For i = 1 To shCount

Selection.TypeText shText(i) & CR

Next i

End If

Selection.HomeKey Unit:=wdStory

Set rng = allText.Content

rng.Revisions.AcceptAll

DoEvents

StatusBar = spcs & "Preparing copied file - 1"

DoEvents

' Delete struck-through text

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.MatchWildcards = False

.Font.StrikeThrough = True

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "["

.MatchWildcards = False

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

' Remove strange unicode characters

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & ChrW(&HA000) & "-" & ChrW(&HD6FF) & "]{1,}"

.MatchWildcards = True

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

End With

DoEvents

StatusBar = spcs & "Preparing copied file - 2"

DoEvents

' Cut all and replace as pure text

Set rng = allText.Content

tmp.FormattedText = rng.FormattedText

rng.Text = LCase(tmp.Text)

tmp.Delete

DoEvents

StatusBar = spcs & "Preparing copied file - 3"

' Use qqq for apostrophe

Set rng = allText.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "n" & ChrW(8217) & "t"

.MatchWildcards = False

.Replacement.Text = "nqqqt"

.Execute Replace:=wdReplaceAll

End With

' Use qq for apostrophe

With rng.Find

.Text = "o'"

.MatchCase = True

.Replacement.Text = "oqqq"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Select just the words

DoEvents

StatusBar = spcs & "Preparing copied file - 4"

DoEvents

myChop = Trim(Str(minLengthCheck))

myFind = "<[a-zA-Z]{" & myChop & ",}"

Set rng = allText.Range

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.MatchWildcards = True

.MatchCase = True

.Replacement.Text = "^&"

.Replacement.Highlight = True

.Replacement.Font.StrikeThrough = True

.Execute Replace:=wdReplaceAll

End With

' Delete all non-strikethrough words

DoEvents

StatusBar = spcs & "Preparing copied file - 5"

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = False

.MatchWildcards = False

.MatchCase = True

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

DoEvents

' Delete the unwanted "proper nouns"

DoEvents

StatusBar = spcs & "Preparing copied file - 6"

igWords = Split(Trim(ignoreWords), " ")

For Each wd In igWords

Set rng = allText.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = wd & "^p"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchCase = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

Next wd

StatusBar = spcs & "Sorting whole file"

DoEvents

i = 0

For ch = 97 To 122

For Each myPara In allText.Paragraphs

If Asc(myPara.Range) = ch Then

DoEvents

myPara.Range.Font.StrikeThrough = False

tmp.InsertAfter myPara.Range.Text

End If

DoEvents

Next myPara

tmp.InsertAfter Text:="zzzzz" & CR

Set rng = tempDoc.Content

For i = 1 To 100

DoEvents

Next i

rng.Sort SortOrder:=wdSortOrderAscending, CaseSensitive:=True

For i = 1 To 100

DoEvents

Next i

' delete initial blank line

If Len(tempDoc.Paragraphs(1)) < 3 Then \_

tempDoc.Paragraphs(1).Range.Delete

' Create a frequency for each highlighted word

thisWord = ""

myCount = 0

For Each myPara In tempDoc.Paragraphs

Set rng = myPara.Range.Words(1)

DoEvents

nextWord = rng

If nextWord <> thisWord Then

' This is a new word

If Len(thisWord) > 1 Then

fnl.InsertAfter Text:=thisWord \_

& leadDots & Trim(Str(myCount)) & CR

End If

thisWord = nextWord

myCount = 1

Else

myCount = myCount + 1

End If

If nextWord = "zzzzz" Then Exit For

i = i + 1:

If i Mod 400 = 4 Then

DoEvents

prmt = Left(thisWord, 1) & " "

prmt = prmt & prmt & prmt & prmt

StatusBar = spcs & \_

"Preparing words for frequency list - " & prmt

DoEvents

End If

Next myPara

' Remove all words except frequency counts

Set rng = tempDoc.Content

rng.Delete

Next ch

' Find any unaccounted-for words, e.g. Ångstrom

For Each myPara In allText.Paragraphs

If myPara.Range.Words(1).Font.StrikeThrough = True Then

tmp.InsertAfter myPara.Range.Text

End If

Next myPara

tempDoc.Close SaveChanges:=False

allText.Close SaveChanges:=False

firstDoc.Activate

' Remove blank lines

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{2,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

' Resort case insensitively

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

CaseSensitive:=False

' Delete rubbish from top and bottom of list

Do

Set rng = ActiveDocument.Paragraphs(1).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until myLen > 9

Do

lastLine = ActiveDocument.Paragraphs.Count

Set rng = ActiveDocument.Paragraphs(lastLine).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until Len(rng.Text) >= 2

' Word list now has freq. count.

Do

lastLine = ActiveDocument.Paragraphs.Count

Set rng = ActiveDocument.Paragraphs(lastLine).Range

myLen = Len(rng.Text)

If myLen < 10 Then

rng.Select

Selection.Delete

End If

Loop Until Len(rng.Text) >= 2

' Create another copy for doing extra tests

Set rng = ActiveDocument.Content

Documents.Add

Set finalList = ActiveDocument

finalList.Range.Text = rng.Text

Selection.HomeKey Unit:=wdStory

' Prepare data for other tests

numWords = ActiveDocument.Paragraphs.Count

For i = 1 To numWords

aWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(aWord)

thisChar = ChrW(n)

If n > 129 Then

If n >= 217 Then aWord = Replace(aWord, thisChar, "U")

If n >= 210 Then aWord = Replace(aWord, thisChar, "O")

If n >= 204 Then aWord = Replace(aWord, thisChar, "I")

If n >= 200 Then aWord = Replace(aWord, thisChar, "E")

If n >= 192 Then aWord = Replace(aWord, thisChar, "A")

End If

allWords = allWords & aWord

jmp = 100

If i Mod jmp = 1 Then

pq = pq + 1

DoEvents

StatusBar = spcs & \_

"Preparing data for other tests - 1 - " & pq

DoEvents

End If

Next i

' ...for the vowel test below

DoEvents

StatusBar = spcs & "Preparing data for other tests - 2"

DoEvents

noVowelWords = " " & allWords

noVowelWords = Replace(noVowelWords, " A", "\_1")

noVowelWords = Replace(noVowelWords, " E", "\_2")

noVowelWords = Replace(noVowelWords, " I", "\_3")

noVowelWords = Replace(noVowelWords, " O", "\_4")

noVowelWords = Replace(noVowelWords, " U", "\_5")

noVowelWords = Replace(noVowelWords, " Y", "\_6")

For k = 2 To Len(noVowelWords) - 1

thisChar = Mid(noVowelWords, k, 1)

n = AscW(thisChar)

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsLC, n - 191, 1)

If myNewChar <> "." Then noVowelWords = \_

Replace(noVowelWords, thisChar, myNewChar)

End If

Next k

noVowelWords = Replace(noVowelWords, "a", "")

noVowelWords = Replace(noVowelWords, "e", "")

noVowelWords = Replace(noVowelWords, "i", "")

noVowelWords = Replace(noVowelWords, "o", "")

noVowelWords = Replace(noVowelWords, "u", "")

noVowelWords = Replace(noVowelWords, "y", "")

noVowelWords = Replace(noVowelWords, "A", "")

noVowelWords = Replace(noVowelWords, "E", "")

noVowelWords = Replace(noVowelWords, "I", "")

noVowelWords = Replace(noVowelWords, "O", "")

noVowelWords = Replace(noVowelWords, "U", "")

noVowelWords = Replace(noVowelWords, "Y", "")

noVowelWords = Replace(noVowelWords, "\_1", " A")

noVowelWords = Replace(noVowelWords, "\_2", " E")

noVowelWords = Replace(noVowelWords, "\_3", " I")

noVowelWords = Replace(noVowelWords, "\_4", " O")

noVowelWords = Replace(noVowelWords, "\_5", " U")

noVowelWords = Replace(noVowelWords, "\_6", " Y")

' ...for the similar words test

DoEvents

StatusBar = spcs & "Preparing data for other tests - 3"

DoEvents

similarAllWords = " " & LCase(allWords)

similarChars = Replace(similarChars, " ", "")

sChars = Replace(similarChars, " ", "")

Do

commaPos = InStr(sChars, ",")

charWas = Left(sChars, commaPos - 1)

sChars = Mid(sChars, commaPos + 1)

semicolonPos = InStr(sChars, ";")

charNew = Left(sChars, semicolonPos - 1)

sChars = Mid(sChars, semicolonPos + 1)

similarAllWords = Replace(similarAllWords, charWas, charNew)

Loop Until Len(sChars) < 2

' Changes all the accented characters to non-accented

DoEvents

StatusBar = spcs & "Preparing data for other tests - 4"

DoEvents

sWd = similarAllWords

For k = 1 To Len(sWd) - 1

thisChar = Mid(sWd, k, 1)

n = AscW(thisChar)

myNewChar = "."

If n > 191 And n < 256 Then

myNewChar = Mid(convCharsLC, n - 191, 1)

If myNewChar <> "." Then sWd = Replace(sWd, \_

thisChar, myNewChar)

End If

Next k

similarAllWords = sWd

' Catch words with only the final two letters the same

i = 0

If checkFinalLetters = True Then

For Each myPara In ActiveDocument.Paragraphs

gotOne = False

myWord = Trim(myPara.Range.Words(1))

myLen = Len(myWord)

If myLen > 6 Then

myTarget = "^p" & Left(myWord, myLen - 2) & "^$^$ "

myCut = 2

Else

myTarget = "^p" & Left(myWord, myLen - 1) & "^$ "

myCut = 1

End If

Set rng = ActiveDocument.Content

rng.Start = myPara.Range.End - 3

rng.Collapse wdCollapseStart

With rng.Find

.Replacement.ClearFormatting

.ClearFormatting

.Text = myTarget

.Replacement.Text = ""

.Forward = True

.MatchCase = True

.MatchWildcards = False

.Wrap = wdFindStop

End With

rng.Find.Execute

Do While rng.Find.Found

gotOne = True

rng.MoveStart 1

rng.End = rng.Start + myLen - myCut

rng.HighlightColorIndex = thisHighlight

rng.Font.Bold = True

rng.Find.Execute

Loop

If gotOne = True Then

Set rng = myPara.Range.Words(1)

rng.End = rng.Start + myLen - myCut

rng.HighlightColorIndex = thisHighlight

rng.Font.Bold = True

End If

i = i + 1

If i Mod 100 = 1 Then

DoEvents

StatusBar = spcs & "Doing test (5) on " & myWord

DoEvents

End If

Next myPara

End If

If doMissingLetter = True Then

' Start of test

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Changes the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (4) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' Check if word reappears with one letter missing (1)

For k = 2 To Len(myWord) - 1

testWord = " " & Left(myWord, k - 1) & Mid(myWord, k + 1)

wordPos = InStr(allWords, testWord)

If wordPos > 0 Then

lastLetter = Mid(myWord, Len(myWord) - 1, 1)

' but not "s" at the end, unless it's a spelling error

If lastLetter = "s" Then

ignoreIt = (Application.CheckSpelling(myWord, \_

MainDictionary:=myLanguage) = True)

Else

ignoreIt = False

End If

If ignoreIt = False And ignorePlurals = True Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

' mark the pair

leftBit = Left(allWords, InStr(allWords, testWord) \_

+ Len(testWord) - 1)

j = Len(leftBit) - Len(Replace(leftBit, " ", ""))

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Bold = True

rng.Font.Color = wdColorBlue

Set rng = ActiveDocument.Paragraphs(j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Bold = True

rng.Font.Color = wdColorBlue

End If

End If

Next k

If Left(myWord, 2) = "Mc" Or Left(myWord, 3) = "Mac" Or \_

Left(myWord, 3) = "Mag" Then

McList = McList & ActiveDocument.Paragraphs(i).Range

End If

Next i

End If

If doSimilarLetters = True Then

doneWords = ""

doneSimilarWords = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Changes the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (3) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check similar spellings: Perutz/Peruts or Chebyshev/Chevychev

similarWord = " " & LCase(myWord)

sChars = similarChars

Do

commaPos = InStr(sChars, ",")

charWas = Left(sChars, commaPos - 1)

sChars = Mid(sChars, commaPos + 1)

semicolonPos = InStr(sChars, ";")

charNew = Left(sChars, semicolonPos - 1)

sChars = Mid(sChars, semicolonPos + 1)

similarWord = Replace(similarWord, charWas, charNew)

Loop Until Len(sChars) < 2

' Changes all the accented characters to non-accented

For k = 1 To Len(myWord) - 1

thisChar = Mid(myWord, k, 1)

n = AscW(thisChar)

If n > 191 And n < 256 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

Next k

similarAllWords = Mid(similarAllWords, Len(similarWord))

theseWords = similarAllWords

If InStr(doneSimilarWords, similarWord) = 0 And \_

InStr(theseWords, similarWord) > 0 Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Underline = True

doneSimilarWords = doneSimilarWords & similarWord

' search through all the following words

theseWords = similarAllWords

For j = 1 To numWords - i

spPos = InStr(Trim(theseWords) & " ", " ")

If Left(theseWords, spPos + 1) = similarWord Then

Set rng = ActiveDocument.Paragraphs(i + j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Underline = True

End If

theseWords = Mid(theseWords, spPos + 1)

capThisLetter = Mid(theseWords, 2, 1)

If capThisLetter <> LCase(captestLetters) Then Exit For

Next j

End If

Next i

End If

If switchTest = True Then

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Changes the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (2) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check for switched chars

wordLen = Len(myWord) - 1

For k = 1 To Len(myWord) - 3

otherWord = Left(myWord, k) & Mid(myWord, k + 2, 1) & \_

Mid(myWord, k + 1, 1) & Mid(myWord, k + 3)

wordPos = InStr(testWords, otherWord)

If wordPos > 0 Then

' Find the position of the matching word

matchWord = Mid(testWords, wordPos, Len(myWord))

leftBit = Left(allWords, InStr(allWords, matchWord) + 1)

j = Len(leftBit) - Len(Replace(leftBit, " ", "")) + 1

ActiveDocument.Paragraphs(i).Range.Font.DoubleStrikeThrough \_

= True

ActiveDocument.Paragraphs(i).Range.HighlightColorIndex \_

= thisCol

ActiveDocument.Paragraphs(j).Range.Font.DoubleStrikeThrough \_

= True

ActiveDocument.Paragraphs(j).Range.HighlightColorIndex \_

= thisCol

End If

Next k

Next i

End If

If doVowelTest = True Then

doneWords = ""

doneSimilarWords = ""

McList = ""

For i = 1 To ActiveDocument.Paragraphs.Count - 1

myWord = ActiveDocument.Paragraphs(i).Range.Words(1)

n = AscW(myWord)

thisChar = ChrW(n)

myNewChar = "."

' Changes the capital letter, if a vowel

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then myWord = Replace(myWord, \_

thisChar, myNewChar)

End If

If i Mod 50 = 1 Then

DoEvents

StatusBar = spcs & "Other tests (1) on " & myWord

DoEvents

End If

testWords = Replace(allWords, myWord, "")

captestLetters = Left(myWord, 1)

' check if there's a word with different vowels

otherWord = " " & Replace(myWord, "a", "")

otherWord = Replace(otherWord, "e", "")

otherWord = Replace(otherWord, "i", "")

otherWord = Replace(otherWord, "o", "")

otherWord = Replace(otherWord, "u", "")

otherWord = Replace(otherWord, "y", "")

' Delete all the accented characters

For k = 3 To Len(otherWord) - 1

thisChar = Mid(otherWord, k, 1)

n = AscW(thisChar)

If InStr("AEIOUY", thisChar) > 0 Then

otherWord = Left(otherWord, k - 1) & "=" & Mid(otherWord, k + 1)

Else

If n > 191 And n < 221 Then

myNewChar = Mid(convCharsUC, n - 191, 1)

If myNewChar <> "." Then

otherWord = Replace(otherWord, thisChar, "=")

End If

End If

End If

Next k

otherWord = Replace(otherWord, "=", "")

' otherWord is now the word under test (vowel-less)

otherWord = Replace(otherWord, ".", "")

noVowelWords = Mid(noVowelWords, Len(otherWord))

If Left(noVowelWords, 1) <> " " Then noVowelWords = \_

" " & noVowelWords

theseWords = noVowelWords

wordPos = InStr(noVowelWords, otherWord)

If InStr(doneWords, otherWord) = 0 And wordPos > 0 Then

colcode = (colcode + 1) Mod maxCol

thisCol = myCol(colcode + 1)

Set rng = ActiveDocument.Paragraphs(i).Range

rng.HighlightColorIndex = thisCol

rng.Font.Italic = True

doneWords = doneWords & otherWord

For j = 1 To numWords - i

spPos = InStr(Trim(theseWords) & " ", " ")

firstWord = Left(theseWords, spPos + 1)

theseWords = Mid(theseWords, spPos + 1)

If firstWord = otherWord Then

Set rng = ActiveDocument.Paragraphs(i + j).Range

rng.HighlightColorIndex = thisCol

rng.Font.Italic = True

End If

capThisLetter = Mid(theseWords, 2, 1)

If capThisLetter > "" And capThisLetter <> \_

captestLetters Then Exit For

Next j

End If

Next i

End If

finishOff:

Selection.EndKey Unit:=wdStory

Selection.TypeText CR2 & McList

Selection.HomeKey Unit:=wdStory

Selection.TypeText title1 & CR

Do

Selection.Expand wdParagraph

If Len(Selection) < 3 Or LCase(Selection) = \_

UCase(Selection) Then Selection.Delete

Loop Until LCase(Selection) <> UCase(Selection)

Selection.HomeKey Unit:=wdStory, Extend:=wdExtend

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

' Restore apostrophes

Set rng = finalList.Range

With rng.Find

.Text = "qqq"

.MatchCase = False

.Replacement.Text = "'"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Find first highlight

Set rng = finalList.Content

With rng.Find

.Text = "Zzzzz"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

End With

Set rng = finalList.Content

With rng.Find

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

rng.Select

Selection.Collapse wdCollapseStart

Set finalList = ActiveDocument

firstDoc.Activate

' Find sets of sounds-like words

StatusBar = spcs & "Sounds-like tests"

k = 0

For Each myPara In ActiveDocument.Paragraphs

myWord = Trim(myPara.Range.Words(1))

k = k + 1

If k Mod 40 = 1 Then

DoEvents

StatusBar = spcs & "Sounds-like test: " & myWord

DoEvents

End If

hasAccent = False

For i = 1 To Len(myWord)

ascChar = AscW(Mid(myWord, i))

If ascChar > 128 Or ascChar = Asc("?") Then hasAccent = True

Next i

' Go and find the first sounds-like word

initLetter = Left(myWord, 1)

If Len(myWord) > 2 And myPara.Range.HighlightColorIndex > 0 And \_

hasAccent = False And InStr(allSets, myWord & leadDots) \_

= 0 Then

Set rng = ActiveDocument.Content

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myWord

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = False

.MatchSoundsLike = True

.Execute

End With

Set myPara = rng.Paragraphs(1).Range

rng.Collapse wdCollapseEnd

Loop Until Left(myPara, 1) = initLetter

setOfWords = myPara

gottaSet = False

rng.Collapse wdCollapseEnd

rng.Find.Execute

Do While rng.Find.Found = True

Set myPara = rng.Paragraphs(1).Range

If Left(myPara, 1) = initLetter Then

gottaSet = True

setOfWords = setOfWords & myPara

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

If gottaSet = True Then allSets = allSets & setOfWords & CR

End If

Next myPara

Selection.WholeStory

If Len(allSets) < 2 Then

Selection.TypeText "None found with this test"

Else

Selection.TypeText allSets

End If

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Similar words by sound" & CR

Selection.HomeKey Unit:=wdStory, Extend:=wdExtend

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = 0

rng.Copy

ActiveDocument.Close SaveChanges:=False

finalList.Activate

' Remove highlighting from second half of words

' that are only case changes of one another

totParas = ActiveDocument.Paragraphs.Count

For i = 1 To totParas - 1

a = Trim(ActiveDocument.Paragraphs(i).Range.Words(1))

b = Trim(ActiveDocument.Paragraphs(i + 1).Range.Words(1))

a = Mid(a, 2)

b = Mid(b, 2)

If LCase(a) = LCase(b) And Len(a) > 2 Then

If (UCase(a) = a And LCase(b) = b) Or (UCase(b) = b And \_

LCase(a) = a) Then

ActiveDocument.Paragraphs(i).Range.Words(1).HighlightColorIndex = 0

ActiveDocument.Paragraphs(i + 1).Range.Words(1).HighlightColorIndex \_

= 0

End If

End If

If i Mod 50 = 0 Then

DoEvents

StatusBar = spcs & "Final checks: " & totParas - i

DoEvents

End If

Next i

myOnames = ""

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13O[!a-z]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchSoundsLike = False

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rng.Collapse wdCollapseEnd

rng.Expand wdWord

wd = Mid(rng.Text, 3)

rng.Expand wdParagraph

pa = rng.Text

Set rng2 = ActiveDocument.Content

With rng2.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13" & wd

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

If rng2.Find.Found Then

rng2.Collapse wdCollapseEnd

rng2.Expand wdParagraph

pa2 = rng2.Text

myOnames = myOnames & pa2 & pa & vbCr

End If

rng.Collapse wdCollapseEnd

rng.End = rng.End - 2

rng.Find.Execute

Loop

If myOnames > "" Then

Selection.EndKey Unit:=wdStory

Selection.TypeText "Possible O'<something> errors" & vbCr

Selection.MoveUp , 1

Selection.Style = ActiveDocument.Styles(wdStyleHeading1)

Selection.EndKey Unit:=wdStory

Selection.TypeText myOnames

Selection.HomeKey Unit:=wdStory

End If

Set rng = ActiveDocument.Content

finalList.Activate

Selection.EndKey Unit:=wdStory

Selection.TypeText vbCr & vbCr & vbCr

Selection.Paste

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myDummy

.Wrap = wdFindContinue

.Replacement.Text = " "

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^$zzz^$" & leadDots & "1" & vbCr

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

' Clear clipboard

Set rng = ActiveDocument.Content

rng.End = 2

rng.Copy

Set finalList = ActiveDocument

StatusBar = "Creating queries list"

Set rng = ActiveDocument.Content

Documents.Add

Selection.FormattedText = rng.FormattedText

Set queriesDoc = ActiveDocument

ActiveDocument.Paragraphs(1).Range.Delete

Set rng = ActiveDocument.Content

rng.Font.StrikeThrough = True

For Each myPara In ActiveDocument.Paragraphs

Set ch = myPara.Range.Characters(1)

chCol = ch.HighlightColorIndex

If chCol > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

myLen = Len(myPara.Range.Text)

If myLen > 4 Then

If chCol > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

Set che = myPara.Range.Characters(myLen - 2)

If che.HighlightColorIndex > 0 Then

myPara.Range.Font.StrikeThrough = False

End If

End If

Next myPara

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = True

.Wrap = wdFindContinue

.Replacement.Text = "^p"

.Forward = True

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

rng.Font.StrikeThrough = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{3,}"

.Wrap = wdFindContinue

.Replacement.Text = "^p^p"

.Forward = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

For Each myPara In ActiveDocument.Paragraphs

myText = myPara.Range.Text

If Len(myText) > 4 Then

Set ch = myPara.Range.Characters(1)

numChars = myPara.Range.Characters.Count

Set myEnd = myPara.Range.Characters(numChars)

colNum = ch.HighlightColorIndex Mod 8

If ch.Font.Bold = True Then

myTxt = "qcqc " & Str(colNum + 1) & " = zczc"

Else

myTxt = "qcqc zczc"

End If

If ch.Font.Underline > 0 And colNum > 0 Then

myBit = "\* "

myTxt = Replace(myTxt, " = ", "")

Else

myBit = ""

End If

myPara.Range.InsertBefore myBit & myTxt

If ch.Font.Italic = True Then

myEnd.InsertBefore "qpqp= " & Chr(65 + colNum)

End If

End If

i = i + 1

If i Mod 20 = 0 And Len(myText) > 4 Then

myText = Replace(myText, vbCr, "")

StatusBar = spcs & "Creating queries list: " & myText

End If

DoEvents

Next myPara

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\\* qcqc(\*)zczc"

.Wrap = wdFindContinue

.Replacement.Text = "\* \1^t"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "qcqc(\*)zczc"

.Wrap = wdFindContinue

.Replacement.Text = "\1^t"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "qpqp(\*)^13"

.Replacement.Text = "^t\1^p"

.Replacement.Highlight = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "= ^$"

.Replacement.Text = ""

.Replacement.Font.Bold = False

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Unhighlight plurals, plurales, pluraled

Set allText = ActiveDocument.Content

For Each wd In allText.Words

myWd = Trim(wd)

myLen = Len(myWd)

If myLen > minLengthCheck - 1 Then

myLast = Right(myWd, 1)

If InStr("ds", myLast) > 0 Then

myShort = Left(myWd, myLen - 1)

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myShort

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.MatchWholeWord = True

.Execute

DoEvents

If .Found Then

rng.Expand wdParagraph

rng.HighlightColorIndex = wdNoHighlight

wd.Expand wdParagraph

wd.HighlightColorIndex = wdNoHighlight

End If

End With

If myLast = "d" Then

myShort = Left(myWd, myLen - 1)

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myShort & "s"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.MatchWholeWord = True

.Execute

DoEvents

If .Found Then

rng.Expand wdParagraph

rng.HighlightColorIndex = wdNoHighlight

wd.Expand wdParagraph

wd.HighlightColorIndex = wdNoHighlight

End If

End With

End If

myWd = myShort

myLen = Len(myWd)

myLast = Right(myWd, 1)

If myLast = "e" Then

myShort = Left(myWd, myLen - 1)

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myShort

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.MatchWholeWord = True

.Execute

DoEvents

If .Found Then

rng.Expand wdParagraph

rng.HighlightColorIndex = wdNoHighlight

wd.Expand wdParagraph

wd.HighlightColorIndex = wdNoHighlight

End If

End With

End If

End If

End If

Next wd

' Tidy up report list

Set rng = ActiveDocument.Content

rng.Font.Bold = False

rng.Font.Italic = False

rng.Font.DoubleStrikeThrough = False

rng.Font.Underline = False

rng.Font.Color = wdColorBlack

Selection.HomeKey Unit:=wdStory

Selection.TypeText title2 & CR

Set rng = ActiveDocument.Content.Paragraphs(2).Range

If rng.Text = vbCr Then rng.Delete

Set rng = ActiveDocument.Content.Paragraphs(1).Range

rng.Style = ActiveDocument.Styles(wdStyleHeading1)

StatusBar = " "

Options.DefaultHighlightColorIndex = oldColour

lighterColour = wdGray25

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "= ^$"

.Replacement.Text = ""

.Replacement.Font.ColorIndex = lighterColour

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = "^#^# ="

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^# ="

.Execute Replace:=wdReplaceAll

.Text = "^# ^t"

.Execute Replace:=wdReplaceAll

.Text = "\*"

.Execute Replace:=wdReplaceAll

DoEvents

End With

Application.ScreenUpdating = True

If doingSeveralMacros = False Then

myTime = (Int(10 \* (Timer - timeStart) / 60) / 10)

Beep

If myTime > 0 Then MsgBox myTime & " minutes"

End If

If InStr(FUT.Name, "zzTestFile") > 0 Then FUT.Activate

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub ChapterStartBreak()**

**'** Paul Beverley - Version 05.03.23

' Finds all chapter titles, adds page break and removes multi-newlines

myResponse = MsgBox("Is the cursor in a chapter title?", \_

vbQuestion + vbYesNo, "ChapterStartBreak")

If myResponse <> vbYes Then Beep: Exit Sub

myStyle = Selection.Style

If Selection.Style = ActiveDocument.Styles(wdStyleNormal) Then

myResponse = MsgBox("Are you sure?!", \_

vbQuestion + vbYesNo, "ChapterStartBreak")

If myResponse <> vbYes Then Beep: Exit Sub

End If

ActiveDocument.Styles(myStyle).ParagraphFormat.PageBreakBefore = True

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13{3,}"

.Replacement.Text = "^p"

.Forward = True

.Wrap = wdFindContinue

.Format = False

.Execute Replace:=wdReplaceAll

.MatchWildcards = True

End With

End Sub

**Sub TextPlusNumber()**

**'** Paul Beverley - Version 11.03.24

' Finds numbers expressed in words + adds "(figures)"

myUnits = ":one:two:three:four:five:six:seven:eight:nine:ten"

myTens = ":ten:twenty:thirty:forty:fifty:sixty:seventy:eighty:ninety:hundred"

myTeens = ":eleven:twelve:thirteen:fourteen:fifteen:sixteen:seventeen:eighteen:nineteen"

allNumberWords = myUnits & myTens & myTeens & ":a:and:-:"

mySmart = Options.SmartCutPaste

Options.SmartCutPaste = False

Dim wd(6) As String

Set rng = Selection.Range.Duplicate

Do

rng.Expand wdWord

rng.Collapse wdCollapseStart

gottaWord = False

For i = 1 To 50

rng.MoveEnd wdWord, 1

thisWord = LCase(Trim(rng.Words(rng.Words.Count)))

If InStr("aand-", thisWord) = 0 And InStr(allNumberWords, \_

":" & thisWord & ":") > 0 Then

If Right(Trim(rng.Text), 6) = "no-one" Then

gottaWord = False

Else

gottaWord = True

Exit For

End If

End If

Next i

rng.Collapse wdCollapseEnd

rng.MoveEnd wdWord, -1

If gottaWord = False Then

rng.Select

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

MsgBox ("Finished")

Exit Sub

End If

gotStart = False

Do While gotStart = False

rng.MoveStart wdWord, -1

If InStr(allNumberWords, ":" & LCase(Trim(rng.Words(1))) & ":") = 0 Then

gotStart = True

rng.MoveStart wdWord, 1

gotStart = True

End If

DoEvents

Loop

gotEnd = False

Do While gotEnd = False

rng.MoveEnd wdWord, 1

lastWord = LCase(Trim(rng.Words(rng.Words.Count)))

If InStr(allNumberWords, ":" & lastWord & ":") = 0 Then

gotEnd = True

rng.MoveEnd wdWord, -1

End If

DoEvents

Loop

fstWd = LCase(Trim(rng.Words(1)))

If InStr("and", fstWd) > 0 Then rng.MoveStart wdWord, 1

' To catch 'a', 'an', and 'and' as final word

lastWd = LCase(Trim(rng.Words(rng.Words.Count)))

If InStr("and", lastWd) > 0 Then rng.MoveEnd wdWord, -1

' Catch missing "and" for American usage, e.g. "two hundred fifty-two"

rText = LCase(rng.Text)

If lastWd <> "hundred" And InStr(rText, "hundred") > 0 And \_

InStr(rText, "hundred and") = 0 Then

rng.Text = Replace(rText, "hundred", "hundred and")

End If

allWords = LCase(rng.Text)

numWords = rng.Words.Count

Dim n(6) As Integer

For i = 1 To numWords

wdPos = InStr(allNumberWords, ":" & LCase(Trim(rng.Words(i))) & ":")

leftWords = Left(allNumberWords, wdPos)

n(i) = Len(leftWords) - Len(Replace(leftWords, ":", ""))

Next

If n(1) = 30 Then n(1) = 1

Select Case numWords

Case 1

myResult = n(1)

If n(1) > 10 Then myResult = 10 \* (n(1) - 10)

If n(1) > 20 Then myResult = n(1) - 10

Case 2

If n(2) = 20 Then ' "hundred"

myResult = n(1) \* 100

Else

myResult = 10 \* (n(1) - 10) + n(2)

If myResult < 21 Then

Beep

rng.Select

Options.SmartCutPaste = mySmart

Exit Sub

End If

End If

Case 3

myResult = 10 \* (n(1) - 10) + n(3)

If n(2) <> 32 Then ' hyphen

If n(2) = 20 Then

myResult = n(3) + 100 \* n(1)

Else

Beep

rng.Select

Options.SmartCutPaste = mySmart

Exit Sub

End If

End If

Case 4

If n(2) <> 20 Then ' "hundred"

Beep

rng.Select

Options.SmartCutPaste = mySmart

Exit Sub

End If

myResult = n(4)

If n(4) > 10 Then myResult = 10 \* (n(4) - 10)

If n(4) > 20 Then myResult = n(4) - 10

myResult = myResult + 100 \* n(1)

Case 5

If n(2) <> 20 Then ' "hundred"

Beep

rng.Select

Options.SmartCutPaste = mySmart

Exit Sub

End If

myResult = 100 \* n(1) + 10 \* (n(3) - 10) + n(5)

Case 6

If n(2) <> 20 Then ' "hundred"

Beep

rng.Select

Options.SmartCutPaste = mySmart

Exit Sub

End If

myResult = 100 \* n(1) + 10 \* (n(4) - 10) + n(6)

Case Else

Beep

Options.SmartCutPaste = mySmart

rng.Select

Exit Sub

End Select

Do While InStr(" ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

rng.Collapse wdCollapseEnd

rng.MoveEnd wdWord, 3

parenPos = InStr(rng.Text, "(")

rng.Collapse wdCollapseStart

If parenPos = 0 Then

rng.InsertAfter Text:=" (" & Trim(Str(myResult)) & ")"

Else

rng.MoveStart , 2

rng.MoveEnd wdWord, 1

If Val(rng.Text) <> myResult Then

rng.Select

Beep

MsgBox ("Please check!")

Options.SmartCutPaste = mySmart

Exit Sub

End If

End If

rng.Collapse wdCollapseEnd

rng.Select

DoEvents

Loop Until 0

Options.SmartCutPaste = mySmart

End Sub

**Sub PageBreakColour()**

**'** Paul Beverley - Version 14.03.24

' Colours/highlights the lines either side of each page break

' Select font colour and/or highlight colour

' = 0 means don't add colour/highlight

' myColour1 =wdColorBlue

myColour1 = 0

myColour2 = wdColorRed

myHiColour1 = wdYellow

' myHiColour2 = wdBrightGreen

myHiColour2 = 0

myScreenOff = True

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

nowPos = 99

Do

myCount = myCount + 1

Do

wasPos = nowPos

nowPos = Selection.Start

If nowPos = wasPos Then

Beep

Exit Sub

End If

Selection.MoveDown , 1

lineNo = Selection.Information(wdFirstCharacterLineNumber)

DoEvents

Loop Until lineNo = 1

If myCount Mod 10 = 0 Then Application.ScreenUpdating = True

Selection.Expand wdLine

If myColour2 <> 0 Then Selection.Font.Color = myColour2

If myHiColour2 <> 0 Then \_

Selection.Range.HighlightColorIndex = myHiColour2

Selection.MoveUp , 1

Selection.Expand wdLine

If myColour1 <> 0 Then Selection.Font.Color = myColour1

If myHiColour1 <> 0 Then \_

Selection.Range.HighlightColorIndex = myHiColour1

Selection.MoveDown , 3

DoEvents

Application.ScreenUpdating = False

Loop Until 0

Beep

Selection.HomeKey Unit:=wdStory

Application.ScreenUpdating = False

Exit Sub

' Switch the screen back on if there's an error

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub HighlightYellow()**

**'** Paul Beverley - Version 11.11.24

' Highlights the selected text (or a word or a paragraph)

defaultSelect = "word"

' defaultSelect = "para"

selectWholeWords = True

myColour = wdYellow

' myColour = wdBrightGreen

' myColour = wdTurquoise

' myColour = wdPink

' myColour = wdRed

' myColour = wdGray50

' myColour = wdGray25

If Selection.start = Selection.End Then

If defaultSelect = "para" Then

Selection.Expand wdParagraph

Else

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

Else

If selectWholeWords = True Then

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.start = Selection.start

rng.Select

End If

End If

Selection.Range.HighlightColorIndex = myColour

End Sub

**Sub HighlightBrightGreen()**

**'** Paul Beverley - Version 15.03.24

' Highlights the selected text (or a word or a paragraph)

defaultSelect = "word"

' defaultSelect = "para"

selectWholeWords = True

' myColour = wdYellow

myColour = wdBrightGreen

' myColour = wdTurquoise

' myColour = wdPink

' myColour = wdRed

' myColour = wdGray50

' myColour = wdGray25

If Selection.Start = Selection.End Then

If defaultSelect = "para" Then

Selection.Expand wdParagraph

Else

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

End If

Else

If selectWholeWords = True Then

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.Start = Selection.Start

rng.Select

End If

End If

Selection.Range.HighlightColorIndex = myColour

End Sub

**Sub CopyAllComments()**

**'** Paul Beverley - Version 17.03.24

' Copies all the comments into a new document

If ActiveDocument.Comments.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdCommentsStory)

Documents.Add

Selection.FormattedText = rng.FormattedText

Else

Beep

MsgBox "No comments found!"

End If

End Sub

**Sub WordCopierDown()**

**'** Paul Beverley - Version 25.03.24

' Copies the word from the beginning of the selection to the end

' insertWord = True

insertWord = False

' i.e. overwrite the word

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

goodWord = rng.Text

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

If insertWord = True Then

rng.InsertAfter Text:=" " & goodWord

Else

rng.Text = goodWord

End If

End Sub

**Sub WordCopierUp()**

**'** Paul Beverley - Version 25.03.24

' Copies the word from the end of the selection to the beginning

' insertWord = True

insertWord = False

' i.e. overwrite the word

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

goodWord = rng.Text

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

If insertWord = True Then

rng.InsertBefore Text:=goodWord & " "

Else

rng.Text = goodWord

End If

End Sub

**Sub NumberSelectedList()**

**'** Paul Beverley - Version 12.03.24

' Applies (or corrects the existing) numbering on the selected text paragraphs

myStart = 1

' myLink = ")" & vbTab

myLink = ") "

Dim rng As Range

i = myStart

For Each myPara In Selection.Range.Paragraphs

Set rng = myPara.Range

If LCase(rng) <> UCase(rng) Then

ch = rng.Characters(1)

If LCase(ch) <> UCase(ch) Then

rng.InsertBefore Text:=Trim(Str(i)) & myLink

Else

j = 1

Do

j = j + 1

ch = rng.Characters(j)

Loop Until LCase(ch) <> UCase(ch)

rng.End = rng.Start + j - 1

rng.Text = Trim(Str(i)) & myLink

End If

i = i + 1

DoEvents

End If

Next myPara

Beep

End Sub

**Sub AynHamzaFormat()**

**'** Paul Beverley - Version 28.03.24

' Corrects formatting of ayns and hamzas

aynColour = wdYellow

hamzaColour = wdBrightGreen

' Superscripted c's

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "c"

.Font.Superscript = True

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ChrW(703)

.Replacement.Font.Superscript = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set rng = ActiveDocument.Content

' Find ayns

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[" & ChrW(703) & "]{1,2}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCountAyn = 0

Do While rng.Find.Found = True

myCountAyn = myCountAyn + 1

rng.Select

Set rngTest = rng.Duplicate

rngTest.Collapse wdCollapseEnd

rngTest.MoveEnd , 1

If UCase(rngTest.Text) = LCase(rngTest.Text) Then

rngTest.MoveStart , -2

rngTest.MoveEnd , -2

End If

nextChar = rngTest.Text

If myCountHamza Mod 20 = 0 Then rng.Select

rng.Font.Size = rngTest.Font.Size

rng.Font.Italic = rngTest.Font.Italic

rng.Font.Bold = rngTest.Font.Bold

rng.Font.Name = rngTest.Font.Name

rng.HighlightColorIndex = aynColour

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Set rng = ActiveDocument.Content

' Find hamzas

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(702)

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myCountHamza = 0

Do While rng.Find.Found = True

myCountHamza = myCountHamza + 1

Set rngTest = rng.Duplicate

rngTest.MoveStart , -1

rngTest.MoveEnd , -1

If myCountHamza Mod 20 = 0 Then rng.Select

rng.Font.Size = rngTest.Font.Size

rng.Font.Italic = rngTest.Font.Italic

rng.Font.Bold = rngTest.Font.Bold

rng.Font.Name = rngTest.Font.Name

rng.HighlightColorIndex = hamzaColour

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

MsgBox "Changed ayns: " & myCountAyn & vbCr & "Changed hamzas: " & myCountHamza

End Sub

**Sub BoldItalicToStyle()**

**'** Paul Beverley - Version 21.09.24

' Changes bold, italic and bold-italic applied directly, into character styles

doHighlight = True

myHiColour = wdYellow

myItalicStyle = "Emphasis"

myBoldStyle = "Strong"

myBoldItalicStyle = "Intense Emphasis"

doItalic = True

doBold = True

doBoldItalic = True

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myHiColour

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Replacement.Text = ""

.Font.Bold = True

.Font.Italic = True

.Replacement.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Font.StrikeThrough = True

.Replacement.Style = ActiveDocument.Styles(myBoldItalicStyle)

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Style = ActiveDocument.Styles(myBoldItalicStyle)

.Replacement.Font.StrikeThrough = True

If doItalic = True Then .Execute Replace:=wdReplaceAll

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Replacement.ClearFormatting

.Font.StrikeThrough = False

.Font.Italic = True

.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Style = ActiveDocument.Styles(myItalicStyle)

If doHighlight = True Then .Replacement.Highlight = True

If doBold = True Then .Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.StrikeThrough = False

.Font.Bold = True

.Font.Italic = False

.Replacement.Font.Bold = False

.Replacement.Style = ActiveDocument.Styles(myBoldStyle)

If doHighlight = True Then .Replacement.Highlight = True

If doBoldItalic = True Then .Execute Replace:=wdReplaceAll

DoEvents

End With

rng.Font.StrikeThrough = False

If ActiveDocument.Footnotes.Count > 0 Then

For Each nt In ActiveDocument.Footnotes

' Strike through all footnote numbers

Set rng = nt.Range

rng.Collapse wdCollapseStart

rng.MoveStart , -2

rng.Font.StrikeThrough = True

rng.Collapse wdCollapseEnd

Next nt

For Each nt In ActiveDocument.Footnotes

' Now make the style changes

With nt.Range.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindContinue

.Font.Superscript = True

.Font.StrikeThrough = False

.Forward = True

.Replacement.Font.Superscript = False

.Replacement.Style = ActiveDocument.Styles("Superscript")

.Replacement.Text = "^&"

.MatchWildcards = False

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.StrikeThrough = False

.Font.Subscript = True

.Replacement.Font.Subscript = False

.Replacement.Style = ActiveDocument.Styles("Subscript")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.StrikeThrough = False

.Font.Italic = True

.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Style = ActiveDocument.Styles("Italic")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Bold = True

.Font.StrikeThrough = False

.Font.Italic = False

.Replacement.Font.Bold = False

.Replacement.Style = ActiveDocument.Styles("Bold")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Underline = True

.Font.StrikeThrough = False

.Replacement.Font.Underline = False

.Replacement.Style = ActiveDocument.Styles("Underline")

.Replacement.Highlight = True

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

.ClearFormatting

.Replacement.ClearFormatting

.Font.Bold = True

.Font.Italic = True

.Font.StrikeThrough = False

.Replacement.Font.Bold = False

.Replacement.Font.Italic = False

.Replacement.Style = ActiveDocument.Styles("Bold Italic")

If doHighlight = True Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Next nt

End If

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

rng.Font.StrikeThrough = False

End If

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

rng.Font.StrikeThrough = False

End If

Options.DefaultHighlightColorIndex = oldColour

Beep

End Sub

**Sub LtrRestore()**

' Paul Beverley - Version 29.03.24

' Restores text direction to left-to-right

If Selection.Start = Selection.End Then

myResponse = MsgBox("Restore the whole document?!", \_

vbQuestion + vbYesNo, "LtrRestore")

If myResponse <> vbYes Then Exit Sub

Selection.WholeStory

Selection.LtrPara

Selection.Collapse wdCollapseStart

Beep

Else

Selection.LtrPara

End If

End Sub

**Sub HighlightInContext()**

**'** Paul Beverley - Version 30.03.24

' Highlights a word or phrase in its context in a sentence/paragraph

myHighColour = wdBrightGreen

myContextColour = wdColorBlue

highlightParagraph = False

Set rngFind = Selection.Range.Duplicate

myEnd = rngFind.End

rngFind.Collapse wdCollapseStart

rngFind.Expand wdWord

myStart = rngFind.Start

rngFind.Select

rngFind.End = myEnd

rngFind.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rngFind.Text, 1)) > 0

rngFind.MoveEnd , -1

DoEvents

Loop

rngFind.Select

myFind = rngFind.Text

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myCount = myCount + 1

myEnd = rng.End

rng.HighlightColorIndex = myHighColour

If highlightParagraph = True Then

rng.Expand wdParagraph

Else

rng.Expand wdSentence

End If

rng.Font.Color = myContextColour

If myCount Mod 20 = 0 Then rng.Select

rng.End = myEnd

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

rngFind.Select

rngFind.Collapse wdCollapseEnd

MsgBox "Found: " & myCount

End Sub

**Sub CitationListSortByName()**

**'** Paul Beverley - Version 10.04.24

' Sorts items in a citation list in the text by name.

sortReversed = False

If Selection.Start = Selection.End Then

Selection.MoveStartUntil cset:="(", Count:=wdBackward

Selection.MoveEndUntil cset:=")", Count:=wdForward

Else

theEnd = Selection.End

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

theStart = Selection.Start

Selection.End = theEnd

Selection.Collapse wdCollapseEnd

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = theStart

End If

myList = Selection

myDelim = ", "

If InStr(myList, ";") > 0 Then myDelim = "; "

myBits = Split(myList, myDelim)

lastItem = UBound(myBits)

ReDim myGroup(lastItem) As String

myGroup = myBits

If sortReversed Then

WordBasic.SortArray myGroup(), 1

Else

WordBasic.SortArray myGroup(), 0

End If

tempStr = ""

For i = 0 To lastItem - 1

tempStr = tempStr & myGroup(i) & myDelim

Next i

tempStr = tempStr & myGroup(i)

Selection.TypeText tempStr

End Sub

**Sub ShiftUp()**

**'** Paul Beverley - Version 09.04.24

' Moves a list item up

If Selection.Start = Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Cut

rng.MoveStart , -1

rng.Expand wdParagraph

rng.Collapse wdCollapseStart

myStart = rng.Start

rng.Paste

rng.End = myStart

rng.Select

Else

End If

End Sub

**Sub ShiftDown()**

**'** Paul Beverley - Version 09.04.24

' Moves a list item up

If Selection.Start = Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Cut

rng.MoveEnd , 1

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

myStart = rng.Start

rng.Paste

rng.End = myStart

rng.Select

Else

End If

End Sub

**Sub ShiftOut()**

**'** Paul Beverley - Version 09.04.24

' Moves items out into a separate list

keyWord = "List"

wordsToAvoid = "switch,FRedit"

putCursorAtStart = True

Set mainDoc = ActiveDocument

Set rng = Selection.Range.Duplicate

If Selection.Start = Selection.End Then

rng.Expand wdParagraph

Else

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

myStart = rng.Start

rng.Start = Selection.End

rng.Expand wdParagraph

rng.Start = myStart

End If

wds = Split("," & LCase(wordsToAvoid), ",")

gottaList = False

For Each myDoc In Application.Documents

thisName = myDoc.Name

nm = LCase(thisName)

If InStr(nm, LCase(keyWord)) > 0 Then gottaList = True

For i = 1 To UBound(wds)

If InStr(nm, wds(i)) > 0 Then gottaList = False

Next i

If gottaList = True Then

Exit For

End If

Next myDoc

If gottaList = False Then

Beep

MsgBox "Can't find a list."

Exit Sub

End If

If myDoc.Name = mainDoc.Name Then

Beep

MsgBox "Place cursor in main document."

Exit Sub

End If

rng.Cut

myDoc.Activate

myStart = Selection.Start

Selection.Paste

If putCursorAtStart = True Then

Selection.Start = myStart

Selection.Collapse wdCollapseStart

End If

rng.Select

End Sub

**Sub ShiftIn()**

**'** Paul Beverley - Version 10.04.24

' Moves items from the list back into the current document

keyWord = "List"

wordsToAvoid = "switch,FRedit"

Set mainDoc = ActiveDocument

wds = Split("," & LCase(wordsToAvoid), ",")

gottaList = False

For Each myDoc In Application.Documents

thisName = myDoc.Name

nm = LCase(thisName)

If InStr(nm, LCase(keyWord)) > 0 Then gottaList = True

For i = 1 To UBound(wds)

If InStr(nm, wds(i)) > 0 Then gottaList = False

Next i

If gottaList = True Then

Exit For

End If

Next myDoc

If gottaList = False Then

Beep

MsgBox "Can't find a list."

Exit Sub

End If

If myDoc.Name = mainDoc.Name Then

Beep

MsgBox "Place cursor in main document."

Exit Sub

End If

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Collapse wdCollapseStart

rng.Select

myDoc.Activate

If InStr(Selection, vbCr) = 0 Then

Selection.Expand wdParagraph

If Len(Selection) < 3 Then

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

End If

Else

Set rng2 = Selection.Range.Duplicate

rng2.Collapse wdCollapseStart

rng2.Expand wdParagraph

myStart = rng2.Start

rng2.Start = Selection.End

rng2.Expand wdParagraph

rng2.Start = myStart

rng2.Select

End If

Selection.Cut

mainDoc.Activate

Selection.Paste

Selection.Collapse wdCollapseEnd

End Sub

**Sub CaseSecondNextWord()**

**'** Paul Beverley - Version 12.08.10

' Changes case of next-but-one word

trackIt = True

Set rng = ActiveDocument.Content

rng.Start = Selection.Start

rng.End = rng.Start

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[!a-zA-Z][0-9a-zA-Z]"

.Replacement.Text = ""

.Wrap = wdFindContinue

.Forward = True

.MatchWildcards = True

.Execute

End With

rng.Start = rng.End

rng.Find.Execute

rng.Select

If Selection.End < wasEnd + 2 Then

Selection.MoveRight Unit:=wdWord, Count:=1

Selection.MoveLeft Unit:=wdCharacter, Count:=1

End If

If trackIt = False Then

Selection.Range.Case = wdToggleCase

Selection.MoveRight Unit:=wdCharacter, Count:=1

Else

Selection.Start = Selection.Start + 1

myChar = Selection

If Asc(myChar) > 96 Then

myChar = UCase(myChar)

Else

myChar = LCase(myChar)

End If

Selection.Start = Selection.End

Selection.TypeBackspace

Selection.TypeText Text:=myChar

End If

End Sub

**Sub CopyAllComments()**

' Paul Beverley - Version 17.03.24

' Copies all the comments into a new document

If ActiveDocument.Comments.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdCommentsStory)

Documents.Add

Selection.FormattedText = rng.FormattedText

Else

Beep

MsgBox "No comments found!"

End If

End Sub

**Sub CopyComments2()**

**'** Paul Beverley - Version 01.06.10

' Create a list of all the comments

DocName = ActiveDocument.FullName

NewDocName = Replace(DocName, ".doc", "\_Cmnts.doc")

If ActiveDocument.Comments.Count >= 1 Then

ActiveDocument.StoryRanges(wdCommentsStory).Copy

End If

Documents.Add

Selection.Paste

ActiveDocument.SaveAs FileName:=NewDocName

End Sub

**Sub FindAdvanced3()**

**'** Paul Beverley - Version 01.06.11

' Call up Advanced Find dialog box

With Selection.Find

.MatchFuzzy = False

.MatchPhrase = False

.MatchAllWordForms = False

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.MatchCase = False

End With

Dialogs(wdDialogEditFind).Show

End Sub

**Sub HighlightRemovePerColour()**

**'** Paul Beverley - Version 03.04.24

' Removes highlight of one colour from selection or whole text

Dim myCol(20)

myCol(0) = wdNoHighlight

myCol(1) = wdBrightGreen

myCol(2) = wdYellow

myCol(3) = wdTurquoise

myCol(4) = wdPink

myCol(5) = wdRed

myCol(6) = wdGray50

myCol(7) = wdGray25

myColTotal = 7

allColours = "No highlight,Bright green,Yellow,Turquoise,Pink,Red,Gray50,Gray25"

myWord = Split(allColours, ",")

For i = 1 To UBound(myWord)

myPrompt = myPrompt & Trim(Str(i)) & " - " & myWord(i) & vbCr

Next i

If Selection.End = Selection.Start Then

myResponse = MsgBox("Remove highlight from the WHOLE file?", \_

vbQuestion + vbYesNo, "HighlightRemovePerColour")

If myResponse = vbNo Then Exit Sub

Set rng = ActiveDocument.Content

Else

Set rng = Selection.Range.Duplicate

End If

rngEnd = rng.End

rngStart = rng.Start

Do

myText = InputBox(myPrompt, "HighlightRemovePerColour")

myNumber = Val(myText)

If myNumber = 0 Then Beep: Exit Sub

DoEvents

Loop Until myNumber > 0 And myNumber <= myColTotal

myHighlight = myCol(myNumber)

With rng.Find

.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

End With

While rng.Find.Found = True

If rng.HighlightColorIndex = myHighlight Then

rng.HighlightColorIndex = wdNoHighlight

End If

rng.Collapse wdCollapseEnd

rng.End = rngEnd

rng.Find.Execute

' DoEvents

Wend

rng.End = rngEnd

rng.Start = rngStart

With rng.Find

.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

While .Found

If rng.HighlightColorIndex = 9999999 Then

For Each ch In rng.Characters

If ch.HighlightColorIndex = myHighlight Then \_

ch.HighlightColorIndex = wdNoHighlight

Next ch

End If

rng.Collapse wdCollapseEnd

rng.End = rngEnd

.Execute

DoEvents

Wend

End With

End Sub

**Sub MatchSquareBrackets()**

**'** Paul Beverley - Version 13.04.24

' Check whether brackets match up

stopEachTime = True

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

Set rngToGo = Selection.Range.Duplicate

rngToGo.End = ActiveDocument.Content.End

For Each myPara In rngToGo.Paragraphs

myText = myPara.Range.Text

If Len(myText) > 4 Then

myLen = Len(myText)

myOpenNum = myLen - Len(Replace(myText, "[", ""))

myCloseNum = myLen - Len(Replace(myText, "]", ""))

' Ignore "1)", or "a)" type lists

Set rng = myPara.Range

Debug.Print rng.Words(1) & rng.Words(2)

rng.End = rng.Start + 3

If InStr(rng.Text, "]") > 0 Then myCloseNum = myCloseNum - 1

If myOpenNum <> myCloseNum Then

If stopEachTime = True Then

Beep

rng.Expand wdParagraph

parenPos = InStr(rng.Text, "[")

If parenPos = 0 Then parenPos = InStr(rng.Text, "]")

If parenPos < 3 Then

rng.Start = rng.Start + 3

parenPos = InStr(rng.Text, "[")

If parenPos = 0 Then parenPos = InStr(rng.Text, "]")

End If

rng.Start = rng.Start + parenPos - 1

rng.End = rng.Start + 1

rng.Select

ActiveDocument.ActiveWindow.LargeScroll down:=1

rng.Select

ActiveDocument.ActiveWindow.SmallScroll down:=1

rng.Select

myResponse = MsgBox("Continue?", \_

vbQuestion + vbYesNo, "MatchParentheses")

If myResponse <> vbYes Then Exit Sub

Else

myPara.Range.Font.Underline = True

myCount = myCount + 1

StatusBar = "Found: " & myCount

DoEvents ' Debug.Print "Found: " & myCount

End If

End If

DoEvents

End If

Next

StatusBar = ""

Selection.HomeKey Unit:=wdStory

If myCount = 0 Then

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

MsgBox ("All clear!")

Else

MsgBox ("Number of suspect paragraphs: " & Trim(myCount))

End If

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Underline = True

.Replacement.Text = ""

.MatchWildcards = False

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute

End With

End Sub

**Sub HighlightOrColourRemove()**

**'** Paul Beverley - Version 10.08.24

' Removes the current highlight or font colour from the whole text

Selection.Collapse wdCollapseStart

Selection.MoveEnd , 1

Set rngWas = Selection.Range.Duplicate

myHighlight = Selection.Range.HighlightColorIndex

myFontColour = Selection.Range.Font.ColorIndex

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = ActiveDocument.Content

If Selection.Information(wdInEndnote) Then \_

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

If Selection.Information(wdInFootnote) Then \_

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

If myHighlight = wdNoHighlight Then

If myFontColour <> wdColorAutomatic And \_

myFontColour <> wdColorBlack Then

DoEvents

rngWas.selct

myResponse = MsgBox("Remove this font colour from this area?", \_

vbQuestion + vbYesNo, "HighlightOrColourRemove")

Selection.Collapse wdCollapseStart

If myResponse = vbNo Then

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.ColorIndex = myFontColour

.Wrap = wdFindContinue

.Replacement.Font.ColorIndex = wdColorBlack

.Execute Replace:=wdReplaceAll

DoEvents

End With

Beep

Else

myResponse = MsgBox("But this character has no highlight or colour!", \_

vbOK, "HighlightOrColourRemove")

End If

ActiveDocument.TrackRevisions = myTrack

Exit Sub

Else

myResponse = MsgBox("Remove this highlight from this area?", \_

vbQuestion + vbYesNo, "HighlightOrColourRemove")

DoEvents

Selection.Collapse wdCollapseStart

If myResponse = vbNo Then

ActiveDocument.TrackRevisions = myTrack

Exit Sub

End If

End If

rng.start = 0

rng.End = endDoc

With rng.Find

.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

While .Found

If rng.HighlightColorIndex = myHighlight Then

rng.HighlightColorIndex = wdNoHighlight

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Wend

End With

rng.start = 0

rng.End = endDoc

With rng.Find

.ClearFormatting

.Text = ""

.Highlight = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.Execute

While .Found

If rng.HighlightColorIndex = 9999999 Then

For Each ch In rng.Characters

If ch.HighlightColorIndex = myHighlight Then \_

ch.HighlightColorIndex = wdNoHighlight

Next ch

End If

rng.Collapse wdCollapseEnd

.Execute

DoEvents

Wend

End With

ActiveDocument.TrackRevisions = myTrack

Beep

End Sub

**Sub StyleLister()**

**'** Paul Beverley - Version 18.04.24

' Lists all paragraph and character styles used

CR = vbCr

paraStyles = CR & "Normal" & CR

chStyles = CR & "Default Paragraph Font" & CR

For myArea = 1 To 3

doThisArea = False

'Main text area

If myArea = 1 Then

If Selection.Start = Selection.End Then

myResponse = MsgBox("Scan the whole document?!", \_

vbQuestion + vbYesNo, "StyleLister")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

Else

Set rng = Selection.Range.Duplicate

End If

doThisArea = True

End If

' Footnotes, if any

If ActiveDocument.Footnotes.Count > 0 And myArea = 2 Then

doThisArea = True

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

StatusBar = "Scanning footnotes"

End If

' Endnotes, if any

If ActiveDocument.Endnotes.Count > 0 And myArea = 3 Then

doThisArea = True

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

StatusBar = "Scanning endnotes"

End If

If doThisArea = True Then

For Each pa In rng.Paragraphs

stName = pa.Style

If InStr(paraStyles, CR & stName & CR) = 0 Then \_

paraStyles = paraStyles & stName & CR

DoEvents

Next pa

i = rng.Characters.Count

For Each ch In rng.Characters

If myArea = 1 Then

i = i - 1

If i Mod 1000 = 0 Then

DoEvents

ch.Select

End If

End If

If ch.Fields.Count = 0 Then

stName = ch.CharacterStyle

If InStr(chStyles, CR & stName & CR) = 0 Then \_

chStyles = chStyles & stName & CR

End If

Next ch

End If

Next myArea

StatusBar = ""

Selection.HomeKey Unit:=wdStory

' Add para styles list & sort

Documents.Add

paraStyles = Mid(paraStyles, 9)

Selection.TypeText Text:=paraStyles

Set rng = ActiveDocument.Content

rng.Sort

paraStylesSorted = rng.Text

' Replace with chara styles list & sort

chStyles = Mid(chStyles, 25)

rng.Text = chStyles

rng.Sort

rng.InsertAfter Text:=CR

Selection.HomeKey Unit:=wdStory

myText = "Paragraph styles" & paraStylesSorted

myText = myText & CR & "Character styles"

Selection.TypeText Text:=myText

Set rng = ActiveDocument.Content

rng.Paragraphs(1).Range.Style = wdStyleHeading1

headingPos = InStr(rng, "Character styles")

rng.Start = headingPos

rng.Paragraphs(1).Range.Style = wdStyleHeading1

End Sub

**Sub GenusSpeciesAlyse()**

**'** Paul Beverley - Version 18.04.24

' Checks the abbreviation of genus/species names

okColour1 = wdGray25

okColour2 = wdGray50

warningColour1 = wdBrightGreen

warningColour2 = wdYellow

warningColour3 = wdRed

doAbbreviate = False

doAbbreviate = True

CR = vbCr

spList = CR

errorList = CR

For myArea = 1 To 3

doThisArea = False

' Main text area

If myArea = 1 Then

If Selection.Start = Selection.End Then

myResponse = MsgBox("Scan the whole document?!", \_

vbQuestion + vbYesNo, "StyleLister")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

Else

Set rng = Selection.Range.Duplicate

End If

doThisArea = True

End If

' Footnotes, if any

If ActiveDocument.Footnotes.Count > 0 And myArea = 2 Then

doThisArea = True

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

StatusBar = "Scanning footnotes"

End If

' Endnotes, if any

If ActiveDocument.Endnotes.Count > 0 And myArea = 3 Then

doThisArea = True

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

StatusBar = "Scanning endnotes"

End If

If doThisArea = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[A-Z][a-zA-Z.]{1,} [a-z]{1,}"

.Font.Italic = True

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

myText = rng.Text

If InStr(myText, ".") = 0 Then

spacePos = InStr(myText, " ")

abbrev = Left(myText, 1) & "." & Mid(myText, spacePos)

End If

' Debug.Print CR & spList

myCount = myCount + 1

If myCount Mod 20 = 0 Then rng.Select

If InStr(spList, CR & myText & CR) > 0 Then

If rng.Words(2) = ". " Then

rng.HighlightColorIndex = okColour1

Else

rng.HighlightColorIndex = warningColour1

If doAbbreviate = True Then rng.Text = abbrev

End If

Else

If rng.Words(2) = ". " Then

rng.HighlightColorIndex = warningColour2

errorList = errorList & myText & CR

Else

spList = spList & myText & CR & abbrev & CR

If InStr(errorList, CR & abbrev & CR) > 0 Then

rng.HighlightColorIndex = warningColour3

Else

rng.HighlightColorIndex = okColour2

End If

End If

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

End If

Next myArea

Selection.HomeKey Unit:=wdStory

Beep

End Sub

**Sub GoogleFetchFinland()**

**'** Paul Beverley - Version 23.04.24

' Launches selected text on Google

alsoCopySubject = True

mySite = "http://www.google.fi/search?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

If alsoCopySubject = True Then Selection.Copy

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

Debug.Print mySite & mySubject

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

End Sub

**Sub SpellAlyseToFRedit()**

**'** Paul Beverley - Version 23.04.24

' Corrects spelling and wrong capitalisation

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

If Len(rng) < 3 Then

rng.MoveEnd , -3

rng.Expand wdWord

End If

myWord = rng.Text

' Check spelling language

' (check only first character, in case of split language)

langName = Languages(rng.Characters(1).LanguageID).NameLocal

' Check spelling

spellOK = Application.CheckSpelling(myWord, MainDictionary:=langName)

Debug.Print myWord

If spellOK Then

Beep

MsgBox "No change: """ & myWord & \_

""" is a correct spelling in " & langName & "."

Exit Sub

Else

Set suggList = rng.GetSpellingSuggestions(myWord, \_

MainDictionary:=langName)

If suggList.Count > 0 Then

newWord = suggList.Item(1).Name

Debug.Print myWord & "|" & newWord

fghdfg = 0

If newWord <> myWord Then

rng.Text = ChrW(172) & myWord & "|" & newWord

Else

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 1

Beep

If langName = "English (United States)" \_

And Application.CheckSpelling(myWord, \_

MainDictionary:="English (United Kingdom)") Then

MsgBox ("Word's spellchecker is playing sillies!")

Exit Sub

Else

MsgBox ("Word offers no suggestion")

End If

End If

Else

Beep

MsgBox "Word offers no alternative! (Empty suggList)"

End If

End If

rng.MoveEnd , 2

rng.Collapse wdCollapseEnd

rng.Select

End Sub

**Sub FReditListTidy()**

**'** Paul Beverley - Version 24.04.24

' Tidies up a FRedit list

Set rng = ActiveDocument.Content

myTest = rng

myTest = Replace(myTest, " ", "")

If InStr(rng, "|FRedit") = 0 Then rng.InsertBefore \_

Text:="| FRedit" & vbCr & vbCr

For i = ActiveDocument.Paragraphs.Count To 1 Step -1

myText = ActiveDocument.Paragraphs(i).Range.Text

If Len(myText) > 3 And InStr(myText, ChrW(124)) = 0 Then \_

ActiveDocument.Paragraphs(i).Range.Delete

DoEvents

Next i

End Sub

**Sub CommentCollector()**

**'** Paul Beverley - Version 30.09.24

' Creates a table of all comments

doLandscape = True

myResponseSpace = 20

titleSize = 16

myResponse = MsgBox("Include formatting? (Takes 2-3 times as long.)", \_

vbQuestion + vbYesNoCancel, "CommentCollector")

If myResponse = vbCancel Then Exit Sub

includeFormatting = (myResponse = vbYes)

t = Timer

Dim myCol(10) As String

myCol(1) = "number #"

myCol(2) = "scope Scope"

myCol(3) = "comment Comment"

myCol(4) = "page Pg"

myCol(5) = "author blank"

myCol(6) = "response Author comments"

totCols = 6

CR = vbCr

TB = vbTab

Set src = ActiveDocument

Application.ScreenUpdating = False

On Error GoTo ReportIt

Documents.Add

If doLandscape = True Then Selection.PageSetup.Orientation = wdOrientLandscape

' Build up a space-reserving text

numExtra = myResponseSpace - 8

If numExtra < 1 Then numExtra = 1

For i = 1 To numExtra

extraSpace = extraSpace & " " & ChrW(160)

Next i

For j = 1 To totCols

myHead = myCol(j)

spPos = InStr(myHead, " ")

addExtra = (Left(myHead, 8) = "response")

myHead = Mid(myHead, spPos + 1)

If myHead = "blank" Then myHead = ""

If addExtra Then myHead = myHead & extraSpace

Selection.TypeText myHead & TB

Next j

Selection.MoveStart , -1

Selection.TypeText CR

For i = 1 To src.Comments.Count

For j = 1 To totCols

doWhat = myCol(j)

spPos = InStr(doWhat, " ")

doWhat = Left(doWhat, spPos - 1)

Select Case LCase(doWhat)

Case "number":

Selection.InsertAfter Text:=src.Comments(i).Index & TB

Selection.Style = ActiveDocument.Styles("Normal")

Selection.Collapse wdCollapseEnd

Case "comment":

Set rng = src.Comments(i).Range

If includeFormatting = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "zczc"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

rng.Copy

.Text = "zczc"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

End With

Selection.Paste

Else

Selection.TypeText Text:=src.Comments(i).Range.Text

End If

Selection.TypeText Text:=TB

Case "author"

Selection.InsertAfter Text:=src.Comments(i).Contact & TB

Selection.Style = ActiveDocument.Styles("Normal")

Selection.Collapse wdCollapseEnd

Case "scope"

Selection.InsertAfter Text:=src.Comments(i).Scope & TB

Selection.Style = ActiveDocument.Styles("Normal")

Selection.Collapse wdCollapseEnd

Case "page"

Set scp = src.Comments(i).Scope

pg = Trim(Str(scp.Information(wdActiveEndAdjustedPageNumber)))

Selection.InsertAfter Text:=pg & TB

Selection.Style = ActiveDocument.Styles("Normal")

Selection.Collapse wdCollapseEnd

End Select

Next j

DoEvents

Selection.TypeText Text:=CR

Next i

If Asc(Right(src.Comments(src.Comments.Count).Range.Text, 1)) = 13 Then

src.Comments(src.Comments.Count).Range.Characters.Last = ""

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^t^p"

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Comments summary" & CR

Set rng = ActiveDocument.Content

rng.start = Selection.End

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

ActiveDocument.Paragraphs(2).Range.Font.Bold = True

ActiveDocument.Paragraphs(2).Range.Font.Size = titleSize

rng.ConvertToTable Separator:=wdSeparateByTabs

rng.Tables(1).AutoFitBehavior (wdAutoFitContent)

With rng.Find

.Text = "zczc"

.Replacement.Text = "^l"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = blankItem

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

DoEvents

End With

Application.ScreenUpdating = True

MsgBox "Time: " & Str(Int(Timer - t)) & " secs"

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub DocumentAddMyBlank()**

**'** Paul Beverley - Version 20.04.24

' Adds document at specific window size, based on my template

Documents.Add Template:= \_

"C:\Users\Paul\AppData\Roaming\Microsoft\Templates\MyBlank.dotx", \_

NewTemplate:=False, DocumentType:=0

ActiveDocument.Windows(1).WindowState = wdWindowStateNormal

Application.Resize Width:=1000, Height:=500

Application.ActiveWindow.View.Zoom.Percentage = 200

End Sub

**Sub FReditCommentToggle()**

**'** Paul Beverley - Version 17.04.24

' Adds or removes the vertical bar that turns FRedit items into comments

' This macro assumes that you've selected whole lines

' But you could add some checking code

FReditComment = "| "

Set rng = Selection.Range.Duplicate

' Include the previous New Line

rng.MoveStart , -1

rng.MoveEnd , -1

pipeFound = InStr(Selection, vbCr & FReditComment) > 0

If pipeFound Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p" & FReditComment

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

Else

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = "^p" & FReditComment

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

rng.MoveStart , 1

rng.MoveEnd , 1

rng.Select

End Sub

**Sub FRskip()**

**'** Paul Beverley - Version 04.05.24

' Finds next match forwards, case insensitively

myspaces = " "

For i = 1 To 3

myspaces = myspaces & myspaces

Next i

mySch = Selection.Find.Text

thisIsWild = ((InStr(mySch, "]") + InStr(myScht, "<") \_

+ InStr(mySch, ">") > 0))

If thisIsWild Then StatusBar = myspaces & "Using WILDCARD find"

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseEnd

With Selection.Find

.Wrap = wdFindStop

.Forward = True

.MatchCase = False

.MatchWildcards = thisIsWild

.Execute

' Leave F&R dialogue in a sensible state

.Wrap = wdFindContinue

End With

If Selection.Find.Found = False Then

Beep

Else

If Selection.End = 0 Then

rng.Select

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Selection.EndKey Unit:=wdStory

With Selection.Find

.Wrap = wdFindStop

.Forward = False

.Execute

' Leave F&R dialogue in a sensible state

.Wrap = wdFindContinue

.Forward = True

End With

StatusBar = "Sorry, Word's Find facility is playing sillies!"

Else

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseStart

Selection.MoveUp , 1

rng.Select

End If

End If

End Sub

**Sub FRstay()**

' Paul Beverley - Version 04.05.24

' Finds and replaces but don't move to next

Call MacroNameShow("FRstay")

Selection.Collapse wdCollapseStart

Selection.Find.Execute Replace:=wdReplaceOne

End Sub

**Sub FRgo()**

' Paul Beverley - Version 04.05.24

' Finds, replaces and moves to the next match

Call MacroNameShow("FRgo")

hereNow = Selection.End

If Selection.End = Selection.Start Then

With Selection.Find

.Wrap = wdFindStop

.Forward = True

.Execute

End With

Else

Selection.End = Selection.Start

With Selection.Find

.Execute Replace:=wdReplaceOne

.Wrap = wdFindStop

.Forward = True

End With

Selection.Start = Selection.End

Selection.Find.Execute

End If

If Selection.Start = hereNow And \_

Selection.Find.Found = False Then Beep

' Leave F&R dialogue in a sensible state

Selection.Find.Wrap = wdFindContinue

End Sub

**Sub FRselective()**

**'** Paul Beverley - Version 02.10.24

' Sets up various Finds and F&Rs

CR2 = vbCr & vbCr

Set para = Selection.Range.Duplicate

' Check if cursor is at start of a para

para.Expand wdParagraph

If para.start = Selection.start Then

myText = Selection.Find.Text

wcPos = InStr(myText, "[!^13]@")

If wcPos > 0 Then

myLast = Left(myText, wcPos - 1)

myFirst = Mid(myText, wcPos + 7)

mySearch = myFirst & "[!^13]@" & myLast

doWC = True

GoTo doSearch

End If

End If

' Check if it's a two-word para

para.MoveEnd , -1

If para.Words.Count = 2 Then

myFirst = Trim(para.Words(1))

myInit = Left(myFirst, 1)

myFirst = "[" & LCase(myInit) & UCase(myInit) \_

& "]" & Mid(myFirst, 2)

myLast = Trim(para.Words(2))

myInit = Left(myLast, 1)

myLast = "[" & LCase(myInit) & UCase(myInit) \_

& "]" & Mid(myLast, 2)

mySearch = myFirst & "[!^13]@" & myLast

doWC = True

GoTo doSearch

End If

' If nothing is selected try to process as an F&R

If Selection.start = Selection.End Then

mySearch = para.Text

doWC = True

' If a tilde at the beginning

If Left(mySearch, 1) = ChrW(126) Then

mySearch = Mid(mySearch, 2)

End If

' A vertical bar (pipe) means it's an F&R

' so split into F and R

myReplace = ""

If InStr(mySearch, ChrW(124)) > 0 Then

pipePos = InStr(mySearch, ChrW(124))

myReplace = Mid(mySearch, pipePos + 1)

mySearch = Left(mySearch, pipePos - 1)

End If

para.Select

' Warn if too long

If Len(mySearch) > 254 Then

Beep

myResponse = MsgBox("Search text too long!", \_

vbOKOnly, "FRselective")

Exit Sub

End If

GoTo doSearch

End If

' Make up a two-word search

Set rng = Selection.Range.Duplicate

If rng.Words.Count > 1 Then

rng.Collapse wdCollapseStart

' First word

rng.Expand wdWord

myStart = rng.start

' Last word

rng.start = Selection.End

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

rng.start = myStart

' Make up the two-word WC Find

myFirst = Trim(rng.Words.First)

myInit = Left(myFirst, 1)

myFirst = "[" & LCase(myInit) & UCase(myInit) \_

& "]" & Mid(myFirst, 2)

myLast = Trim(rng.Words.Last)

If UCase(myLast) <> LCase(myLast) Or Len(myLast) > 1 Then

myInit = Left(myLast, 1)

myLast = "[" & LCase(myInit) & UCase(myInit) \_

& "]" & Mid(myLast, 2)

mySearch = myFirst & "[!^13]@" & myLast

Else

mySearch = myFirst

End If

Else

myFirst = Trim(rng.Words.First)

myInit = Left(myFirst, 1)

mySearch = "[" & LCase(myInit) & UCase(myInit) \_

& "]" & Mid(myFirst, 2)

End If

doWC = True

doSearch:

On Error GoTo ReportIt

Selection.Collapse wdCollapseEnd

Set rng = Selection.Range.Duplicate

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = mySearch

.Replacement.Text = myReplace

.MatchWildcards = doWC

.Forward = True

.Execute

If (Selection.End = 0 And Selection.start > 0) Then

Beep

MsgBox "Word is playing sillies, sorry!"

If ActiveDocument.TablesOfContents.Count > 0 Then

MsgBox "Word is falling over the content list, sorry!"

End If

Exit Sub

End If

If .Found = False Then

wcPos = InStr(mySearch, "[!^13]@")

myPrompt = mySearch

If wcPos > 0 Then

myFirst = Left(mySearch, wcPos - 1)

myLast = Mid(mySearch, wcPos + 7)

myNewSearch = myLast & "[!^13]@" & myFirst

myFirst = Mid(myFirst, 2, 1) & Mid(myFirst, 5)

myLast = Mid(myLast, 2, 1) & Mid(myLast, 5)

myPrompt = myFirst & " FOLLOWED BY " & myLast

myPrompt = myPrompt & CR2 & "Try the reverse order?"

myResponse = MsgBox(myPrompt, \_

vbQuestion + vbYesNoCancel, "FRselective")

If myResponse = vbYes Then

rng.Select

.Text = myNewSearch

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

If Selection.End < Selection.start Then

' VBA's gone bonkers

rng.Select

Beep

Exit Sub

End If

If .Found = False Then

rng.Select

Beep

.Text = myNewSearch

.Wrap = wdFindStop

.Forward = False

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

If .Found = False Then

Beep

rng.Select

Exit Sub

End If

End If

Else

Beep

End If

Exit Sub

End If

Beep

myPrompt = "Can't find:" & CR2 & myPrompt

myResponse = MsgBox(myPrompt, vbOKOnly, "FRselective")

rng.Select

End If

End With

Exit Sub

ReportIt:

If Err.Number = 5560 Then

Beep

MsgBox ("Bad pattern match!")

Else

On Error GoTo 0

Resume

End If

End Sub

**Sub FourDigitFixer()**

**'** Paul Beverley - Version 10.05.24

' Adds a delimiter to four-digit numbers, but not dates

' deLimiter = ","

deLimiter = ChrW(8201): ' thin space = hex 2009

minDate = 1900

maxDate = 2100

' Adjust to taste!

myHighColour = wdYellow

' myHighColour = 0 : ' no highlight, thanks

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[0-9]{4}>"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

If myCount Mod 20 = 0 Then rng.Select

myDate = Val(rng.Text)

If myDate < minDate Or myDate > maxDate Then

rng.Text = Left(rng.Text, 1) & deLimiter & Right(rng.Text, 3)

If myHighColour > 0 Then rng.HighlightColorIndex = myHighColour

myCount = myCount + 1

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Selection.HomeKey Unit:=wdStory

Beep

MsgBox "Changed: " & myCount

End Sub

**Sub LinksToURLtextGlobal()**

**'** Paul Beverley - Version 17.05.24

' Converts URL links from descriptive text to actual URL address

' asdfa = ActiveDocument.Fields(1).

For hit = 1 To 3

doIt = False

If hit = 1 Then

If ActiveDocument.Endnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

doIt = True

End If

End If

If hit = 2 Then

If ActiveDocument.Footnotes.Count > 0 Then

Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

doIt = True

End If

End If

If hit = 3 Then

Set rng = ActiveDocument.Range

doIt = True

End If

DoEvents

If doIt = True Then

xcvbcsv = rng.Fields.Count

For i = rng.Fields.Count To 1 Step -1

myType = rng.Fields(i).Type

If myType = 88 Then

myURLtext = rng.Fields(i).Code

myDisplayText = rng.Fields(i).Result

myURLtext = Replace(myURLtext, "HYPERLINK """, "")

qtPos = InStr(myURLtext, """")

myURLtext = Trim(Left(myURLtext, qtPos - 1))

myURLtext = Replace(myURLtext, remove1, "")

myURLtext = Replace(myURLtext, remove2, "")

Debug.Print myDisplayText

Debug.Print myURLtext & vbCr

If myDisplayText <> myURLtext Then

rng.Fields(i).Select

Selection.Collapse wdCollapseEnd

rng.Fields(i).Delete

Selection.InsertAfter Text:=myURLtext

Selection.Hyperlinks.Add \_

Anchor:=Selection.Range, Address:=myURLtext

End If

End If

Next i

End If

Next hit

End Sub

**Sub DuplicateSentenceCount()**

**'** Paul Beverley - Version 21.05.21

' Counts frequency of any duplicated sentences + highlights

minWords = 10

myTab = " . . . "

myColour = wdBrightGreen

dupSents = ""

myExtraLines = ""

sentNum = ActiveDocument.Sentences.Count

If sentNum > 2000 Then splitLevel = 150 \_

Else splitLevel = 60

Selection.HomeKey Unit:=wdStory

Set myDoc = ActiveDocument.Content

Dim num(27) As Integer

' Find out how many sentences there are for each

' letter of the alphabet

For Each sent In myDoc.Sentences

init = Left(Trim(sent.Text), 1)

If InStr("ABCDEFGHIJKLMNOPQRSTUVWXYZ", init) > 0 Then

n = Asc(init) - 64

num(n) = num(n) + 1

Else

num(27) = num(27) + 1

End If

If Asc(sent) = Asc("|") Then

parenPos = InStr(sent, " (")

addBit = Left(sent, parenPos - 1)

myExtraLines = myExtraLines & addBit & vbCr

End If

Next sent

Documents.Add

Set myFinal = ActiveDocument

Documents.Add

' If there are a lot of sentences for a given letter

' then split them up based on a second character.

' I randomly chose the fifth letter (rather than

' the more obvious second character

For a = Asc("A") To Asc("Z") + 1

If num(a - 64) > splitLevel Then

' Split up on fifth letter

For b = Asc("A") To Asc("Z") + 1

Set tmp = ActiveDocument.Content

If b = 91 Then

' 91 is the next number to Z's ASCII value

' Use it to catch then non-A-Z letters,

' such as parentheses and bullets

For Each sent In myDoc.Sentences

If sent.Words.Count > minWords And \_

UCase(sent) <> LCase(sent) Then

thisSent = Replace(Trim(sent.Text), vbCr, "")

' CHeck the fifth letter

myTest = UCase(Mid(thisSent, 5, 1))

q = Asc(thisSent)

If q = a And LCase(myTest) = myTest Then

Selection.TypeText Text:=Trim(thisSent) & vbCr

DoEvents

End If

k = Asc(myTest)

' Allow chars in the range 34 to 64, or over 90 (q)

' And similarly for the fifth character (k)

If a = 91 And (q < 65 Or q > 90) And q > 33 And \_

(k < 65 Or k > 90) And k > 33 Then

Selection.TypeText Text:=Trim(thisSent) & vbCr

DoEvents

End If

End If

DoEvents

Next sent

Else

For Each sent In myDoc.Sentences

' This is where we just test all of a set of lettered

' sentences when there aren't too many of them.

If sent.Words.Count > minWords Then

thisSent = Replace(Trim(sent.Text), vbCr, "")

myTest = UCase(Mid(thisSent, 5, 1))

q = Asc(thisSent)

If q = a And myTest = Chr(b) Then

Selection.TypeText Text:=Trim(thisSent) & vbCr

DoEvents

End If

If a = 91 And (q < 65 Or q > 90) And q > 33 And \_

myTest = Chr(b) Then

Selection.TypeText Text:=Trim(thisSent) & vbCr

DoEvents

End If

End If

Next sent

DoEvents

End If

numSents = ActiveDocument.Sentences.Count

dupSents = ""

For i = 1 To numSents

If tmp.Sentences(i).Words.Count > minWords Then

testSent = Replace(Trim(tmp.Sentences(i).Text), vbCr, "")

If InStr(dupSents, testSent) = 0 And Len(testSent) > 10 Then

myCount = 1

For j = i + 1 To numSents

DoEvents

compSent = Replace(Trim(tmp.Sentences(j).Text), vbCr, "")

If compSent = testSent Then

myCount = myCount + 1

End If

Next j

StatusBar = numSents - i

If myCount > 1 Then

DoEvents

StatusBar = testSent

' Create the lines that list the duplicated sentences

' giving the frequency of each multiple sentence

sentPlusCount = testSent & myTab & Trim(Str(myCount)) \_

& vbCr & vbCr

dupSents = dupSents + sentPlusCount

End If

End If

End If

Next i

If dupSents > "" Then myFinal.Content.InsertAfter Text:=dupSents

tmp.Text = ""

DoEvents

Next b

Else

Set tmp = ActiveDocument.Content

For Each sent In myDoc.Sentences

thisSent = Trim(Replace(sent.Text, vbCr, "")) & vbCr

q = Asc(thisSent)

If q = a Then

Selection.TypeText Text:=thisSent

DoEvents

End If

If a = 91 And (q < 65 Or q > 90) And q > 32 Then

Selection.TypeText Text:=thisSent

DoEvents

End If

Next sent

' Check to see if there any duplicate sentences

' in the current batch.

numSents = ActiveDocument.Sentences.Count

dupSents = ""

For i = 1 To numSents

If tmp.Sentences(i).Words.Count > minWords Then

testSent = Replace(tmp.Sentences(i).Text, vbCr, "")

If InStr(dupSents, testSent) = 0 And Len(testSent) > 10 Then

myCount = 1

For j = i + 1 To numSents

DoEvents

compSent = Replace(tmp.Sentences(j).Text, vbCr, "")

If compSent = testSent Then

myCount = myCount + 1

End If

Next j

StatusBar = numSents - i

If myCount > 1 Then

DoEvents

StatusBar = testSent

sentPlusCount = testSent & myTab & Trim(Str(myCount)) \_

& vbCr & vbCr

dupSents = dupSents + sentPlusCount

End If

End If

End If

DoEvents

Next i

If dupSents > "" Then myFinal.Content.InsertAfter Text:=dupSents

tmp.Text = ""

End If

Next a

Selection.InsertAfter Text:=myFinal.Content

Selection.Collapse wdCollapseStart

' Create the two display files

' This one creates the FRedit list

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myTab & "[0-9]{1,}" & vbCr

.Wrap = wdFindContinue

.Replacement.Text = "|^^&"

.MatchWildcards = True

.MatchWholeWord = False

.MatchSoundsLike = False

.Execute Replace:=wdReplaceAll

End With

gogo = True

Do While gogo = True

gogo = False

' But if any of the FRedit items is too long, it'll

' cause an error in FRedit, so split them into sub-items

For i = ActiveDocument.Paragraphs.Count To 1 Step -1

myLine = ActiveDocument.Paragraphs(i).Range.Text

myLen = Len(myLine)

If myLen > 200 Then

gogo = True

myLeft = Int(myLen / 2)

newLine1 = Left(myLine, myLeft) & "|^&"

newLine2 = Mid(myLine, myLeft + 1)

ActiveDocument.Paragraphs(i).Range.Text = newLine1 \_

& vbCr & newLine2

End If

DoEvents

Next i

Loop

Set rng = ActiveDocument.Content

rng.HighlightColorIndex = myColour

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="|!FRedit" & vbCr & vbCr

ActiveDocument.Paragraphs(1).Range.HighlightColorIndex \_

= wdNoHighlight

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:=myExtraLines

myFinal.Activate

Beep

End Sub

**Sub QuoteSingleDoubleSwitchGlobal()**

**'** Paul Beverley - Version 15.05.24

' Changes double quotes to single and vice versa, avoiding apostrophes

myColour = wdBrightGreen

myFind = """,',!t,!ve,!m,!n,!s,!d,!r,!l,s!,',zczc"

myFind = Replace(myFind, "!", ChrW(8217))

myReplace = myReplace & "zczc,',^&,^&,^&,^&,^&,^&,^&,^&,^&,"",'"

fnd = Split(myFind, ",")

rpl = Split(myReplace, ",")

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

Set rngT = ActiveDocument.Content

If ActiveDocument.Footnotes.Count > 0 Then \_

Set rngF = ActiveDocument.StoryRanges(wdFootnotesStory)

If ActiveDocument.Endnotes.Count > 0 Then \_

Set rngE = ActiveDocument.StoryRanges(wdEndnotesStory)

For i = 0 To UBound(fnd)

Debug.Print fnd(i), rpl(i)

With rngT.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

If rpl(i) = """" Then .Font.StrikeThrough = False

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

If rpl(i) = "^&" Then .Replacement.Font.StrikeThrough = True

If Asc(fnd(i)) = Asc("s") Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

If ActiveDocument.Footnotes.Count > 0 Then

With rngF.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

If rpl(i) = """" Then .Font.StrikeThrough = False

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

If rpl(i) = "^&" Then .Replacement.Font.StrikeThrough = True

If Asc(fnd(i)) = Asc("s") Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

If ActiveDocument.Endnotes.Count > 0 Then

With rngE.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = fnd(i)

If rpl(i) = """" Then .Font.StrikeThrough = False

.Wrap = wdFindContinue

.Replacement.Text = rpl(i)

.MatchCase = False

.MatchWildcards = False

If rpl(i) = "^&" Then .Replacement.Font.StrikeThrough = True

If Asc(fnd(i)) = Asc("s") Then .Replacement.Highlight = True

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

Next i

rngT.Font.StrikeThrough = False

If ActiveDocument.Footnotes.Count > 0 Then \_

rngF.Font.StrikeThrough = False

If ActiveDocument.Endnotes.Count > 0 Then \_

rngE.Font.StrikeThrough = False

Options.DefaultHighlightColorIndex = oldColour

End Sub

**Sub SentenceHighlight()**

**'** Paul Beverley - Version 25.05.24

' Creates a list of all styles name

If Selection.Start = Selection.End Then

Set rng = ActiveDocument.Content

Else

Set rng = Selection.Range.Duplicate

End If

If rng.Font.DoubleStrikeThrough <> False Then

foundDstrike = False

For i = 2 To rng.Sentences.Count

If rng.Sentences(i).Font.DoubleStrikeThrough <> False Then

foundDstrike = True

Exit For

Beep

End If

If i Mod 20 = 0 Then DoEvents

Next i

If foundDstrike = True Then

rng.Font.DoubleStrikeThrough = False

Beep

Exit Sub

End If

End If

For Each sn In rng.Sentences

i = i + 1

If i Mod 2 = 0 Then

sn.Font.DoubleStrikeThrough = True

End If

DoEvents

Next sn

Beep

End Sub

**Sub GoToTop()**

**'** Paul Beverley - Version 31.05.24

' Moves the cursor to the top of the document (for Mac users)

Selection.HomeKey Unit:=wdStory

End Sub

**Sub GoToBottom()**

' Paul Beverley - Version 31.05.24

' Moves the cursor to the bottom of the document (for Mac users)

Selection.EndKey Unit:=wdStory

End Sub

**Sub SelectToTop()**

' Paul Beverley - Version 31.05.24

' Selects all the text from the cursor to the top of the document (for Mac users)

Selection.HomeKey Unit:=wdStory, Extend:=wdExtend

End Sub

**Sub SelectToBottom()**

' Paul Beverley - Version 31.05.24

' Selects all the text from the cursor to the bottom of the document (for Mac users)

Selection.EndKey Unit:=wdStory, Extend:=wdExtend

End Sub

**Sub ProperNounsInSectionsMultiword()**

**'** Paul Beverley - Version 30.01.23

' Collects multi-word proper nouns by section and alpha lists

myHeading = "Heading 1"

minLength = 3

notWords = " and for the this there those their they then these that "

notWords = notWords & " when you "

Dim headStart(1000) As Long

Dim textStart(1000) As Long

pf = Split(" " & Trim(notPrefixes), " ")

CR = vbCr

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Style = ActiveDocument.Styles(myHeading)

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

' If you want to count them...

myCount = myCount + 1

' Note where the end of the found item is

headStart(myCount) = rng.Start

textStart(myCount) = rng.End

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

headStart(myCount + 1) = ActiveDocument.Content.End

myResponse = MsgBox("Found: " & Trim(Str(myCount)) & \_

" Headings." & vbCr & vbCr & "Continue?", vbQuestion \_

+ vbYesNoCancel, "ProperNounsInSectionsMultiword")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

Documents.Add

Set rng2 = ActiveDocument.Content

Set rng3 = ActiveDocument.Content

For i = 1 To myCount

rng.Start = headStart(i)

rng.End = textStart(i)

rng.Copy

rng2.Paste

rng2.Collapse wdCollapseEnd

rng2.Select

rng.Start = textStart(i)

rng.End = headStart(i + 1)

rng3.Start = rng2.End - 1

rng3.End = ActiveDocument.Content.End

myEnd = rng.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "from"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True And rng.Start < myEnd

Debug.Print rng.Start, rng.End, myEnd

rng.Select

rng.HighlightColorIndex = wdBrightGreen

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

rng2.End = ActiveDocument.Content.End

rng2.Sort

rng2.Characters(1).Delete

rng2.Collapse wdCollapseEnd

rng2.InsertAfter Text:=CR & CR

rng2.Start = ActiveDocument.Content.End

Next i

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub CommentTableMaker()**

**'** Paul Beverley - Version 09.05.24

' Creates a table of all comments

doLandscape = True

myResponseSpace = 20

titleSize = 16

' Standard response from user

myResponse = MsgBox("Include formatting? (Takes 2-3 times as long.)", \_

vbQuestion + vbYesNoCancel, "CommentTableMake")

If myResponse = vbCancel Then Exit Sub

includeFormatting = (myResponse = vbYes)

t = Timer

Dim myCol(10) As String

myCol(1) = "number #"

myCol(2) = "comment Comment"

myCol(3) = "page Pg"

myCol(4) = "author blank"

myCol(5) = "response Author comments"

totCols = 5

CR = vbCr

TB = vbTab

Set src = ActiveDocument

myTrack = src.TrackRevisions

' Switch tracking off if it's not wanted

src.TrackRevisions = False

Application.ScreenUpdating = False

On Error GoTo ReportIt

numComments = src.Comments.Count

Set myDoc = Documents.Add

Set myTable = myDoc.Tables.Add(Range:=Selection.Range, \_

NumRows:=numComments + 1, NumColumns:=totCols)

' Build up a space-reserving text

numExtra = myResponseSpace - 8

If numExtra < 1 Then numExtra = 1

For i = 1 To numExtra

extraSpace = extraSpace & " " & ChrW(160)

Next i

For i = 1 To totCols

myHead = myCol(i)

With myTable.Rows(1)

If myHead = "blank" Then myHead = ""

addExtra = (Left(myHead, 8) = "response")

myHead = Mid(myHead, spPos + 1)

If addExtra Then myHead = myHead & extraSpace

' Debug.Print "|" & myHead & "|"

If myHead > "" Then \_

.Cells(i).Range.Text = myHead

.Range.Font.Bold = True

End With

Next i

myTable.AutoFitBehavior (wdAutoFitContent)

For myCmt = 1 To numComments

For myCell = 1 To totCols

myHead = myCol(myCell)

spPos = InStr(myHead, " ")

' Prepare the pointer to column content

myContent = Left(myHead, spPos - 1)

Select Case LCase(myContent)

Case "number":

myTable.Rows(myCmt + 1).Cells(myCell).Range.Text = \_

Trim(Str(src.Comments(myCmt).Index))

Case "comment":

Set rng = src.Comments(myCmt).Range

' Debug.Print rng.Text & "|"

If includeFormatting = True Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "zczc"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

rng.Copy

.Text = "zczc"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

Debug.Print rng.Text & "|"

End With

' If rng.Characters.Last = CR Then rng.Characters.Last.Delete

myTable.Rows(myCmt + 1).Cells(myCell).Range.Paste

Else

myTable.Rows(myCmt + 1).Cells(myCell).Range.Text = src.Comments(myCmt).Range.Text

End If

Case "author"

myTable.Rows(myCmt + 1).Cells(myCell).Range.Text = src.Comments(myCmt).Contact

Case "page"

Set scp = src.Comments(myCmt).Scope

pg = Trim(Str(scp.Information(wdActiveEndAdjustedPageNumber)))

myTable.Rows(myCmt + 1).Cells(myCell).Range.Text = pg

End Select

Next

DoEvents

Next myCmt

MsgBox "Time: " & Str(Int(Timer - t)) & " secs"

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub ClipboardSave()**

**'** Paul Beverley - Version 15.06.24

' Saves contents of clipboard as a new file in your temp folder

leaveFileOpen = False

myName = "tmp"

' Choose your temporary folder:

' On a Mac, it will need to be something like this:

tempFolder = "/Users/Paul/My Documents/Temp\_files/"

' On Windows, it will need to be something like this:

tempFolder = "C:\Documents and Settings\Paul\Temp\_files\"

Randomize 10

ltrOne = Chr(64 + Int(26 \* Rnd() + 1))

ltrTwo = Chr(64 + Int(26 \* Rnd() + 1))

myName = myName & "\_" & ltrOne & ltrTwo

Documents.Add

Selection.Paste

Selection.HomeKey Unit:=wdStory

For i = 1 To 10

DoEvents

Next i

myNewFile = tempFolder & myName

myWords = ActiveDocument.Words.Count

' myResponse = MsgBox("Save file with " & Str(myWords) & " words?", \_

vbQuestion + vbYesNoCancel, "ClipboardSave")

' If myResponse <> vbYes Then Exit Sub

ActiveDocument.SaveAs fileName:=myNewFile

If leaveFileOpen = False Then ActiveDocument.Close SaveChanges:=wdDoNotSaveChanges

End Sub

**Sub DocumentCleaner()**

**'** Paul Beverley - Version 19.06.24

' Removes styles and fonts but preserves bold, italic, highlights and font colours

doBoldItalAsStyle = False

myItalStyle = "Emphasis"

myBoldStyle = "Strong"

Set rngOld = ActiveDocument.Content

Documents.Add

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

' Code italic and bold

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.Italic = True

.Wrap = wdFindContinue

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Replacement.Text = "[[^&]]"

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Text = ""

.Font.Bold = True

.Replacement.Text = "{{^&}}"

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Text = ""

.Font.Superscript = True

.Replacement.Text = "\\^&;;"

.Execute Replace:=wdReplaceAll

.ClearFormatting

.Text = ""

.Font.Subscript = True

.Replacement.Text = "//^&::"

.Execute Replace:=wdReplaceAll

End With

' Codes all the colours

Set rng = ActiveDocument.Content

rng.InsertBefore Text:="Dummy" & vbCr

rng.InsertAfter Text:="<<Dummy" & vbCr

rng.Paragraphs(1).Style = wdStyleNormal

numParas = rng.Paragraphs.Count

rng.Paragraphs(numParas).Style = wdStyleNormal

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindStop

.Font.Color = wdColorAutomatic

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

doStop = False

Do While rng.Find.Found = True And doStop = False

rng.Collapse wdCollapseEnd

ggo = True

Do While ggo = True

rng.InsertBefore Text:="<<0>>"

rng.Collapse wdCollapseEnd

rng.MoveEnd , 1

myCol = rng.Font.Color

If myCol = wdColorAutomatic Then Exit Do

rng.Collapse wdCollapseStart

rng.InsertBefore Text:="<<" & Trim(Str(myCol)) & ">>"

newCol = myCol

Do While newCol = myCol

rng.MoveEnd , 1

newCol = rng.Font.Color

If newCol <> myCol Then

If newCol = wdColorAutomatic Then

rng.Collapse wdCollapseStart

rng.InsertBefore Text:="<<0>>"

rng.Collapse wdCollapseEnd

rng.Select

ggo = False

End If

If newCol <> myCol Then

rng.Collapse wdCollapseStart

rng.InsertBefore Text:="<<" & Trim(Str(newCol)) & ">>"

rng.Collapse wdCollapseEnd

rng.Collapse wdCollapseStart

myCol = newCol

End If

End If

rng.Collapse wdCollapseEnd

DoEvents

If ActiveDocument.Content.End - rng.End < 10 Then

ggo = False

doStop = True

newCol = 99999

End If

Loop

DoEvents

Loop

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Beep

Selection.HomeKey Unit:=wdStory

' Reset to style Normal and no applied font attributes

Set rng = ActiveDocument.Content

rng.Style = ActiveDocument.Styles(wdStyleNormal)

rng.Font.Reset

Beep

' Add back italic and bold etc

' Italic first

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = "\[\[\*\]\]"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

If doBoldItalAsStyle = True Then

rng.Style = myItalStyle

Else

rng.Font.Italic = True

End If

startNow = rng.start

rng.start = rng.End - 2

endNow = rng.End

rng.Delete

rng.start = startNow

rng.End = rng.start + 2

rng.Delete

' Go and find the next occurrence (if there is one)

rng.Find.Execute

DoEvents

Loop

' Now bold

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = "\{\{\*\}\}"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

If doBoldItalAsStyle = True Then

rng.Style = myBoldStyle

Else

rng.Font.Bold = True

End If

startNow = rng.start

rng.start = rng.End - 2

endNow = rng.End

rng.Delete

rng.start = startNow

rng.End = rng.start + 2

rng.Delete

' Go and find the next occurrence (if there is one)

rng.Find.Execute

DoEvents

Loop

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = "\[\[\*\]\]"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

If doBoldItalAsStyle = True Then

rng.Style = myBoldStyle

Else

rng.Font.Bold = True

End If

startNow = rng.start

rng.start = rng.End - 2

endNow = rng.End

rng.Delete

rng.start = startNow

rng.End = rng.start + 2

rng.Delete

' Go and find the next occurrence (if there is one)

rng.Find.Execute

DoEvents

Loop

' Superscript

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = "\\\\\*\;\;"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rng.Font.Superscript = True

startNow = rng.start

rng.start = rng.End - 2

endNow = rng.End

rng.Delete

rng.start = startNow

rng.End = rng.start + 2

rng.Delete

' Go and find the next occurrence (if there is one)

rng.Find.Execute

DoEvents

Loop

' Subscript

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = "//\*\:\:"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rng.Font.Subscript = True

startNow = rng.start

rng.start = rng.End - 2

endNow = rng.End

rng.Delete

rng.start = startNow

rng.End = rng.start + 2

rng.Delete

' Go and find the next occurrence (if there is one)

rng.Find.Execute

DoEvents

Loop

' Replace all the codes with the correct font colour

Set rng = ActiveDocument.Content

Set rng2 = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\<\<\*\<\<"

.Wrap = wdFindStop

.Replacement.Text = "<<"

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True

rng.Select

If myCount Mod 20 = 0 Then rng.Select

anglePos = InStr(rng.Text, ">")

colNumText = Mid(rng.Text, 3, anglePos - 3)

colNum = Val(colNumText)

rng.Font.Color = colNum

rng2.start = rng.start

rng2.End = rng.start + anglePos + 1

rng2.Select

rng2.Delete

rng.Select

rng.Collapse wdCollapseStart

rng.Find.Execute

DoEvents

Loop

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<<Dummy"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = "Dummy"

.Execute Replace:=wdReplaceAll

DoEvents

End With

Beep

Selection.HomeKey Unit:=wdStory

End Sub

**Sub MultiSwitch()**

**'** Paul Beverley - Version 11.06.24

' Scripted word/phrase switching

maxWords = 5

myListName = "zzSwitchList.docx"

Set rng = ActiveDocument.Content

commentNearCheck = True

' Set min number of chars for an abbreviation

minChars = 3

includeApostrophe = True

linksOFF = True

addMarkerInList = True

notTheseChars = ",!?.)>] "

' If you want to load the SwitchList file automatically...

' then...

' On a Mac, you will need something like this:

myList = "/Users/Paul/My Documents/Macro stuff/zzSwitchList.docx"

' On Windows, you will need something like this:

myList = "C:\Documents and Settings\Paul\My Documents\zzSwitchList.docx"

CR = vbCr

CR2 = CR & CR

Dim myStr(20) As String

Dim endStr(20) As Long

On Error GoTo ReportIt

Set theDoc = ActiveDocument

' Just in case they are using "smart" cut/paste option

mySmartOpt = Options.SmartCutPaste

Options.SmartCutPaste = False

nowSearch = Selection.Find.Text

' Read the context, noting if the word is selected

wasSelected = (Selection.End > Selection.Start)

If wasSelected = True Then

maxWords = 1

Else

myTest = Selection.Text

Debug.Print "|" & Selection.Text & "|"

Debug.Print "!" & myTest & "!"

Debug.Print Asc(Selection.Text)

If Asc(myTest) = 13 Then Selection.MoveLeft , 1

If InStr(notTheseChars, myTest) > 0 Then Selection.MoveLeft , 1

Selection.Expand wdWord

End If

' If no word selected

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Debug.Print Asc(rng)

okChars = ".,;:" & Chr(11) & vbCr & ChrW(8211) & ChrW(8212) & ChrW(8221)

If InStr(okChars, Left(Selection, 1)) > 0 Then

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.Expand wdWord

End If

isAbbrev = Not (Len(Selection) > minChars)

startNow = Selection.Start

Set startRng = Selection.Range.Duplicate

iMax = maxWords

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.MoveEnd Unit:=wdWord, Count:=maxWords

numTCs = rng.Revisions.Count

If numTCs > 0 Then

rng.Revisions.AcceptAll

rng.MoveEnd Unit:=wdWord, Count:=maxWords

End If

wdsLeft = rng.Words.Count

If maxWords > wdsLeft Then maxWords = wdsLeft

If Asc(rng.Words(wdsLeft)) = 13 Then maxWords = maxWords - 1

For i = 1 To maxWords

endStr(i) = rng.Words(i).End

If rng.Words(i) = vbCr Then

maxWords = i - 1

Exit For

End If

Debug.Print "|" & rng.Text & "|"

If Right(rng.Words(i), 1) = " " Then endStr(i) = endStr(i) - 1

myStr(i) = Trim(Left(rng, endStr(i) - rng.Start))

DoEvents

Next i

If numTCs > 0 Then WordBasic.EditUndo

Selection.Collapse wdCollapseStart

iMax = maxWords

gottaList = False

For Each myDoc In Application.Documents

thisName = myDoc.Name

If thisName = myListName Then

Set theList = Documents(myListName)

gottaList = True

Exit For

End If

Next myDoc

' Find the zzSwitchList file

If gottaList = False Then

Documents.Open fileName:=myList

Set theList = ActiveDocument

theDoc.Activate

End If

Set rng = theList.Content

' Check if list has LFs

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Text = ChrW(11)

.Replacement.Text = ""

.Execute

DoEvents

If .Found = True Then

Beep

rng.Start = rng.Start - 1

rng.Collapse wdCollapseStart

rng.Expand wdWord

rng.HighlightColorIndex = wdYellow

rng.Collapse wdCollapseEnd

rng.Select

MsgBox "The switch list must use paragraphs, not line breaks!"

ActiveWindow.ActivePane.View.ShowAll = True

rng.Find.Text = nowSearch

Exit Sub

End If

End With

' Just in case the switch list has any URLs

If linksOFF = True Then

rng.Fields.Unlink

End If

lastTwo = Right(rng, 2)

If lastTwo <> CR2 Then rng.InsertAfter Text:=CR2

rng.Start = 0

rng.End = theList.Content.End

' Try to locate the selected text in the switch list

allWords = rng.Text

lfPos = InStr(allWords, ChrW(11))

If lfPos > 0 Then

rng.Start = lfPos - 2

rng.End = lfPos + 1

rng.Select

Beep

Application.Run "ShowFormatting"

MsgBox "The switch list must use paragraphs, not line breaks!"

Options.SmartCutPaste = mySmartOpt

Selection.Find.Text = nowSearch

Exit Sub

End If

For i = iMax To 1 Step -1

myPos = InStr(allWords, CR2 & Replace(myStr(i), CR, "") & CR)

If myPos > 0 Then

allWords = Mid(allWords, myPos + 1)

gottaMatch = True

numWords = i

'Display the options

numAlts = 1

endPos = InStr(allWords, CR2)

allWords = Left(allWords, endPos + 2)

altText = Split(allWords, Chr(13))

j = 0

Do

j = j + 1

Loop Until Len(altText(j)) = 0 Or j = 20

numAlts = j - 2

' We now have j = number of options

' altText contains all numAlts texts

Exit For

Else

gottaMatch = False

End If

Next i

' If it's not found at all, give up

If gottaMatch = False Then

myWd = myStr(1)

Do

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p^p" & myWd

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = True

.MatchWildcards = False

.Execute

.MatchCase = False

End With

If Len(myWd) > 1 Then myWd = Left(myWd, Len(myWd) - 1)

Loop Until rng.Find.Found = True Or Len(myWd) = 1

rng.Collapse wdCollapseStart

rng.Select

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Options.SmartCutPaste = mySmartOpt

Selection.Find.Text = nowSearch

myPrompt = "Can't find """ & myStr(1) & """"

myResponse = MsgBox(myPrompt, vbOKOnly, "MultiSwitch")

theDoc.Activate

Exit Sub

End If

'Offer j optional texts to the user

If numAlts > 1 And wasSelected = False Then

For i = 1 To numAlts

myPrompt = myPrompt & i & ": " & altText(i + 1) & CR

Next

Beep

myResponse = InputBox(myPrompt, "MultiSwitch")

myChoice = Val(myResponse)

If myChoice > numAlts Or myChoice = 0 Then

Options.SmartCutPaste = mySmartOpt

Selection.Find.Text = nowSearch

Exit Sub

End If

Else

myChoice = 1

End If

rng.Start = myPos + 2

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

For i = 1 To myChoice

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

Next i

' Copy chosen line (check first for format change symbol)

useFormat = False

rng.MoveEnd , -1

myNewItem = rng

Dim gotFormat As Boolean

gotFormat = rng.Font.Bold Or rng.Font.Italic Or rng.Font.SmallCaps \_

Or rng.Font.Underline Or rng.Font.Superscript Or rng.Font.Subscript \_

Or rng.InlineShapes.Count > 0

If AscW(rng) = 172 Or gotFormat Then

useFormat = True

If AscW(rng) = 172 Then rng.MoveStart , 1

rng.Copy

End If

noTrack = (rng.Font.StrikeThrough)

startRng.Select

Selection.End = endStr(numWords)

' Check if too near to a comment

If Selection.Information(wdInCommentPane) = 0 And \_

Selection.Information(wdInFootnote) = 0 And \_

Selection.Information(wdInEndnote) = 0 Then

numCmts = ActiveDocument.Comments.Count

If numCmts > 0 And commentNearCheck = True Then

' Find scope of (range covered by) the next comment

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Start = 0

cmtNumber = rng.Comments.Count + 1

If cmtNumber <= numCmts Then

Set nextCmtRange = ActiveDocument.Comments(cmtNumber).Scope

' Does the scope of that comment overlap the current word?

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

If rng.End > nextCmtRange.Start Then

Beep

rng.Select

myResponse = MsgBox("Selection contains a comment. Word might crash!" \_

& CR2 & "Please use a manual change.", \_

vbOKOnly, "MultiSwitch")

Options.SmartCutPaste = mySmartOpt

Selection.Find.Text = nowSearch

Exit Sub

End If

End If

End If

End If

myTrack = ActiveDocument.TrackRevisions

myState = ActiveWindow.View.ShowInsertionsAndDeletions

If noTrack Then ActiveDocument.TrackRevisions = False

If useFormat = True Then

Selection.Delete

Selection.Paste

Else

Selection.Text = myNewItem

End If

Selection.Start = startNow

textEnd = Selection.End

If InStr(Selection, "^p") > 0 Then

numCRs = 0

oldFind = Selection.Find.Text

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^^p"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute

End With

Do While Selection.Find.Found = True And Selection.Start < textEnd

Selection.TypeText Text:=vbCr

Selection.Find.Execute

numCRs = numCRs + 1

Loop

Selection.Start = startNow

Selection.End = textEnd - numCRs

Selection.Find.Text = oldFind

End If

If InStr(Selection, "^t") > 0 Then

numTabs = 0

oldFind = Selection.Find.Text

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^^t"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = False

.Execute

End With

Do While Selection.Find.Found = True And Selection.Start < textEnd

Selection.TypeText Text:=vbTab

Selection.Find.Execute

numTabs = numTabs + 1

Loop

Selection.Start = startNow

Selection.End = textEnd - numCRs - numTabs

Selection.Find.Text = oldFind

End If

If Asc(Selection) = Asc("!") Then

myLen = Len(Selection)

Selection.End = Selection.Start + 1

Selection.Start = Selection.Start - 1

Selection.Delete

Selection.MoveEnd , myLen - 1

End If

tildePos = InStr(Selection, "~")

Selection.Font.StrikeThrough = False

If tildePos = 0 Then

If isAbbrev = True Then

Selection.Collapse wdCollapseEnd

Else

Selection.Collapse wdCollapseStart

End If

Else

Do While InStr(Selection, "~") > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Selection.MoveEnd 1

Selection.Delete

End If

If addOriginal = True Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=" (" & myStr(1) & ")"

End If

If noTrack Then

ActiveDocument.TrackRevisions = myTrack

ActiveWindow.View.ShowInsertionsAndDeletions = myState

End If

Options.SmartCutPaste = mySmartOpt

Selection.Find.Text = "^p^p" & myStr(1)

Exit Sub

ReportIt:

myErr = Err.Number

If myErr = 4160 Then

DoEvents

gottaList = False

Resume Next

Else

If myErr = 5174 Then

Beep

myPrompt = "Please open your switch list file: " & myListName \_

& CR2 & CR & "I did try looking for file: " & CR2 & myList

myResponse = MsgBox(myPrompt, vbQuestion + vbOKOnly, "MultiSwitch")

Exit Sub

End If

DoEvents

Resume

End If

End Sub

**Sub MacroFetchPlus()**

**'** Paul Beverley - Version 17.06.24

' Fetches the macro named in the clipboard or at the cursor, and it loads the URL into the clipboard. If on a blank line, it uses the clipboard text as the name

Selection.Collapse wdCollapseEnd

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

If rng.Text = vbCr Then

rng.Collapse wdCollapseStart

rng.Select

Selection.Paste

Selection.start = rng.start

myMacroName = Trim(Selection)

If InStr(myMacroName, " ") > 0 Or Len(myMacroName) > 20 Then

myResponse = MsgBox("Fetch this macro?", vbQuestion \_

+ vbYesNoCancel, "MacroFetchPlus")

If myResponse <> vbYes Then Exit Sub

End If

Else

Selection.MoveLeft , 2

Selection.Expand wdWord

myMacroName = Trim(Selection)

End If

Selection.Collapse wdCollapseEnd

myFolder = Left(myMacroName, 1)

myName = ""

For i = 1 To Len(myMacroName)

ch = Mid(myMacroName, i, 1)

If AscW(ch) > 128 Then

myName = myName & "%" & Hex(AscW(ch))

myFolder = "ES"

Else

myName = myName & ch

End If

Next i

myFullName = "https://www.wordmacrotools.com/macros/" & myFolder & "/" & myName

Selection.InsertAfter Text:=myFullName

Selection.Range.Copy

WordBasic.EditUndo

If Len(Selection) < 3 Then

Selection.MoveLeft , 3

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

myMacroName = Selection

thisIsMacro = True

ch1 = Left(myMacroName, 1)

If LCase(ch1) = ch1 Then thisIsMacro = False

For i = 1 To Len(myMacroName) - 3

ch = Mid(myMacroName, i, 1)

If LCase(ch) = UCase(ch) Then

thisIsMacro = False

Exit For

End If

DoEvents

Next i

GotMacro = True

ch1 = Left(myMacroName, 1)

If LCase(ch1) = ch1 Then GotMacro = False

For i = 1 To Len(myMacroName) - 3

ch = Mid(myMacroName, i, 1)

If LCase(ch) = UCase(ch) Then

GotMacro = False

Exit For

End If

DoEvents

Next i

If Len(myMacroName) > 3 And GotMacro = True Then

myFolder = Left(myMacroName, 1)

myFullName = "https://www.wordmacrotools.com/macros/" & \_

myFolder & "/" & myMacroName

ActiveDocument.FollowHyperlink Address:=myFullName

End If

If GotMacro = True Then

ShowVisualBasicEditor = True

Else

Beep

MsgBox myMacroName & "?? - Are you sure this is a macro name?"

End If

Selection.Collapse wdCollapseStart

End Sub

**Sub BoldQuickSwitch()**

**'** Paul Beverley - Version 01.07.24

' Adds/removes bold to/from selected text

If Selection.start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.start = startNow

End If

Set firstChar = Selection.Range.Characters(1)

Selection.Font.Bold = Not (firstChar.Font.Bold)

Selection.Collapse wdCollapseStart

End Sub

**Sub ItalicQuickSwitch()**

**'** Paul Beverley - Version 01.07.24

' Adds/removes italic to/from selected text

If Selection.start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.start = startNow

End If

Set firstChar = Selection.Range.Characters(1)

Selection.Font.Italic = Not (firstChar.Font.Italic)

Selection.Collapse wdCollapseStart

End Sub

**Sub MultiFileBackgroundColourChange()**

**'** Paul Beverley - Version 03.07.24

' Changes the background colour of all files in a list

' pale blue

' myColour = RGB(202, 237, 251)

' yellow

myColour = RGB(255, 255, 200)

' back to white

' myColour = RGB(255, 255, 255)

Dim allMyFiles2(200) As String

Set rng = ActiveDocument.Content

myExtent = 250

If rng.End - rng.start > myExtent Then rng.End = rng.start + myExtent

If InStr(LCase(rng.Text), ".doc") = 0 And InStr(LCase(rng.Text), ".rtf") = 0 Then

' If not a file list then open a file in the relevant folder

myResponse = MsgBox("Navigate to the required folder; then press 'Cancel'" \_

, , "MultifilePDF")

docCount = Documents.Count

Dialogs(wdDialogFileOpen).Show

If Documents.Count > docCount Then ActiveDocument.Close

dirPath = CurDir()

ChDir dirPath

' Read the names of all the files in this directory

myFile = Dir(CurDir() & Application.PathSeparator)

Documents.Add

numFiles = 0

Do While myFile <> ""

If InStr(LCase(myFile), ".doc") > 0 Or InStr(LCase(myFile), ".rtf") > 0 Then

Selection.TypeText myFile & vbCr

numFiles = numFiles + 1

End If

myFile = Dir()

Loop

' Now sort the file list (only actually needed for Macs)

Selection.WholeStory

Selection.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.HomeKey Unit:=wdStory

Selection.TypeText dirPath

' Go back until you hit myDelimiter

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

dirName = Selection

Selection.HomeKey Unit:=wdStory

If pdfFilesInFolder = True Then

mySubFolder = "\" & pdfFolderName

Else

mySubFolder = ""

End If

myResponse = MsgBox("Change the screen colour and resave ALL the files in this folder: " & mySubFolder & dirName \_

& " ?", vbQuestion + vbYesNoCancel, "MultiFile Background Colour Change")

If myResponse <> vbYes Then Exit Sub

End If

' Pick up the folder name and the filenames from the file list

numFiles = 0

myFolder = ""

For Each myPara In ActiveDocument.Paragraphs

myPara.Range.Select

Selection.MoveEnd , -1

lineText = Selection

If myFolder = "" Then

myFolder = lineText

Selection.Collapse wdCollapseEnd

Selection.MoveStartUntil cset:=":\", Count:=wdBackward

Selection.MoveStart , -1

myDelimiter = Left(Selection, 1)

Else

thisFile = lineText

If Len(thisFile) > 2 Then

If Left(thisFile, 1) <> "|" Then

numFiles = numFiles + 1

allMyFiles2(numFiles) = thisFile

End If

End If

End If

Next myPara

If pdfFilesInFolder = True Then

myResponse = MsgBox("Change the screen colour and resave ALL the files listed here, in this folder: " & \_

mySubFolder & myFolder & " ?", \_

vbQuestion + vbYesNoCancel, "Multifile Background Colour Change")

If myResponse <> vbYes Then Exit Sub

End If

For i = 1 To numFiles

thisName = allMyFiles2(i)

myFileType = Mid(thisName, InStr(thisName, "."))

justName = Replace(thisName, myFileType, "")

thisFile = myFolder & myDelimiter & thisName

Set myDoc = Application.Documents.Open(fileName:=thisFile)

StatusBar = thisName

With ActiveDocument.Background.Fill

.ForeColor.RGB = myColour

.Visible = msoTrue

.Solid

End With

myDoc.Save

myDoc.Close

Next

Selection.HomeKey Unit:=wdStory

MsgBox (numFiles & " files saved with new screen colour")

End Sub

**Sub JumpPage()**

**'** Paul Beverley - Version 31.07.24

' Jumps to the head of the next page

myExtra = 4

Selection.GoTo What:=wdGoToPage, Which:=wdGoToPrevious

Set rng = Selection.Range.Duplicate

Selection.GoTo What:=wdGoToPage, Which:=wdGoToNext

rng.Select

ActiveDocument.ActiveWindow.SmallScroll up:=myExtra

End Sub

**Sub JumpPageUp()**

**'** Paul Beverley - Version 31.07.24

' Jumps to the head of the next page

myExtra = 4

Selection.GoTo What:=wdGoToPage, Which:=wdGoToNext

Set rng = Selection.Range.Duplicate

Selection.GoTo What:=wdGoToPage, Which:=wdGoToNext

rng.Select

ActiveDocument.ActiveWindow.SmallScroll up:=myExtra

End Sub

**Sub MidSentenceCapitalRemove()**

**'** Paul Beverley - Version 05.07.24

' Finds initial caps within sentences and offers to lowercase

Set rng = Selection.Range.Duplicate

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[A-Z]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

notThese = ";I ;I’m ;I’ve "

Do While rng.Find.Found = True

Set rng2 = rng.Duplicate

rng2.Expand wdSentence

If rng2.start <> rng.start Then

rng.Expand wdWord

If InStr(notThese, rng) = 0 Then

rng.Select

myResponse = MsgBox("Lowercase?", \_

vbQuestion + vbYesNoCancel, "Mid Sentence Capital Remove")

If myResponse = vbCancel Then

Selection.Collapse wdCollapseEnd

Exit Sub

End If

If myResponse = vbYes Then

rng.End = rng.start + 1

rng.Text = LCase(rng.Text)

End If

End If

rng.Collapse wdCollapseEnd

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Beep

End Sub

**Sub TextPickup()**

**'** Paul Beverley - Version 06.07.24

' Collects text from a series of text areas

maxWords = 1000

beepTime = 3

Set rng = Selection.Range.Duplicate

CR = vbCr

nameNow = ActiveDocument.Name

Set myDoc = ActiveDocument

Do

DoEvents

Loop Until ActiveDocument.Name <> nameNow

Do

Selection.WholeStory

If Selection.Words.Count > maxWords Then

myResponse = MsgBox("Add this large area of text?", \_

vbQuestion + vbYesNo, "Text Hoover")

If myResponse <> vbYes Then Exit Sub

End If

Selection.Copy

rng.Paste

If Right(rng, 2) = " " & CR Or Right(rng, 2) = "-" & CR Then

rng.start = rng.End - 1

rng.Delete

End If

rng.Collapse wdCollapseEnd

myTime = Timer

Do

DoEvents

If Timer - myTime > beepTime Then

Beep

myTime = Timer

StatusBar = "Press up-arrow key to stop"

End If

Loop Until Selection.start = Selection.End

DoEvents

Loop Until Selection.End = 0

MsgBox "Finished by user"

End Sub

**Sub TextHoover()**

**'** Paul Beverley - Version 13.07.24

' Copies formatted text from all of the text areas

' mainTextFirst = True

mainTextFirst = False

tidyFormatting = True

timeNow = Timer

Set sourceDoc = ActiveDocument

Documents.Add

'Iterate through all story types in the current document

For Each rngStory In sourceDoc.StoryRanges

'Iterate through all linked stories

Do

If rngStory.StoryType <> wdMainTextStory Then

rngStory.Copy

Selection.Paste

End If

'Get next linked story (if any)

Set rngStory = rngStory.NextStoryRange

Loop Until rngStory Is Nothing

DoEvents

Next

' Add main story, but it brings text frames too

Set rng = ActiveDocument.Content

If mainTextFirst = True Then

rng.Collapse wdCollapseStart

Else

rng.Collapse wdCollapseEnd

End If

rng.FormattedText = sourceDoc.Content.FormattedText

' Delete the text frames

numShapes = ActiveDocument.Shapes.Count

For i = numShapes To 1 Step -1

ActiveDocument.Shapes(i).Delete

Next i

' Improve the formatting

Set rng = ActiveDocument.Content

If tidyFormatting = True Then

rng.ParagraphFormat.Alignment = wdAlignParagraphLeft

rng.PageSetup.TextColumns.SetCount NumColumns:=1

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = " ^p"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = " "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

.Text = "-^p"

.Replacement.Text = "-"

.Execute Replace:=wdReplaceAll

' Turn white text black

.Text = ""

.Replacement.Text = "^&"

.Font.Color = wdColorWhite

.Replacement.Font.Color = wdColorBlack

.Execute Replace:=wdReplaceAll

End With

End If

MsgBox "Finished in " & Str(Int(Timer - timeNow)) & " secs"

End Sub

**Sub HighlightingRemoveAll()**

**'** Paul Beverley - Version 12.07.24

' Remove all highlighting, including in notes, boxes & comments

' Main text

ActiveDocument.Content.HighlightColorIndex = wdNoHighlight

' Notes

If ActiveDocument.Footnotes.Count > 0 Then \_

ActiveDocument.StoryRanges(wdFootnotesStory).HighlightColorIndex = wdNoHighlight

If ActiveDocument.Endnotes.Count > 0 Then \_

ActiveDocument.StoryRanges(wdEndnotesStory).HighlightColorIndex = wdNoHighlight

' Comments

If ActiveDocument.Comments.Count > 0 Then

For Each cmt In ActiveDocument.Comments

cmt.Range.HighlightColorIndex = wdNoHighlight

DoEvents

Next cmt

End If

' Text boxes

If ActiveDocument.Shapes.Count > 0 Then

For Each Shp In ActiveDocument.Shapes

If Shp.Type <> 24 And Shp.Type <> 3 Then

If Shp.TextFrame.HasText Then

Shp.TextFrame.TextRange.HighlightColorIndex = 0

End If

End If

Next

DoEvents

End If

Beep

End Sub

**Sub FindSpecialCharacters()**

**'** Paul Beverley - Version 08.08.24

' Finds next group of special characters, e.g. Chinese/Arabic

myRange = "100-7FFF"

' decimal characters to ignore

notRangeFrom = 8211 ' en dash

'notRangeTo = 8221 ' close double quote

'notRangeTo = 8226 ' bullet

notRangeTo = 8230 ' ellipsis

myStart = Val("&H" & Left(myRange, 4))

myEnd = Val("&H" & Right(myRange, 4))

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = "[" & ChrW(myStart) & "-" & ChrW(myEnd) & "]{1,}"

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

gotSome = False

Do

.Execute

' Test if at least one actual 'special'

myText = Selection

For i = 1 To Len(myText)

tst = AscW(Mid(myText, i, 1))

If tst < notRangeFrom Or tst > notRangeTo Then

gotSome = True

Exit For

End If

Next i

Loop Until gotSome = True

End With

End Sub

**Sub KerningAdjustShortest()**

**'** Paul Beverley - Version 24.07.24

' Kerns selected text or selects the shortest line in the paragraph

inc = 0.05

If Selection.start <> Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

If rng.start = Selection.start And \_

rng.End = Selection.End Then

Selection.Range.Font.Spacing = 0

End If

Selection.Range.Font.Spacing \_

= Selection.Range.Font.Spacing + inc

Exit Sub

End If

Dim lineLen(100)

Dim lineStart(100) As Long

Selection.HomeKey Unit:=wdLine

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

paraEnd = rng.End

rng.Collapse wdCollapseStart

rng.Select

i = 0

' Locate the start of each line of text

Do Until Selection.End > paraEnd

i = i + 1

lineStart(i) = Selection.start

Selection.MoveDown Unit:=wdLine, Count:=1

DoEvents

Loop

iMax = i - 1

' iMax = number of lines in the para

minLen = 1000

For i = 1 To iMax - 1

' i.e. ignore the final line

Selection.End = lineStart(i + 1) - 1

Selection.Collapse wdCollapseEnd

lineLen(i) = Selection.Information(wdHorizontalPositionRelativeToPage)

If lineLen(i) < minLen Then

shortLine = i

minLen = lineLen(i)

End If

DoEvents

Next i

Selection.start = lineStart(shortLine)

Selection.End = lineStart(shortLine + 1)

End Sub

**Sub GoogleFetchSpecificBrowser()**

**'** Version 24.07.24

' Launches selected text to Google using a specified browser

runBrowser = "C:\Program Files\Mozilla Firefox\Firefox"

' runBrowser = "C:\Program Files\Google\Chrome\Application\chrome"

' runBrowser = "C:\Program Files\Internet Explorer\iexplore"

mySite = "http://www.google.co.uk/search?q="

If Selection.start = Selection.End Then

Selection.Expand wdWord

If Len(Selection) < 3 Then

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, " ", "+")

Shell (runBrowser & " " & mySite & mySubject)

End Sub

**Sub DocAlyseMedical()**

**'** Paul Beverley - Version 03.08.24

' Analyses the use of various abbreviations in a medical document

' prompts to count number of tests

cc = 53

myScreenOff = True

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

If doingSeveralMacros = False Then

myResponse = MsgBox(" DocAlyse Medical" & vbCr & vbCr & \_

"Analyse this document?", vbQuestion \_

+ vbYesNoCancel, "DocAlyse Medical")

If myResponse <> vbYes Then Exit Sub

End If

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

ss = "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" & \_

"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rngOld = ActiveDocument.Content

Documents.Add

Set tempDoc = ActiveDocument

Set rng = ActiveDocument.Content

rng.FormattedText = rngOld.FormattedText

Selection.EndKey Unit:=wdStory

If ActiveDocument.Footnotes.Count > 0 Then

ActiveDocument.StoryRanges(wdFootnotesStory).Copy

Selection.Paste

Selection.Collapse wdCollapseEnd

End If

If ActiveDocument.Endnotes.Count > 0 Then

ActiveDocument.StoryRanges(wdEndnotesStory).Copy

Selection.Paste

Selection.Collapse wdCollapseEnd

End If

Set endTempfile = ActiveDocument.Content

endTempfile.Collapse wdCollapseEnd

' collect text in all the textboxes (if any)

sh = ActiveDocument.Shapes.Count

If sh > 0 Then

For Each shp In ActiveDocument.Shapes

If shp.Type <> 24 And shp.Type <> 3 Then

If shp.TextFrame.HasText Then

shp.TextFrame.TextRange.Copy

endTempfile.Select

Selection.Paste

End If

End If

Next

End If

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.Revisions.AcceptAll

DoEvents

myTot = ActiveDocument.Range.End

Selection.HomeKey Unit:=wdStory

Set rngOld = ActiveDocument.Content

ActiveDocument.TrackRevisions = myTrack

Documents.Add

Set rng = ActiveDocument.Content

rng.Text = rngOld.Text

myRslt = ""

myTot = ActiveDocument.Range.End

CR = vbCr: CR2 = CR & CR

tr = Chr(9) & "0zczc" & CR: SP = " "

Selection.HomeKey Unit:=wdStory

Set newDoc = ActiveDocument

cc = cc - 1

DoEvents

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

.Replacement.Text = "^&!"

.MatchWildcards = True

DoEvents

End With

' bd, bds, bid b.i.d

With rng.Find

.Text = "[Bb][Dd]>"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "[Bb][Dd][Ss]>"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "[Bb][Ii][Dd]>"

.Execute Replace:=wdReplaceAll

End With

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "[Bb].[Ii].[Dd]>"

.Execute Replace:=wdReplaceAll

End With

l = ActiveDocument.Range.End - myTot

If l > 0 Then WordBasic.EditUnDo

If i + j + k + l > 0 Then myRslt = myRslt & "bd" & \_

vbTab & Trim(Str(i)) & CR & "bds" & vbTab & \_

Trim(Str(j)) & CR & "bid (?word or abbr.)" & \_

vbTab & Trim(Str(k)) & CR & "b.i.d" & vbTab \_

& Trim(Str(l)) & CR2

'tds, tid, t.i.d

With rng.Find

.Text = "[Tt][Dd][Ss]>"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "[Tt][Ii][Dd]>"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "[Tt].[Ii].[Dd]>"

.Execute Replace:=wdReplaceAll

End With

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUnDo

If i + j + k > 0 Then myRslt = myRslt & "tds" & vbTab \_

& Trim(Str(i)) & CR & "tid" & vbTab & Trim(Str(j)) \_

& CR & "t.i.d" & vbTab & Trim(Str(k)) & CR2

'qds, qid, q.i.d

With rng.Find

.Text = "[Qq][Dd][Ss]>"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "[Qq][Ii][Dd]>"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "[Qq].[Ii].[Dd]>"

.Execute Replace:=wdReplaceAll

End With

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUnDo

If i + j + k > 0 Then myRslt = myRslt & "qds" & vbTab \_

& Trim(Str(i)) & CR & "qid" & vbTab & Trim(Str(j)) \_

& CR & "q.i.d" & vbTab & Trim(Str(k)) & CR2

'#hrly

With rng.Find

.Text = "[0-9]hrly>"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

'#[ -]hrly

With rng.Find

.Text = "[0-9][ -]hrly>"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

'q#h

With rng.Find

.Text = "[Qq][0-9][Hh]>"

.Execute Replace:=wdReplaceAll

End With

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUnDo

'qqh

With rng.Find

.Text = "[Qq][Qq][Hh]>"

.Execute Replace:=wdReplaceAll

End With

l = ActiveDocument.Range.End - myTot

If l > 0 Then WordBasic.EditUnDo

If i + j + k + l > 0 Then myRslt = myRslt & "#hrly" \_

& vbTab & Trim(Str(i)) & CR & "# hrly" & vbTab \_

& Trim(Str(j)) & CR & "q#h" & vbTab & Trim(Str(k)) & CR \_

& "qqh" & vbTab & Trim(Str(l)) & CR2

'prn

With rng.Find

.Text = "[Pp][Rr][Nn]>"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

'p.r.n

With rng.Find

.Text = "[Pp].[Rr].[Nn]>"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

'sos

With rng.Find

.Text = "[Ss][Oo][Ss]>"

.Execute Replace:=wdReplaceAll

End With

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUnDo

's.o.s

With rng.Find

.Text = "[Ss].[Oo].[Ss]>"

.Execute Replace:=wdReplaceAll

End With

l = ActiveDocument.Range.End - myTot

If l > 0 Then WordBasic.EditUnDo

If i + j + k + l > 0 Then myRslt = myRslt & "prn" & vbTab \_

& Trim(Str(i)) & CR & "p.r.n" & vbTab & Trim(Str(j)) \_

& CR & "sos" & vbTab & Trim(Str(k)) & CR \_

& "s.o.s" & vbTab & Trim(Str(l)) & CR2

'IV / i.v.

With rng.Find

.Text = "<iv>"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "<i.v>"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "<IV>"

.Execute Replace:=wdReplaceAll

End With

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "<I.V>"

.Execute Replace:=wdReplaceAll

End With

l = ActiveDocument.Range.End - myTot

If l > 0 Then WordBasic.EditUnDo

If i + j + k + l > 0 Then myRslt = myRslt & "iv" & vbTab \_

& Trim(Str(i)) & CR & "i.v." & vbTab & Trim(Str(j)) \_

& CR & "IV" & vbTab & Trim(Str(k)) & CR \_

& "I.V." & vbTab & Trim(Str(l)) & CR2

'IM / i.m.

With rng.Find

.Text = "<im>"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "<i.m>"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "<IM>"

.Execute Replace:=wdReplaceAll

End With

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "<I.M>"

.Execute Replace:=wdReplaceAll

End With

l = ActiveDocument.Range.End - myTot

If l > 0 Then WordBasic.EditUnDo

If i + j + k + l > 0 Then myRslt = myRslt & "im" & vbTab \_

& Trim(Str(i)) & CR & "i.m." & vbTab & Trim(Str(j)) \_

& CR & "IM" & vbTab & Trim(Str(k)) & CR \_

& "I.M." & vbTab & Trim(Str(l)) & CR2

'SC / s.c.

With rng.Find

.Text = "<sc>"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "<s.c>"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "<SC>"

.Execute Replace:=wdReplaceAll

End With

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "<S.C>"

.Execute Replace:=wdReplaceAll

End With

l = ActiveDocument.Range.End - myTot

If l > 0 Then WordBasic.EditUnDo

If i + j + k + l > 0 Then myRslt = myRslt & "sc" & vbTab \_

& Trim(Str(i)) & CR & "s.c." & vbTab & Trim(Str(j)) \_

& CR & "SC" & vbTab & Trim(Str(k)) & CR \_

& "S.C." & vbTab & Trim(Str(l)) & CR2

'# ?

With rng.Find

.Text = "[0-9]" & Chr(181)

.Execute Replace:=wdReplaceAll

End With

h = ActiveDocument.Range.End - myTot

If h > 0 Then WordBasic.EditUnDo

'# ?

With rng.Find

.Text = "[0-9] " & Chr(181)

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

'#micro

With rng.Find

.Text = "[0-9]micro"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

'# micro

With rng.Find

.Text = "[0-9] micro"

.Execute Replace:=wdReplaceAll

End With

k = ActiveDocument.Range.End - myTot

If k > 0 Then WordBasic.EditUnDo

If h + i + j + k > 0 Then myRslt = myRslt & "# " \_

& Chr(181) & vbTab & Trim(Str(h + i)) & CR \_

& "# " & "micro" & vbTab & Trim(Str(j + k)) & CR2

'count/minute

With rng.Find

.Text = "cpm>"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "c.p.m>"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

If i + j > 0 Then myRslt = myRslt & "cpm" & vbTab & Trim(Str(i)) \_

& CR & "c.p.m." & vbTab & Trim(Str(j)) & CR2

'beats/minute

With rng.Find

.Text = "bpm>"

.Execute Replace:=wdReplaceAll

End With

i = ActiveDocument.Range.End - myTot

If i > 0 Then WordBasic.EditUnDo

With rng.Find

.Text = "b.p.m>"

.Execute Replace:=wdReplaceAll

End With

j = ActiveDocument.Range.End - myTot

If j > 0 Then WordBasic.EditUnDo

If i + j > 0 Then myRslt = myRslt & "bpm" & vbTab & Trim(Str(i)) & CR \_

& "b.p.m." & vbTab & Trim(Str(j)) & CR2

myRslt = myRslt & CR

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="DocAlyse Medical" & vbCr & vbCr

newDoc.Paragraphs(1).Style = ActiveDocument.Styles(wdStyleHeading1)

newDoc.Paragraphs(3).Range.Select

Selection.End = newDoc.Content.End

Selection.TypeText CR & myRslt & CR2

Selection.Font.Bold = True

Set rng = ActiveDocument.Content

rng.ParagraphFormat.TabStops.ClearAll

rng.ParagraphFormat.TabStops.Add Position:=CentimetersToPoints(4.5), \_

Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces

' Grey out the zero lines

cc = cc - 1

DoEvents

Set rng = ActiveDocument.Content

StatusBar = ss & " " & Trim(Str(cc)) & vbCr

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^13([!^13]@)^t0"

.Wrap = wdFindContinue

.Replacement.Text = "^p\1^t^="

.Replacement.Font.Bold = False

.Replacement.Font.Color = wdColorGray25

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

.Text = "^t^=zczc"

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

With Selection.Find

.Text = ""

.Replacement.Text = ""

.MatchWildcards = False

.Execute

End With

Selection.HomeKey Unit:=wdStory

tempDoc.Close SaveChanges:=False

If doingSeveralMacros = False Then

Beep

newDoc.Activate

Else

FUT.Activate

End If

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub TitlesRevealBoxes()**

**'** Paul Beverley - Version 08.08.24

' Switches the next light grey font colour to full colour

myColorBlue = &HF0B000

myColorBlue = &HFF0000

preBlue = &HF6F6F6

preRed = &HF4F4F4

preBlack = &HF5F5F5

Set docNow = ActiveDocument

pbTitlesFile.Activate

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

rng.MoveStart wdParagraph, -2

stopNow = False

If rng.ShapeRange.Count = 0 Then Beep: Exit Sub

For Each myShp In rng.ShapeRange

If myShp.TextFrame.HasText Then

Set rngBox = myShp.TextFrame.TextRange

For Each wd In rngBox.Words

myColour = wd.Font.Color

If myColour = 9999999 Then

For Each ch In wd.Characters

myColour = ch.Font.Color

Select Case myColour

Case preBlue: wd.Font.Color = myColorBlue: stopNow = True

Case preRed: wd.Font.Color = wdColorRed: stopNow = True

Case preBlack: wd.Font.Color = wdColorBlack: stopNow = True

End Select

Next ch

Else

Select Case myColour

Case preBlue: wd.Font.Color = myColorBlue: stopNow = True

Case preRed: wd.Font.Color = wdColorRed: stopNow = True

Case preBlack: wd.Font.Color = wdColorBlack: stopNow = True

End Select

End If

Next wd

End If

If stopNow = True Then Exit For

Next myShp

ActiveDocument.TrackRevisions = myTrack

docNow.Activate

End Sub

**Sub FindTwoWordsSwap()**

**'** Paul Beverley - Version 05.08.24

' Finds two words in the opposite order within a paragraph

CR2 = vbCr & vbCr

Set rng = Selection.Range.Duplicate

Selection.Collapse wdCollapseEnd

With Selection.Find

mySearch = .Text

myPrompt = mySearch

wcPos = InStr(mySearch, "[!^13]@")

If wcPos > 0 Then

myFirst = Left(mySearch, wcPos - 1)

myLast = Mid(mySearch, wcPos + 7)

myNewSearch = myLast & "[!^13]@" & myFirst

.Text = myNewSearch

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

If Selection.End < Selection.start Then

' VBA's gone bonkers

rng.Select

Beep

myPrompt = "Can't find:" & CR2 & myNewSearch

myPrompt = myPrompt & CR2 & "Word's Find is probably tripping over URL links"

myResponse = MsgBox(myPrompt, vbOKOnly, "FindTwoWordsSwap")

Exit Sub

End If

If .Found = False Then

rng.Select

Beep

.Text = myNewSearch

.Wrap = wdFindStop

.Forward = False

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

If .Found = False Then

Beep

rng.Select

Exit Sub

End If

End If

Exit Sub

End If

Beep

End With

End Sub

**Sub FindTwoWords()**

**'** Paul Beverley - Version 02.10.24

' Finds two words within a paragraph

CR2 = vbCr & vbCr

If Selection.start = Selection.End And \_

Asc(Selection) = 13 Then \_

Selection.MoveStart wdWord, -2

If Selection.start = Selection.End Then

Selection.Expand wdWord

Selection.MoveEnd wdWord, 1

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.start = startNow

End If

' Build up the wildcard search

myFirst = Trim(Selection.Words.First)

myInit = Left(myFirst, 1)

myFirst = "[" & LCase(myInit) & UCase(myInit) \_

& "]" & Mid(myFirst, 2)

myLast = Trim(Selection.Words.Last)

myInit = Left(myLast, 1)

myLast = "[" & LCase(myInit) & UCase(myInit) \_

& "]" & Mid(myLast, 2)

mySearch = myFirst & "[!^13]@" & myLast

'Execute WC find

Selection.Collapse wdCollapseEnd

Set rng = Selection.Range.Duplicate

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindStop

.Text = mySearch

.Replacement.Text = myReplace

.MatchWildcards = True

.Forward = True

.Execute

' If it's not found and/or if VBA has screwed up...

If .Found = False Or (Selection.End = 0 \_

And Selection.start > 0) Then

wcPos = InStr(mySearch, "[!^13]@")

myPrompt = mySearch

If wcPos > 0 Then

myFirst = Left(mySearch, wcPos - 1)

myLast = Mid(mySearch, wcPos + 7)

myNewSearch = myLast & "[!^13]@" & myFirst

myFirst = Mid(myFirst, 2, 1) & Mid(myFirst, 5)

myLast = Mid(myLast, 2, 1) & Mid(myLast, 5)

myPrompt = "Can't find: " & myFirst & " FOLLOWED BY " & myLast

myPrompt = myPrompt & CR2 & "Try the reverse order?"

myResponse = MsgBox(myPrompt, \_

vbQuestion + vbYesNoCancel, "FindTwoWords")

If myResponse = vbYes Then

rng.Select

.Text = myNewSearch

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

If Selection.End < Selection.start Then

' VBA's gone bonkers

rng.Select

Beep

myPrompt = "Can't find:" & CR2 & myNewSearch

myPrompt = myPrompt & CR2 & "Word's Find is probably tripping over the internal links"

myResponse = MsgBox(myPrompt, vbOKOnly, "FindTwoWords")

Exit Sub

End If

If .Found = False Then

rng.Select

Beep

.Text = myNewSearch

.Wrap = wdFindStop

.Forward = False

.Replacement.Text = ""

.MatchWildcards = True

.Execute

DoEvents

If .Found = False Then

Beep

rng.Select

Exit Sub

End If

End If

Else

Beep

End If

Exit Sub

End If

Beep

myPrompt = "Can't find:" & CR2 & myPrompt

myResponse = MsgBox(myPrompt, vbOKOnly, "FindTwoWords")

rng.Select

End If

End With

End Sub

**Sub DuplicateSentencesFind()**

**'** Paul Beverley - Version 22.11.24

' Lists any sentences that are duplicated

minWords = 3

Set myDoc = ActiveDocument

Set tempDoc = Documents.Add

Set myResults = Documents.Add

Set res = myResults.Content

totalNumSents = myDoc.Sentences.Count

ReDim myText(totalNumSents) As String

allInits = ""

j = 0

For sn = 1 To totalNumSents

thisSentence = Replace(myDoc.Sentences(sn).Text, vbCr, "")

numWds = myDoc.Sentences(sn).Words.Count

If Not (numWds < minWords) And LCase(thisSentence) <> \_

UCase(thisSentence) Then

For k = 1 To 3

myInit = Left(thisSentence, 1)

If UCase(myInit) = LCase(myInit) Then

thisSentence = Mid(thisSentence, 2)

End If

myLast = Right(thisSentence, 1)

If UCase(myLast) = LCase(myLast) Then

thisSentence = Left(thisSentence, Len(thisSentence) - 1)

myInit = Left(thisSentence, 1)

End If

Next k

myInit = Left(thisSentence, 1)

If InStr(allInits, myInit) = 0 Then

allInits = allInits & myInit

End If

j = j + 1

myText(j) = thisSentence

End If

DoEvents

Next sn

numInits = Len(allInits)

For a = 1 To numInits

Set rng = tempDoc.Content

rng.Text = ""

fstLttr = Mid(allInits, a, 1)

For sn = 1 To totalNumSents

pText = myText(sn)

If myDoc.Sentences(sn).Words.Count > minWords - 1 Then

init = UCase(Left(pText, 1))

If init = fstLttr Then rng.InsertAfter Text:=pText & vbCr

If init = fstLttr Then Debug.Print pText

End If

DoEvents

Next sn

Set rng = tempDoc.Content

rng.Sort

tempDoc.Paragraphs(1).Range.Delete

rng.InsertAfter Text:=vbCr & "Rubbish" & vbCr

numParas = tempDoc.Paragraphs.Count

prevPara = tempDoc.Paragraphs(1).Range.Text

wasAmatch = False

numDupl = 1

For i = 2 To numParas

thisPara = tempDoc.Paragraphs(i).Range.Text

gottaMatch = (thisPara = prevPara)

If Not (gottaMatch) Then

If wasAmatch Then

res.InsertAfter Text:=Replace(prevPara, vbCr, "") \_

& " . . [" & Trim(Str(numDupl)) & "]" & vbCr

numDupl = 1

End If

Else

numDupl = numDupl + 1

End If

wasAmatch = gottaMatch

prevPara = thisPara

DoEvents

Next i

DoEvents

Next a

tempDoc.Close SaveChanges:=False

myResults.Activate

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Duplicates List" & vbCr & vbCr

End Sub

**Sub GoogleFetchIncognito()**

**'** Paul Beverley - Version 14.09.24

' Launches selected text to Google Chrome but incognito

mySite = "http://www.google.co.uk/search?q="

' Adjust the address above for your country’s Google URL

' Adjust the address below for your browser

' runBrowser = "C:\Program Files\Mozilla Firefox\Firefox"

' runBrowser = "C:\Program Files\Internet Explorer\iexplore"

runBrowser = "C:\Program Files\Google\Chrome\Application\Chrome"

If Selection.start = Selection.End Then

Selection.Expand wdWord

If Len(Selection) < 3 Then

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.start = startNow

End If

mySubject = Selection.Text

Shell (runBrowser & " " & mySite & mySubject & " --incognito")

End Sub

**Sub DuplicatePhrasesFind()**

**'** Paul Beverley - Version 10.08.24

' Lists any sentences that are duplicated

minWords = 5

maxWords = 12

doNotesToo = True

Set myDoc = ActiveDocument

Set copyDoc = Documents.Add

Set rngOld = myDoc.Content

Set rng = copyDoc.Content

rng.Text = rngOld.Text

rng.Collapse wdCollapseEnd

If myDoc.Footnotes.Count > 0 And doNotesToo = True Then

rng.InsertAfter Text:=myDoc.StoryRanges(wdFootnotesStory)

End If

If myDoc.Endnotes.Count > 0 And doNotesToo = True Then

rng.InsertAfter Text:=myDoc.StoryRanges(wdEndnotesStory)

End If

Set rng = copyDoc.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[;:," & ChrW(8211) & ChrW(8212) & "]"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ". "

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set tempDoc = Documents.Add

Set myResults = Documents.Add

Set res = myResults.Content

totalNumSents = copyDoc.Sentences.Count

ReDim myText(totalNumSents) As String

allInits = ""

j = 0

For sn = 1 To totalNumSents

thisSentence = Replace(copyDoc.Sentences(sn).Text, vbCr, "")

numWds = copyDoc.Sentences(sn).Words.Count

If Not (numWds < minWords) And Not (numWds > maxWords) \_

And Len(thisSentence) > 10 Then

For k = 1 To 3

myInit = Left(thisSentence, 1)

If UCase(myInit) = LCase(myInit) Then

thisSentence = Mid(thisSentence, 2)

End If

myLast = Right(thisSentence, 1)

If UCase(myLast) = LCase(myLast) Then

thisSentence = Left(thisSentence, Len(thisSentence) - 1)

myInit = Left(thisSentence, 1)

End If

Next k

myInit = Left(thisSentence, 1)

If InStr(allInits, myInit) = 0 Then

allInits = allInits & myInit

End If

j = j + 1

myText(j) = thisSentence

End If

DoEvents

Next sn

numInits = Len(allInits)

For a = 1 To numInits

Debug.Print "To go: " & Str(numInits - a)

Set rng = tempDoc.Content

rng.Text = ""

fstLttr = Mid(allInits, a, 1)

For sn = 1 To totalNumSents

pText = myText(sn)

If copyDoc.Sentences(sn).Words.Count > minWords - 1 Then

init = UCase(Left(pText, 1))

If init = fstLttr Then rng.InsertAfter Text:=pText & vbCr

' If init = fstLttr Then Debug.Print pText

End If

DoEvents

Next sn

Set rng = tempDoc.Content

rng.Sort

tempDoc.Paragraphs(1).Range.Delete

rng.InsertAfter Text:=vbCr & "Rubbish" & vbCr

numParas = tempDoc.Paragraphs.Count

prevPara = tempDoc.Paragraphs(1).Range.Text

wasAmatch = False

numDupl = 1

For i = 2 To numParas

thisPara = tempDoc.Paragraphs(i).Range.Text

gottaMatch = (thisPara = prevPara)

If Not (gottaMatch) Then

If wasAmatch Then

res.InsertAfter Text:=Replace(prevPara, vbCr, "") \_

& " . . [" & Trim(Str(numDupl)) & "]" & vbCr

numDupl = 1

End If

Else

numDupl = numDupl + 1

End If

wasAmatch = gottaMatch

prevPara = thisPara

Next i

DoEvents

Next a

tempDoc.Close SaveChanges:=False

copyDoc.Close SaveChanges:=False

myResults.Activate

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Duplicates List" & vbCr & vbCr

End Sub

**Sub CommentGoTo()**

**'** Paul Beverley - Version 19.08.24

' Moves to a specific numbered comment

maxNum = ActiveDocument.Comments.Count

myPrompt = "Go to comment number (max: " & \_

Trim(Str(maxNum)) & "):"

Do

myText = InputBox(myPrompt, "CommentGoTo")

myNumber = Val(myText)

If myNumber = 0 Then Beep: Exit Sub

If myNumber > maxNum Then Beep: myPrompt = "Maximum is <" & Trim(Str(maxNum)) & ">!"

Loop Until Not (myNumber > maxNum)

Selection.GoTo What:=wdGoToComment, Count:=myNumber

End Sub

**Sub CommentNameChangerWithReplies()**

**'** Paul Beverley - Version 19.08.24

' Changes the name attached to comments

newName = "Referee"

nameNow = Application.UserName

Application.UserName = newName

CR = vbCr

Set myDoc = ActiveDocument

Set commentsDoc = Documents.Add

Selection.TypeText Text:="===" & CR

numCmts = myDoc.Comments.Count

ReDim myStart(numCmts) As Long

ReDim myEnd(numCmts) As Long

ReDim cmtStart(numCmts) As Long

ReDim cmtEnd(numCmts) As Long

For i = 1 To numCmts

Set cmt = myDoc.Comments(i)

myStart(i) = cmt.Scope.start

myEnd(i) = cmt.Scope.End

cmt.Range.Copy

cmtStart(i) = Selection.start

Selection.Paste

cmtEnd(i) = Selection.start

Selection.TypeText Text:=vbCr & "===" & CR

Next i

myStart(0) = 0

myEnd(0) = 0

myDoc.Activate

ActiveDocument.DeleteAllComments

Set cmt = commentsDoc.Content

Set rngDoc = myDoc.Content

For i = 1 To numCmts

cmt.start = cmtStart(i)

cmt.End = cmtEnd(i)

cmt.Copy

rngDoc.start = myStart(i)

rngDoc.End = myEnd(i)

If myStart(i) = myStart(i - 1) Then

rngDoc.start = myEnd(i)

rngDoc.End = myEnd(i) + 1

Set newCmt = rngDoc.Comments.Add(Range:=rngDoc)

ActiveDocument.Comments(i).Edit

Selection.InsertAfter Text:="Reply:" & vbCr

Selection.Font.Bold = True

Selection.Collapse wdCollapseEnd

Else

Set newCmt = rngDoc.Comments.Add(Range:=rngDoc)

ActiveDocument.Comments(i).Edit

End If

Selection.Paste

DoEvents

Next i

commentsDoc.Close SaveChanges:=False

Application.UserName = nameNow

ActiveDocument.ActiveWindow.View.SplitSpecial = wdPaneNone

Beep

End Sub

**Sub PhraseAlyse()**

**'** Paul Beverley - Version 24.08.24

' Lists phrases that are duplicated

minWords = 4

maxWords = 8

minFreqToList = 3

doNotesToo = True

strttime = Timer

On Error GoTo ReportIt

Set myDoc = ActiveDocument

Set copyDoc = Documents.Add

Set rngOld = myDoc.Content

Set rng = copyDoc.Content

rng.Text = rngOld.Text

rng.Collapse wdCollapseEnd

If myDoc.Footnotes.Count > 0 And doNotesToo = True Then

rng.InsertAfter Text:=myDoc.StoryRanges(wdFootnotesStory)

End If

If myDoc.Endnotes.Count > 0 And doNotesToo = True Then

rng.InsertAfter Text:=myDoc.StoryRanges(wdEndnotesStory)

End If

Set rng = copyDoc.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[;:,^t" & ChrW(8211) & ChrW(8212) & "]"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ". "

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^13[0-9]{1,}[.\) ]{1,2}"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "[ ]{2,}"

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "N.B."

.Replacement.Text = " "

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "^p "

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

DoEvents

End With

Set tempDoc = Documents.Add

Set myResults = Documents.Add

Set res = myResults.Content

totalNumSents = copyDoc.Sentences.Count

' This should be enough, as we're doing it

' one letter at a time

Dim myText(30000) As String

allInits = ""

j = 0

For sn = 1 To totalNumSents

thisSentence = Trim(Replace(copyDoc.Sentences(sn).Text, vbCr, ""))

numWds = copyDoc.Sentences(sn).Words.Count

If InStr(Right(thisSentence, 2), ".") Then numWds = numWds - 1

If (numWds < minWords) = False And Len(thisSentence) > 10 Then

' Three goes at removing start & end 'stuff'

For k = 1 To 3

myInit = Left(thisSentence, 1)

If UCase(myInit) = LCase(myInit) Then

thisSentence = Mid(thisSentence, 2)

End If

myLast = Right(thisSentence, 1)

If UCase(myLast) = LCase(myLast) Then

thisSentence = Left(thisSentence, Len(thisSentence) - 1)

End If

Next k

myInit = LCase(Left(thisSentence, 1))

If InStr(allInits, myInit) = 0 Then

allInits = allInits & myInit

End If

j = j + 1

myText(j) = thisSentence

If j Mod 100 = 0 Then Debug.Print "Phrases logged: ", j

End If

DoEvents

Next sn

numSentencesNow = j

numInits = Len(allInits)

sp = ChrW(160)

For i = 1 To 4

sp = sp & sp

Next i

init = ""

For a = 1 To numInits

Debug.Print sp & "To go: " & Str(numInits - a)

Set rng = tempDoc.Content

rng.Text = ""

fstLttr = Mid(allInits, a, 1)

Debug.Print fstLttr

Application.ScreenUpdating = False

For sn = 1 To numSentencesNow

pText = myText(sn)

init = LCase(Left(pText, 1))

If init = fstLttr Then

rng.InsertAfter Text:=pText

totWords = rng.Words.Count

rng.Delete

If totWords > maxWords Then totWords = maxWords

For n = minWords To totWords

' Paste in the whole sentence

rng.InsertAfter Text:=pText & " Dummy"

rng.MoveStart wdWord, n

rng.Select

rng.Text = vbCr

rng.Expand wdParagraph

If InStr(rng, "Dummy") > 0 Then rng.Delete

rng.Collapse wdCollapseEnd

Next n

End If

DoEvents

Next sn

Application.ScreenUpdating = True

Set rng = tempDoc.Content

rng.Sort

tempDoc.Paragraphs(1).Range.Text = "Rubbish" & vbCr

rng.InsertAfter Text:=vbCr & "Rubbish" & vbCr

numParas = tempDoc.Paragraphs.Count

prevPara = tempDoc.Paragraphs(1).Range.Text

wasAmatch = False

numDupl = 1

For i = 2 To numParas

thisPara = tempDoc.Paragraphs(i).Range.Text

gottaMatch = (thisPara = prevPara)

If Not (gottaMatch) Then

If wasAmatch Then

lenDiff = Len(prevPara) - Len(Replace(prevPara, ".", ""))

If Not (numDupl < minFreqToList) And lenDiff = 0 Then

res.InsertAfter Text:=Trim(Replace(prevPara, vbCr, "")) \_

& " . . [" & Trim(Str(numDupl)) & "]" & vbCr

End If

numDupl = 1

End If

Else

numDupl = numDupl + 1

End If

wasAmatch = gottaMatch

prevPara = thisPara

Next i

Application.ScreenUpdating = True

DoEvents

Next a

tempDoc.Close SaveChanges:=False

copyDoc.Close SaveChanges:=False

myResults.Activate

Set rng = ActiveDocument.Content

rng.Sort

rng.InsertAfter Text:=vbCr

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="Duplicates List" & vbCr

Beep

timGone = Timer - strttime

Application.ScreenUpdating = True

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

m = Int(timGone / 60)

s = Int(timGone) - m \* 60

timeAll = "Time: " & Trim(Str(m)) & " m " & \_

Trim(Str(s)) & " s"

MsgBox ("Phrases checked: " & Trim(Str(numSentencesNow)) \_

& vbCr & vbCr & timeAll)

Exit Sub

' Switch the screen back on if there's an error

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub GoogleScholarFetchQuotes()**

**'** Paul Beverley - Version 24.08.24

' Launches selected text - with quotes - on Google Scholar

mySite = "https://scholar.google.com/scholar?q="

If Selection.Start = Selection.End Then

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.Start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.Start = startNow

End If

mySubject = Trim(Selection)

mySubject = Replace(mySubject, vbCr, "")

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = "%22" & mySubject & "%22"

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub FigureTableBoxRenumber()**

**'** Paul Beverley - Version 26.08.24

' Renumbers figures, tables and boxes when some have been deleted

totChaps = 50

totNumbers = 200

ReDim numUsed(totChaps, totNumbers) As Boolean

ReDim numNew(totChaps, totNumbers) As Integer

ReDim maxNum(totChaps) As Integer

For myEntity = 1 To 3

myWord = "Fig[.ure]{1,}"

If myEntity = 2 Then myWord = "Table"

If myEntity = 3 Then myWord = "Box"

' Reset the arrays

totChaps = 30

totNumbers = 100

For ch = 1 To totChaps

For Num = 1 To totNumbers

numNew(ch, Num) = Num

numUsed(ch, Num) = False

Next Num

DoEvents

Next ch

'Check which numbers are used.

' Read existing numbers

Set rng = ActiveDocument.Content

gotAny = False

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myWord & " " & "[0-9]{1,}.[0-9]{1,}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

finalChap = 1

myCount = 0

Do While rng.Find.Found = True

gotAny = True

spPos = InStr(rng, " ")

rng.MoveStart , spPos

ch = Int(Val(rng))

dotPos = InStr(rng, ".")

rng.MoveStart , dotPos

Num = Int(Val(rng))

If ch > finalChap Then finalChap = ch

numUsed(ch, Num) = True

If Num > maxNum(ch) Then maxNum(ch) = Num

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

If gotAny = True Then

' Check all numbers in all chapters

numsMissing = ""

For ch = 1 To finalChap

For Num = 1 To maxNum(ch)

If numUsed(ch, Num) = False Then

For i = Num To maxNum(ch)

numNew(ch, i) = numNew(ch, i) - 1

numMissed = Trim(Str(ch)) & "." & Trim(Str(Num))

If InStr(numsMissing, numMissed) = 0 Then

numsMissing = numsMissing & numMissed & "vbCr"

End If

Next i

End If

Next Num

DoEvents

Next ch

' If there are numbers missing, renumber all those above it

numsMissing = Replace(numsMissing, "vbCr", vbCr)

If numsMissing > "" Then

myResponse = MsgBox(myWord & vbCr & vbCr & \_

numsMissing & vbCr & "Make changes?", \_

vbQuestion + vbYesNo, "FigureTableBoxRenumber")

If myResponse <> vbYes Then Exit Sub

myWord = Replace(myWord, ".ure", ".ure ")

Set rng = ActiveDocument.Content

For ch = 1 To finalChap

For Num = 1 To maxNum(ch)

If numNew(ch, Num) <> Num Then

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

If myEntity = 1 Then

.Text = "(" & myWord & Trim(Str(ch)) & ".)" & \_

Trim(Str(Num)) & ">"

Else

.Text = "(" & myWord & " " & Trim(Str(ch)) & \_

".)" & Trim(Str(Num)) & ">"

End If

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "\1" & Trim(Str(numNew(ch, Num)))

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

End If

DoEvents

Next Num

DoEvents

Next ch

End If

End If

Next myEntity

Beep

End Sub

**Sub DateFormatAlyse()**

**'** Paul Beverley - Version 29.08.24

' Counts the occurrences of different date formats

' dateRange = "[12][09][0-9][0-9]>"

' or include 19th century – 18xx

dateRange = "[12][089][0-9][0-9]>"

doSummary = True

doHighColoubr = True

doFontColour = True

Dim myCount(10) As Integer

Dim myHighColour(10) As Variant

Dim myFontColour(10) As Variant

Dim mySummary(10) As Variant

myHighColour(1) = wdNoHighlight

myHighColour(2) = wdNoHighlight

myHighColour(3) = wdNoHighlight

myHighColour(4) = wdNoHighlight

myHighColour(5) = wdYellow

myHighColour(6) = wdBrightGreen

myHighColour(7) = wdPink

myHighColour(8) = wdTurquoise

myHighColour(9) = wdGray25

myHighColour(10) = wdGray50

myFontColour(1) = wdColorRed

myFontColour(2) = wdColorBlue

myFontColour(3) = wdColorGreen

myFontColour(4) = wdColorPink

myFontColour(5) = wdBlack

myFontColour(6) = wdBlack

myFontColour(7) = wdBlack

myFontColour(8) = wdBlack

myFontColour(9) = wdBlack

myFontColour(10) = wdBlack

CR = vbCr

allMonths = "Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec"

allDays = "Mon Tue Wed Thu Fri Sat Sun"

Set rng = ActiveDocument.Content

dateRange = "[ ]{1,}" & dateRange

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = dateRange

.Replacement.Text = ""

.Wrap = wdFindStop

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True

myIndex = myIndex + 1

If myIndex Mod 20 = 0 Then rng.Select

Do

rng.MoveStart , -1

DoEvents

Loop Until InStr(ChrW(13) & " ", Left(rng.Text, 1)) > 0

rng.MoveStart , 1

wd1 = Trim(rng.Words(1))

If Trim(rng.Words(1)) = "," Then wd1 = wd1 & ","

rng.MoveStart , -1

Do

rng.MoveStart , -1

DoEvents

Loop Until InStr(ChrW(13) & " ", Left(rng.Text, 1)) > 0

rng.MoveStart , 1

wd2 = Trim(rng.Words(1))

rng.MoveStart , -1

Do

rng.MoveStart , -1

DoEvents

Loop Until InStr(ChrW(13) & " ", \_

Left(rng.Text, 1)) > 0 Or rng.start = 0

rng.MoveStart , 1

wd3 = Trim(rng.Words(1))

' Debug.Print wd1 & "|" & wd2 & "|" & wd3 & "|"

n = 0

' December|3|? - 1

If n = 0 And Val(wd2) > 0 And LCase(wd2) = UCase(wd2) \_

And InStr(allDays, Left(wd3, 3)) = 0 \_

And InStr(allMonths, Left(wd1, 3)) > 0 Then n = 1

' February|31st|? - 2

If n = 0 And Val(wd2) > 0 And LCase(wd2) <> UCase(wd2) \_

And InStr(allDays, Left(wd3, 3)) = 0 Then n = 2

' March|of|31st - 3

If Val(wd3) > 0 And wd2 = "of" And \_

InStr(allMonths, Left(wd1, 3)) > 0 Then n = 3:

' 31st|July|? - 6

If n = 0 And Val(wd1) > 0 And LCase(wd1) <> UCase(wd1) \_

And InStr(allDays, Left(wd3, 3)) = 0 Then n = 6

' August|31|Wednesday - 7

If n = 0 And Val(wd2) > 0 And InStr(allDays, Left(wd3, 3)) > 0 \_

And InStr(allMonths, Left(wd1, 3)) > 0 Then n = 7

' 31|September|Tuesday - 8

If n = 0 And InStr(allDays, Left(wd3, 3)) > 0 And \_

InStr(allMonths, Left(wd2, 3)) > 0 \_

And Val(wd1) > 0 Then n = 8

' August|31st|Wednesday - 9

If n = 7 And LCase(wd2) <> UCase(wd2) Then n = 9

' 31st|September|Tuesday - 10

If n = 8 And LCase(wd1) <> UCase(wd1) Then n = 10

' April|? - 4

If n = 0 And Len(wd1) > 3 And \_

InStr(allMonths, Left(wd1, 3)) > 0 Then n = 4

' 3|May|? - 5

If n = 0 And Val(wd1) > 0 And \_

InStr(allMonths, Left(wd2, 3)) > 0 Then n = 5

myCount(n) = myCount(n) + 1

' Debug.Print n

cn = cn + 1

If cn Mod 10 = 0 Then rng.Select

If n > 0 Then

' Reduce range of highlighting

Select Case n

Case 1: mv = 1

Case 2: mv = 1

Case 3: mv = 0

Case 4: mv = 2

Case 5: mv = 1

Case 6: mv = 1

Case 7: mv = 0

Case 8: mv = 0

Case 9: mv = 0

Case 10: mv = 0

If n < 7 And n <> 3 Then rng.MoveStart wdWord, 1

If n = 4 Then rng.MoveStart wdWord, 1

End Select

If mv > 0 Then rng.MoveStart wdWord, mv

If doHighColour = True Then \_

rng.HighlightColorIndex = myHighColour(n)

If doFontColour = True Then \_

rng.Font.Color = myFontColour(n)

If doSummary = True Then \_

mySummary(n) = mySummary(n) & rng.Text & CR

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Selection.HomeKey Unit:=wdStory

Documents.Add

Selection.TypeText Text:="Date format analysis" & CR & CR

ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading2

For n = 1 To 10

Select Case n

Case 1: typeDescrip = "1) 3 July 2024" & vbTab & vbTab

Case 2: typeDescrip = "2) 31st December 2024" & vbTab

Case 3: typeDescrip = "3) 31st of December 2024" & vbTab

Case 4: typeDescrip = "4) December 2024" & vbTab & vbTab

Case 5: typeDescrip = "5) December 31, 2024" & vbTab

Case 6: typeDescrip = "6) December 31st, 2024" & vbTab

Case 7: typeDescrip = "7) Tuesday, 31 July 2024" & vbTab

Case 8: typeDescrip = "8) Tuesday, July 31, 2024" & vbTab

Case 9: typeDescrip = "9) Tuesday, 31st July 2024" & vbTab

Case 10: typeDescrip = "10) Tuesday, July 31st, 2024"

End Select

Selection.TypeText Text:=typeDescrip & vbTab & \_

Str(myCount(n)) & CR

Next n

Selection.HomeKey Unit:=wdStory

If doSummary = True Then

Documents.Add

For i = 1 To 10

Selection.TypeText Text:=mySummary(i) & ChrW(12)

DoEvents

Next i

End If

Beep

End Sub

**Sub HyphenAlyse()**

**'** Paul Beverley - Version 05.09.24

' Creates a frequency list of all possible hyphenations

myList = "anti,cross,eigen,hyper,inter,meta,mid,multi," \_

& "non,over,post,pre,pseudo,quasi,semi,sub,super,un"

myList = "anti,cross,eigen,hyper,inter,meta,mid,multi," \_

& "non,over,post,pre,pseudo,quasi,semi,sub,super,un"

includeNumbers = True

deleteTableBorders = True

lighterColour = wdGray25

' lighterColour = wdColor50

Dim myResult As String

myList = "," & myList

myList = Replace(myList, ",,", ",")

pref = Split(myList, ",")

Set FUT = ActiveDocument

doingSeveralMacros = (InStr(FUT.Name, "zzTestFile") > 0)

sp = ChrW(160)

sp = sp & sp & sp

sp = sp & sp & sp

sp = sp & sp & sp

If doingSeveralMacros = False Then

myResponse = MsgBox(" HyphenAlyse" & vbCr & vbCr & \_

"Analyse hyphenated words?", vbQuestion \_

+ vbYesNoCancel, "HyphenAlyse")

If myResponse <> vbYes Then Exit Sub

End If

Dim pr(8000) As String

strttime = Timer

Set rng = ActiveDocument.Content

Documents.Add

Set myDoc = ActiveDocument

Selection.FormattedText = rng.FormattedText

Selection.EndKey Unit:=wdStory

Application.ScreenUpdating = False

On Error GoTo ReportIt

If myDoc.Endnotes.Count > 0 Then Selection.FormattedText = \_

myDoc.StoryRanges(wdEndnotesStory).FormattedText

If myDoc.Footnotes.Count > 0 Then Selection.FormattedText = \_

myDoc.StoryRanges(wdFootnotesStory).FormattedText

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Font.StrikeThrough = True

.Wrap = wdFindContinue

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

allTheText = myDoc.Content.Text

myDoc.Content.Text = LCase(allTheText)

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8217) & "[!a-z]"

.Wrap = wdFindContinue

.Replacement.Text = "!!"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

If includeNumbers = True Then

schStr = "[a-z0-9]{1,}[-^=][0-9a-z-]{1,}"

Else

schStr = "[a-z]{1,}[-^=][a-z-]{1,}"

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = schStr

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

' Find all hyphenated/dashed word pairs

myPairs = 0

allWords = ","

Do While rng.Find.Found = True

wdPair = Replace(rng.Text, ChrW(8211), "-")

If InStr(allWords, "," & wdPair & ",") = 0 \_

And (UCase(wdPair) <> wdPair) Then

myPairs = myPairs + 1

pr(myPairs) = wdPair

allWords = allWords & wdPair & ","

If myPairs Mod 20 = 0 Then

If doingSeveralMacros = False Then \_

Debug.Print rng.Text & " " & myPairs

StatusBar = sp & rng.Text & " " & myPairs

End If

End If

If Right(wdPair, 1) <> "s" Then

wdPairs = wdPair & "s"

If InStr(allWords, "," & wdPairs & ",") = 0 Then

myPairs = myPairs + 1

pr(myPairs) = wdPairs

allWords = allWords & wdPairs & ","

If myPairs Mod 20 = 0 Then

If doingSeveralMacros = False Then \_

Debug.Print rng.Text, myPairs

StatusBar = sp & rng.Text & " " & myPairs

DoEvents

End If

End If

End If

rng.Find.Execute

Loop

' Collect words with each prefix

For i = 1 To UBound(pref)

hPos = Len(pref(i))

allPreWords = ","

If includeNumbers = True Then

schStr = "<" & pref(i) & "[0-9a-z]{2,}"

Else

schStr = "<" & pref(i) & "[a-z]{2,}"

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = schStr

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

wd = rng.Text

If InStr(wd, "-") = 0 Then wd = Left(wd, hPos) \_

& "-" & Mid(wd, hPos + 1)

If InStr(allPreWords, "," & wd & ",") = 0 And \_

InStr(allWords, "," & wd & ",") = 0 Then

myPairs = myPairs + 1

pr(myPairs) = wd

allPreWords = allPreWords & wd & ","

allWords = allWords & wd & ","

If myPairs Mod 20 = 0 Then

If doingSeveralMacros = False Then \_

Debug.Print wd & " " & myPairs

StatusBar = sp & wd & " " & myPairs

DoEvents

End If

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Next i

' Collect word pairs with each prefix, e.g. "mid height"

For i = 1 To UBound(pref)

hPos = Len(pref(i))

If includeNumbers = True Then

schStr = "<" & pref(i) & " [0-9a-z]{2,}"

Else

schStr = "<" & pref(i) & " [a-z]{2,}"

End If

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<" & pref(i) & " [0-9a-z]{2,}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

wd = rng.Text

If InStr(wd, " ") = 0 Then wd = Left(wd, hPos) \_

& " " & Mid(wd, hPos + 1)

wd = Replace(wd, " ", "-")

If InStr(allPreWords, "," & wd & ",") = 0 And \_

InStr(allWords, "," & wd & ",") = 0 Then

myPairs = myPairs + 1

pr(myPairs) = wd

allPreWords = allPreWords & wd & ","

If myPairs Mod 20 = 0 Then

If doingSeveralMacros = False Then \_

Debug.Print wd & " " & myPairs

StatusBar = sp & wd & " " & myPairs

DoEvents

End If

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

Next i

halfTime = Timer

' Count the frequencies

Selection.HomeKey Unit:=wdStory

Selection.TypeText vbCr & vbCr

Selection.HomeKey Unit:=wdStory

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

allText = " " & ActiveDocument.Range.Text & " "

' At this point, change all "^p" to "^p "

' all punctuation to " "

chs = " , . ! : ; [ ] { } ( ) / \ + "

chs = chs & ChrW(8220) & " "

chs = chs & ChrW(8221) & " "

chs = chs & ChrW(8201) & " "

chs = chs & ChrW(8222) & " "

chs = chs & ChrW(8217) & " "

chs = chs & ChrW(8216) & " "

chs = chs & ChrW(8212) & " "

chs = chs & ChrW(8722) & " "

chs = chs & vbCr & " "

chs = chs & vbTab & " "

' To force space at start; no space at end

' i.e. one space for each character that

' needs changing to a space

chs = " " & chs & " "

chs = Replace(chs, " ", " ")

chs = Replace(chs, " ", " ")

chs = Left(chs, Len(chs) - 1)

chars = Split(chs, " ")

For i = 1 To UBound(chars)

allText = Replace(allText, chars(i), " ")

Next i

allText = Replace(allText, " ", " ")

cnt = Len(allText)

For i = 1 To myPairs

totFinds = 0

thisFind = ""

Set rng = ActiveDocument.Content

myTot = rng.End

wdHyph = pr(i)

wd = Replace(wdHyph, "-", "")

For j = 1 To 4

Select Case j

Case 1: schWd = wdHyph

Case 2: schWd = Replace(wdHyph, "-", " ")

Case 3: schWd = wd

Case 4: schWd = Replace(wdHyph, "-", ChrW(8211))

End Select

sc = " " & schWd & " "

myCount = Len(Replace(allText, sc, sc & "!")) - cnt

If myCount > 0 Then

totFinds = totFinds + 1

Selection.HomeKey Unit:=wdStory

thisFind = thisFind & schWd & " . ." & \_

Str(myCount) & ":"

Else

thisFind = thisFind & ":"

End If

DoEvents

Next j

If (myPairs - i) Mod 20 = 0 Then

If doingSeveralMacros = False Then \_

Debug.Print "To go: ", myPairs - i

StatusBar = sp & "To go: " & myPairs - i

End If

If Len(thisFind) > 8 Then myResult = myResult & "%" & \_

wd & "%" & thisFind & "!"

Next i

myResult = Replace(myResult, ":!", vbCr)

myResult = Replace(myResult, ":", vbTab)

Selection.WholeStory

Selection.Delete

Set rng = ActiveDocument.Content

rng.InsertAfter myResult

Selection.Sort SortOrder:=wdSortOrderAscending

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "%\*%"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Forward = True

.MatchCase = False

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

End With

Selection.HomeKey Unit:=wdStory

Selection.TypeText "Hyphenation use"

startTable = Selection.End + 1

ActiveDocument.Paragraphs(1).Style = \_

ActiveDocument.Styles(wdStyleHeading1)

Selection.Start = startTable

Selection.End = ActiveDocument.Range.End

Selection.ConvertToTable Separator:=wdSeparateByTabs

Set TB = ActiveDocument.Tables(1)

For i = 1 To TB.Rows.Count

q = q + 1

If q Mod 20 = 0 Then

Debug.Print "Formatting results: " & q

StatusBar = sp & "Formatting results: " & q

DoEvents

End If

q = 0

Num = 0

For j = 1 To 4

If Len(TB.Cell(i, j).Range.Text) > 2 Then Num = Num + 1

Next j

If Num = 1 Then

For j = 1 To 4

TB.Cell(i, j).Range.Font.ColorIndex = lighterColour

Next j

End If

Next i

Set TB = ActiveDocument.Tables(1)

For i = 1 To TB.Rows.Count

q = q + 1

If q Mod 20 = 0 Then

Debug.Print "Formatting results: " & q

StatusBar = sp & "Formatting results: " & q

DoEvents

End If

q = 0

For j = 1 To 4

hyphPos = 0

txt = TB.Cell(i, j).Range.Text

hyphPos = InStr(txt, "-")

dashPos = InStr(txt, ChrW(8211))

tstText = txt

If hyphPos + dashPos > 0 Then

tstText = "," & Left(txt, hyphPos + dashPos \_

- 1) & ","

If InStr(myList, tstText) > 0 Then

TB.Cell(i, j).Range.Font.ColorIndex = wdBlue

End If

Else

For k = 1 To UBound(pref)

If InStr("," & txt, "," & pref(k)) > 0 Then

TB.Cell(i, j).Range.Font.ColorIndex = wdBlue

End If

Next k

End If

Next j

Next i

For i = 1 To TB.Rows.Count

q = q + 1

If q Mod 20 = 0 Then

Debug.Print "Formatting results: " & q

StatusBar = sp & "Formatting results: " & q

DoEvents

End If

q = 0

s = 0

If Len(TB.Cell(i, 1).Range.Text) > 2 Then s = s + 1

If Len(TB.Cell(i, 3).Range.Text) > 2 Then s = s + 1

If Len(TB.Cell(i, 4).Range.Text) > 2 Then s = s + 1

If Len(TB.Cell(i, 2).Range.Text) > 2 And \_

Len(TB.Cell(i, 4).Range.Text) > 2 Then s = 2

If s > 1 Then

For j = 1 To 4

TB.Cell(i, j).Range.Font.ColorIndex = wdRed

Next j

End If

If InStr(TB.Cell(i, 1).Range.Text, "ly-") > 0 And \_

Len(TB.Cell(i, 2).Range.Text) > 2 Then

For j = 1 To 4

TB.Cell(i, j).Range.Font.ColorIndex = wdRed

Next j

End If

Next i

StatusBar = sp & "Formatting results"

allText = ActiveDocument.Content

For Each myCell In TB.Range.Cells

q = q + 1

If q Mod 20 = 0 Then

Debug.Print "Formatting results: " & q

StatusBar = sp & "Formatting results: " & q

DoEvents

End If

myText = myCell.Range.Text

Set rng = myCell.Range.Duplicate

rng.End = rng.Start + 1

myColour = rng.Font.ColorIndex

i = InStr(myText, " . .")

If myColour = lighterColour And i > 2 Then

myWord = Left(myText, i - 1)

If Right(myWord, 1) = "s" Then

mySingular = Left(myText, i - 2)

If InStr(allText, mySingular & " . .") > 0 Then \_

myCell.Range.Font.Color = wdColorAutomatic

myTest = Replace(mySingular, "-", "")

If InStr(allText, mySingular & " . .") > 0 Then \_

myCell.Range.Font.Color = wdColorAutomatic

myTest = Replace(mySingular, "-", " ")

If InStr(allText, myTest & " . .") > 0 Then \_

myCell.Range.Font.Color = wdColorAutomatic

End If

If InStr(allText, myWord & "s . .") > 0 Then \_

myCell.Range.Font.Color = wdColorAutomatic

myTest = Replace(myWord, "-", "")

If InStr(allText, myTest & "s . .") > 0 Then \_

myCell.Range.Font.Color = wdColorAutomatic

myTest = Replace(myText, "-", " ")

If InStr(allText, myWord & "s . .") > 0 Then \_

myCell.Range.Font.Color = wdColorAutomatic

End If

Next myCell

TB.AutoFitBehavior (wdAutoFitContent)

If deleteTableBorders = True Then

TB.Borders(wdBorderTop).LineStyle = wdLineStyleNone

TB.Borders(wdBorderLeft).LineStyle = wdLineStyleNone

TB.Borders(wdBorderBottom).LineStyle = wdLineStyleNone

TB.Borders(wdBorderRight).LineStyle = wdLineStyleNone

TB.Borders(wdBorderHorizontal).LineStyle = wdLineStyleNone

TB.Borders(wdBorderVertical).LineStyle = wdLineStyleNone

End If

Selection.HomeKey Unit:=wdStory

timNow = Timer

Application.ScreenUpdating = True

If doingSeveralMacros = False Then

timGone = timNow - strttime

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

m = Int(timGone / 60)

s = Int(timGone) - m \* 60

timeAll = "Time: " & Trim(Str(m)) & " m " & \_

Trim(Str(s)) & " s"

Selection.HomeKey Unit:=wdStory

numPairs = ActiveDocument.Tables(1).Rows.Count

MsgBox "Items: " & Trim(Str(numPairs)) & vbCr & vbCr \_

& timeAll

Else

FUT.Activate

End If

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub WordFrequency()**

**'** Paul Beverley - Version 20.09.24

' Counts the frequency of words in a document

' ignoreWords = " the and "

ignoreWords = ""

' Don't bother counting words with fewer characters than this

minChars = 3

' Don't bother displaying words with fewer occurrences than this

minCount = 2

' Include apostrophe-s

incApostrophe = False

timeNow = Timer

Dim WordDict As Object

Dim myWords() As String

Dim myWordsTable As Variant

CR = vbCr

CR2 = CR & CR

Set sourceDoc = ActiveDocument

Set rngFrom = sourceDoc.Content

Set myDoc = Documents.Add

myDoc.Content.Text = rngFrom.Text

Set rng = myDoc.Content

rng.Collapse wdCollapseEnd

If sourceDoc.Endnotes.Count > 0 Then \_

rng.Text = \_

sourceDoc.StoryRanges(wdEndnotesStory).Text

If sourceDoc.Footnotes.Count > 0 Then \_

rng.Text = \_

sourceDoc.StoryRanges(wdFootnotesStory).Text

DoEvents

Set rng = ActiveDocument.Content

rng.Text = LCase(rng.Text)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Wrap = wdFindContinue

' URLs

.Text = "http\*[ ^13]"

.Replacement.Text = "^p"

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

Debug.Print 1

' URLs

.Text = "www\*[ ^13]"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

DoEvents

Debug.Print 2

' hyphenated words

.Text = "--"

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

DoEvents

Debug.Print 3

' hyphenated words

.Text = "([a-z])[-^~]([a-z])"

.Replacement.Text = "\1hzzh\2"

.Execute Replace:=wdReplaceAll

DoEvents

Debug.Print 4

' it's

.Text = "<it['" & ChrW(8217) & "]s"

.Replacement.Text = "itpzzps"

.Execute Replace:=wdReplaceAll

DoEvents

Debug.Print 5

' apostrophes

If incApostrophe = True Then

.Text = "([a-z])['" & ChrW(8217) & "]([a-z])"

Else

.Text = "([a-z])['" & ChrW(8217) & "]([a-rt-z])"

End If

.Replacement.Text = "\1pzzp\2"

.Execute Replace:=wdReplaceAll

DoEvents

Debug.Print 6

' email addresses

.Text = "<[a-z.]{1,}\@[a-z.]{1,}>"

.Replacement.Text = "^p"

.Execute Replace:=wdReplaceAll

DoEvents

Debug.Print 7

' hard space, hard hyphen, slash, dashes

.Text = "[^s//^~^+^=,.]"

.Replacement.Text = " "

.Execute Replace:=wdReplaceAll

DoEvents

End With

totTime1 = Timer - timeNow

' Initialize the dictionary

Set WordDict = CreateObject("Scripting.Dictionary")

WordDict.CompareMode = vbTextCompare

' Loop through each paragraph in the document

For Each myPar In myDoc.Paragraphs

' Split the paragraph into words

myText = LCase(myPar.Range.Text)

myText = Replace(myText, " ", " ")

myWords = Split(myText, " ")

' Loop through each word in the paragraph

For i = LBound(myWords) To UBound(myWords)

myWord = Trim(myWords(i))

If UCase(myWord) <> myWord Then

newWord = ""

For j = 1 To Len(myWord)

myChar = Mid(myWord, j, 1)

If UCase(myChar) <> myChar \_

Then newWord = newWord & myChar

If AscW(myChar) = 8217 Then Exit For

Next j

Debug.Print newWord

myWord = newWord

' Only consider words with minChars or more characters

If Len(myWord) >= minChars And \_

InStr(ignoreWords, " " & newWord & " ") = 0 Then

If WordDict.Exists(myWord) Then

WordDict(myWord) = WordDict(myWord) + 1

Else

WordDict.Add myWord, 1

End If

End If

End If

Next i

DoEvents

Next myPar

totTime2 = Timer - timeNow

' Use the new document to display the results

Set rng = myDoc.Content

rng.Text = "Word Frequency List (descending):" & CR2

' Sort the dictionary by frequency

myWordsTable = WordDict.Keys

myCount = WordDict.Count

For i = 0 To myCount - 1

myWord = myWordsTable(i)

If WordDict(myWord) >= minCount Then \_

myDoc.Content.InsertAfter myWord & vbTab & WordDict(myWord) & vbCr

If i Mod 50 = 0 Then DoEvents

Next i

totTime3 = Timer - timeNow

' sort out and display results

Set rng = myDoc.Content

Set rng = ActiveDocument.Content

With rng.Find

.Text = "hzzh"

.Replacement.Text = "-"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

.Text = "pzzp"

.Replacement.Text = ChrW(8217)

.Execute Replace:=wdReplaceAll

DoEvents

End With

DoEvents

rng.MoveStart wdParagraph, 2

rng.ConvertToTable Separator:=wdSeparateByTabs

DoEvents

Set tbl = myDoc.Content.Tables(1)

DoEvents

tbl.AutoFitBehavior (wdAutoFitContent)

DoEvents

myDoc.Content.Tables(1).Sort ExcludeHeader:=False, \_

FieldNumber:=1, \_

SortFieldType:=wdSortFieldAlphanumeric, \_

SortOrder:=wdSortOrderAscending

DoEvents

tbl.Range.Copy

DoEvents

Selection.EndKey Unit:=wdStory

Selection.InsertAfter Text:=CR2 & CR2 & "Word Frequency List (alphabetic):" & CR2

Selection.MoveStart wdParagraph, 4

Selection.MoveEnd wdParagraph, -1

Selection.Paragraphs(1).Style = "Heading 1"

Selection.EndKey Unit:=wdStory

DoEvents

Selection.Paste

DoEvents

' Sort the table by the second column (frequency) in descending order

myDoc.Tables(1).Sort ExcludeHeader:=False, \_

FieldNumber:=2, \_

SortFieldType:=wdSortFieldNumeric, \_

SortOrder:=wdSortOrderDescending

Selection.HomeKey Unit:=wdStory

Set rng = ActiveDocument.Content

rng.Paragraphs(1).Style = "Heading 1"

totTime4 = Timer - timeNow

Debug.Print ((Int(10 \* totTime1) / 10) & " Seconds" & vbCr & vbCr \_

& (Int(10 \* (totTime2 - totTime1)) / 10) & " Seconds" & vbCr & vbCr \_

& (Int(10 \* (totTime3 - totTime2)) / 10) & " Seconds" & vbCr & vbCr \_

& (Int(10 \* (totTime4 - totTime3)) / 10) & " Seconds" & vbCr & vbCr \_

& (Int(10 \* (Timer - timeNow)) / 10) & " Total" & vbCr & vbCr)

DoEvents

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

MsgBox "The macro has finished, but click OK," & CR \_

& "then wait until Word has formatted the file;" & CR \_

& "the cursor will start to flash when it is ready."

' Clean up

Set WordDict = Nothing

End Sub

**Sub CollinsFetch()**

**'** Paul Beverley - Version 25.09.24

' Launches selected text on Google

alsoCopySubject = False

mySite = "https://www.collinsdictionary.com/dictionary/english/"

If Selection.start = Selection.End Then

Selection.Expand wdWord

If Len(Selection) < 3 Then

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.start = startNow

End If

If alsoCopySubject = True Then Selection.Copy

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

Debug.Print mySite & mySubject

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub OUPFetchPremiumMac()**

**'** Paul Beverley - Version 25.09.24

' Launches selected text to OUP Premium website on a Mac

mySite = "https://premium.oxforddictionaries.com/definition/english/"

If Selection.start = Selection.End Then

Selection.Expand wdWord

If LCase(Selection) = UCase(Selection) Then

Selection.Collapse wdCollapseStart

Selection.MoveStart wdWord, -1

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.start = startNow

End If

Selection.Copy

ActiveDocument.FollowHyperlink Address:=mySite

End Sub

**Sub OxfordPremiumThesaurusFetch()**

**'** Paul Beverley - Version 26.09.24

' Launches selected text on OUP Premium thesaurus

alsoCopySubject = False

mySite = "https://premium.oxforddictionaries.com/definition/english-thesaurus/"

If Selection.start = Selection.End Then

Selection.Expand wdWord

If Len(Selection) < 3 Then

Selection.Collapse wdCollapseStart

Selection.MoveLeft , 1

Selection.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Else

endNow = Selection.End

Selection.MoveLeft wdWord, 1

startNow = Selection.start

Selection.End = endNow

Selection.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(Selection.Text, 1)) > 0

Selection.MoveEnd , -1

DoEvents

Loop

Selection.start = startNow

End If

If alsoCopySubject = True Then Selection.Copy

mySubject = Trim(Selection)

mySubject = Replace(mySubject, " ", "+")

mySubject = Replace(mySubject, "&", "%26")

mySubject = Replace(mySubject, ChrW(8217), "'")

Debug.Print mySite & mySubject

ActiveDocument.FollowHyperlink Address:=mySite & mySubject

Selection.Collapse wdCollapseEnd

End Sub

**Sub SpellcheckingOffForTurquoiseWords()**

**'** Paul Beverley - Version 30.09.24

' Set any words in turquoise to not be spellchecked

Selection.HomeKey Unit:=wdStory

myTrack = ActiveDocument.TrackRevisions

ActiveDocument.TrackRevisions = False

Application.ScreenUpdating = False

' For each highlight colour

modifiedCount = 0

fNotes = ActiveDocument.Footnotes.Count

eNotes = ActiveDocument.Endnotes.Count

For j = 1 To 3

If j = 1 And fNotes = 0 Then j = 2

If j = 2 And eNotes = 0 Then j = 3

Select Case j

Case 1: Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case 2: Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

Case 3: Set rng = ActiveDocument.Content

End Select

' Loop through all the words in the document

For Each wd In rng.Words

' Check if the word is highlighted in turquoise

If wd.HighlightColorIndex = wdTurquoise Then

' Apply SpellingChecked to the word

wd.SpellingChecked = True

modifiedCount = modifiedCount + 1

DoEvents

End If

Next wd

Next

Application.ScreenUpdating = True

ActiveDocument.TrackRevisions = myTrack

MsgBox """Spelling Checked"" has been applied to " & modifiedCount & " turquoise-highlighted words."

End Sub

**Sub HashtagCamelizer()**

**'** Paul Beverley - Version 07.10.24

' Finds the next # and camel-cases the tag following

' myFontColour = 0

myFontColour = wdColorBlue

myTwos = " aa ad ae ah ai am an ar as at aw ax " & \_

"ay ba be bo by ch da di do ea ee ef eh el em " & \_

"en er es ex fa fy gi go gu ha he hi ho id if " & \_

"in io is it jo ka ko ky la li lo ma me mi mo mu " & \_

"my na ne no nu ny ob od oe of oh oiom on oo op " & \_

"or os ou ow ox oy pa pi po re sh si so st ta te " & \_

"ti to ug um un up ur us ut we wo xi ye yo yu zo "

myPNs = " monday tuesday wednesday thursday friday " & \_

"saturday sunday january february april june july " & \_

"august september october november december beverley"

restart:

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "#"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

If .Found = False Then

Beep

myResponse = MsgBox("No more hashtags found!", \_

vbOKOnly, "HashtagCamelizer")

Selection.EndKey Unit:=wdStory

Exit Sub

End If

End With

' Two-word hashtags

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

myText = LCase(Trim(rng.Text))

' Uppercase to avoid "july" etc being spelling errors!

myPN = Split(myPNs)

For i = 1 To UBound(myPN)

myText = Replace(myText, myPN(i), UCase(myPN(i)))

Next i

rng.Text = LCase(rng.Text)

For i = 2 To Len(myText) - 2

wd1 = Left(myText, i)

wd2 = Mid(myText, i + 1)

If Len(wd1) = 2 Then

wd1good = InStr(myTwos, " " & wd1 & " ") > 0

Else

wd1good = Application.CheckSpelling(wd1)

End If

If Len(wd2) = 2 Then

wd2good = InStr(myTwos, " " & wd2 & " ") > 0

Else

wd2good = Application.CheckSpelling(wd2)

End If

If wd1good And wd2good Then

rng.Characters.First = UCase(rng.Characters.First)

rng.MoveStart , i

rng.End = rng.start + 1

rng.Text = UCase(rng.Text)

rng.Expand wdWord

If myFontColour > 0 Then

rng.Font.Color = myFontColour

End If

rng.Collapse wdCollapseEnd

rng.Select

GoTo restart

End If

DoEvents

Next i

' Three-word hashtags

For i = 2 To Len(myText) - 4

For j = i + 2 To Len(myText) - 2

wd1 = Left(myText, i)

wd2 = Mid(myText, i + 1, j - i)

wd3 = Mid(myText, j + 1)

If Len(wd1) = 2 Then

wd1good = InStr(myTwos, " " & wd1 & " ") > 0

Else

wd1good = Application.CheckSpelling(wd1)

End If

If Len(wd2) = 2 Then

wd2good = InStr(myTwos, " " & wd2 & " ") > 0

Else

wd2good = Application.CheckSpelling(wd2)

End If

If Len(wd3) = 2 Then

wd3good = InStr(myTwos, " " & wd3 & " ") > 0

Else

wd3good = Application.CheckSpelling(wd3)

End If

If wd1good And wd2good And wd3good Then

rng.Characters.First = UCase(rng.Characters.First)

rng.Characters(i + 1).Text = UCase(rng.Characters(i + 1).Text)

rng.Characters(j + 1).Text = UCase(rng.Characters(j + 1).Text)

rng.Expand wdWord

If myFontColour > 0 Then

rng.Font.Color = myFontColour

End If

rng.Collapse wdCollapseEnd

rng.Select

GoTo restart

End If

Next j

DoEvents

Next i

' Four-word hashtags

For i = 2 To Len(myText) - 6

For j = i + 2 To Len(myText) - 4

For k = j + 2 To Len(myText) - 2

wd1 = Left(myText, i)

wd2 = Mid(myText, i + 1, j - i)

wd3 = Mid(myText, j + 1, k - j)

wd4 = Mid(myText, k + 1)

If Len(wd1) = 2 Then

wd1good = InStr(myTwos, " " & wd1 & " ") > 0

Else

wd1good = Application.CheckSpelling(wd1)

End If

If Len(wd2) = 2 Then

wd2good = InStr(myTwos, " " & wd2 & " ") > 0

Else

wd2good = Application.CheckSpelling(wd2)

End If

If Len(wd3) = 2 Then

wd3good = InStr(myTwos, " " & wd3 & " ") > 0

Else

wd3good = Application.CheckSpelling(wd3)

End If

If Len(wd4) = 2 Then

wd4good = InStr(myTwos, " " & wd4 & " ") > 0

Else

wd4good = Application.CheckSpelling(wd4)

End If

If wd1good And wd2good And wd3good And wd4good Then

rng.Characters.First = UCase(rng.Characters.First)

rng.Characters(i + 1).Text = UCase(rng.Characters(i + 1).Text)

rng.Characters(j + 1).Text = UCase(rng.Characters(j + 1).Text)

rng.Characters(k + 1).Text = UCase(rng.Characters(k + 1).Text)

rng.Expand wdWord

If myFontColour > 0 Then

rng.Font.Color = myFontColour

End If

rng.Collapse wdCollapseEnd

rng.Select

GoTo restart

End If

Next k

Next j

DoEvents

Next i

' Five-word hashtags

Beep

myResponse = MsgBox("This might take a while. Continue?", \_

vbQuestion + vbYesNoCancel, "HashtagCamelizer")

If myResponse <> vbYes Then Exit Sub

For i = 2 To Len(myText) - 8

For j = i + 2 To Len(myText) - 6

For k = j + 2 To Len(myText) - 4

For m = k + 2 To Len(myText) - 2

wd1 = Left(myText, i)

wd2 = Mid(myText, i + 1, j - i)

wd3 = Mid(myText, j + 1, k - j)

wd4 = Mid(myText, k + 1, m - k)

wd5 = Mid(myText, m + 1)

If Len(wd1) = 2 Then

wd1good = InStr(myTwos, " " & wd1 & " ") > 0

Else

wd1good = Application.CheckSpelling(wd1)

End If

If Len(wd2) = 2 Then

wd2good = InStr(myTwos, " " & wd2 & " ") > 0

Else

wd2good = Application.CheckSpelling(wd2)

End If

If Len(wd3) = 2 Then

wd3good = InStr(myTwos, " " & wd3 & " ") > 0

Else

wd3good = Application.CheckSpelling(wd3)

End If

If Len(wd4) = 2 Then

wd4good = InStr(myTwos, " " & wd4 & " ") > 0

Else

wd4good = Application.CheckSpelling(wd4)

End If

If Len(wd5) = 2 Then

wd5good = InStr(myTwos, " " & wd5 & " ") > 0

Else

wd5good = Application.CheckSpelling(wd5)

End If

If wd1good And wd2good And wd3good And wd4good And wd5good Then

rng.Characters.First = UCase(rng.Characters.First)

rng.Characters(i + 1).Text = UCase(rng.Characters(i + 1).Text)

rng.Characters(j + 1).Text = UCase(rng.Characters(j + 1).Text)

rng.Characters(k + 1).Text = UCase(rng.Characters(k + 1).Text)

rng.Characters(m + 1).Text = UCase(rng.Characters(m + 1).Text)

rng.Expand wdWord

If myFontColour > 0 Then

rng.Font.Color = myFontColour

End If

rng.Collapse wdCollapseEnd

rng.Select

GoTo restart

End If

Next m

Next k

Next j

DoEvents

Next i

Beep

rng.Select

myResponse = MsgBox("Can't work this one out, sorry!", \_

vbOKOnly, "HashtagCamelizer")

End Sub

**Sub MacroMenuSearch()**

**'** Paul Beverley - Version 09.10.24

' Searches the Macro Menu (or The Book) for two (parts of) words

myColour = wdTurquoise

Dim hereNow As Long

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

' Check if it's a two-word myPar

' If so load and find

rng.MoveEnd , -1

If rng.Words.Count = 2 And rng.Fields.Count = 0 Then

pbWordOne = LCase(Trim(rng.Words(1)))

pbWordTwo = LCase(Trim(rng.Words(2)))

rng.Collapse wdCollapseEnd

rng.Select

GoTo doSearch

End If

If Selection.start = Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

GoTo doSearch

End If

' If text is selected, do what?

Set rng = Selection.Range.Duplicate

myTest = LCase(rng)

getNewSearch = True

If InStr(myTest, pbWordTwo) And InStr(myTest, \_

pbWordOne) > 0 Then getNewSearch = False

If rng.Fields.Count > 0 Then getNewSearch = False

If getNewSearch = False Then

rng.Collapse wdCollapseEnd

GoTo doSearch

End If

rng.Collapse wdCollapseStart

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

pbWordOne = LCase(rng.Text)

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.Expand wdWord

If LCase(rng) = UCase(rng) Then

rng.MoveEnd , -1

rng.Expand wdWord

End If

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

pbWordTwo = LCase(rng.Text)

rng.Collapse wdCollapseEnd

doSearch:

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = pbWordOne

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True

' Check if the other word is in the paragraph

rng.Expand wdParagraph

gottaField = (rng.Fields.Count > 0)

rngTest = LCase(rng)

wd1Pos = InStr(rngTest, pbWordOne)

wd2Pos = InStr(rngTest, pbWordTwo)

If wd2Pos > 0 Then

If gottaField Then

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

rng.MoveEnd , -1

Do

rng.MoveStart wdWord, -1

Loop Until (InStr(rng, pbWordOne) + \_

InStr(rng, pbWordTwo)) > 0

GoTo theEnd

End If

If wd1Pos < wd2Pos Then

rng.End = rng.start + wd2Pos + Len(pbWordTwo) - 1

rng.MoveStart , wd1Pos - 1

GoTo theEnd

Else

rng.End = rng.start + wd1Pos + Len(pbWordOne) - 1

rng.MoveStart , wd2Pos - 1

GoTo theEnd

End If

End If

rng.Collapse wdCollapseEnd

If rng.End = hereNow Then

rng.Select

Selection.MoveDown , 1

Beep

MsgBox "Trouble!"

End If

hereNow = rng.End

rng.Find.Execute

DoEvents

Loop

Beep

Selection.EndKey Unit:=wdStory

myResponse = MsgBox("Can't find any more: """ & pbWordOne & \_

""" """ & pbWordTwo & """", vbOKOnly, "FindTwoWords")

Exit Sub

theEnd:

rng.Expand wdParagraph

rng.HighlightColorIndex = myColour

rng.Font.Size = 11.5

rng.Collapse wdCollapseEnd

rng.MoveEnd , -1

rng.Select

End Sub

**Sub MacroMenuSearchUp()**

**'** Paul Beverley - Version 09.10.24

' Searches the Macro Menu (or The Book) (upwards)

Selection.MoveUp , 1

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Collapse wdCollapseStart

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = pbWordOne

.Font.Size = 11.5

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = False

.MatchWildcards = False

.MatchWholeWord = False

.Execute

If .Found = False Then

Beep

Selection.HomeKey Unit:=wdStory

Else

rng.Expand wdParagraph

rng.Collapse wdCollapseStart

rng.Select

End If

End With

End Sub

**Sub CalloutTextboxAdd()**

**'** Paul Beverley - Version 17.10.24

' Adds a call-out box of the selected text

' If useTextbox = False then macro uses a single-cell table

useTextbox = False

If Selection.Type = wdNoSelection Then

Beep

MsgBox "Please select the text you want to call out.", \_

vbExclamation

Exit Sub

End If

Selection.Copy

Set rng = Selection.Range.Duplicate

rng.Expand wdParagraph

rng.Collapse wdCollapseEnd

If useTextbox = True Then

rng.InsertAfter Text:=vbCr & vbCr

rng.Collapse wdCollapseStart

rng.Select

hPosition = Selection.Information(wdHorizontalPositionRelativeToPage)

vPosition = Selection.Information(wdVerticalPositionRelativeToPage)

Dim shape As shape

Set shape = ActiveDocument.Shapes.AddTextbox(msoTextOrientationHorizontal, \_

hPosition, vPosition, 200, 50)

shape.TextFrame.TextRange.Paste

Selection.Collapse wdCollapseEnd

Else

rng.InsertAfter Text:=vbCr

rng.Collapse wdCollapseEnd

rng.Select

Set tbl = ActiveDocument.Tables.Add(Range:=rng, NumRows:=1, NumColumns:=1)

tbl.Cell(1, 1).Range.Paste

End If

End Sub

**Sub CountReferenceDates()**

**'** Paul Beverley - Version 06.10.24

' Counts numbers of references older or newer than five years

myYear = Year(Now())

myLowerLimit = 1900 ' to avoid 'funny' numbers

myLowYear = myYear - 5 ' from 2024 dates 2019 and lower are "old"

myUpperLimit = myYear + 1 ' to avoid 'funny' numbers

If Selection.Start = Selection.End Then

  MsgBox "Please select the region for the count."

  Exit Sub

End If

numOld = 0

numNew = 0

For Each pa In Selection.Range.Paragraphs

  For Each wd In pa.Range.Words

    myVal = Val(wd)

    If myVal > myLowerLimit And myVal < myUpperLimit Then

      If myVal > myLowYear Then

        numNew = numNew + 1

        wd.HighlightColorIndex = wdBrightGreen

      Else

        numOld = numOld + 1

        wd.HighlightColorIndex = wdYellow

      End If

      Exit For

    End If

  Next wd

  DoEvents

Next pa

MsgBox "Five or more: " & numOld & vbCr \_

     & "Less than five: " & numNew

End Sub

**Sub CitationAlyseAlpha()**

**'** Paul Beverley - Version 28.10.24

' Checks citations against references

myColour = wdBrightGreen

myFontColour = wdColorBlue

myDateColour = wdColorRed

includeNotes = True

notNames = "In From For Act Jan January Feb February "

notNames = notNames & " Mar March Apr April NotMay June "

notNames = notNames & " July Aug August Sept September Oct Initially "

notNames = notNames & " October Nov November Dec December Finally "

notNames = notNames & " Also As Conversely Correspondingly Consequently "

notNames = notNames & " Equally Furthermore Lastly Moreover However "

notNames = notNames & " Secondly Thirdly Similarly Additionally "

CR = vbCr

CR2 = CR & CR

myScreenOff = False

Dim myCol(4)

myCol(0) = wdYellow

myCol(1) = wdBrightGreen

myCol(2) = wdRed

myCol(3) = wdTurquoise

myCol(4) = wdGray25

' If text is selected, sort it

If Selection.start <> Selection.End Then

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.Expand wdParagraph

Selection.start = rng.start

Set rng = Selection.Range.Duplicate

If Right(rng.Text, 1) <> vbCr Then

rng.Collapse wdCollapseEnd

rng.Expand wdParagraph

Selection.End = rng.End

End If

myResponse = MsgBox("Sort selection by year?", vbQuestion \_

+ vbOK, "CitationAlyse")

If myResponse <> vbOK Then Exit Sub

If myScreenOff = True Then

Application.ScreenUpdating = False

On Error GoTo ReportIt

End If

Dim myPara As Paragraph

Set rng = Selection.Range.Duplicate

For Each myPara In rng.Paragraphs

myText = myPara.Range.Text

For i = 2 To Len(myText) - 3

myYearText = Mid(myText, i, 5)

If InStr("abcdefghijkl", Right(myYearText, 1)) = 0 Then \_

myYearText = Left(myYearText, 4)

myYear = Val(myYearText)

If myYear > 1000 And Left(myYearText, 1) <> " " Then

myPara.Range.InsertBefore Text:="]" & myYearText & "["

Exit For

End If

DoEvents

Next i

DoEvents

Next myPara

Selection.Sort

With Selection.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "\][0-9a-h]@\["

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Application.ScreenUpdating = True

Exit Sub

End If

' Main citation listing section

Beep

myResponse = MsgBox("Is the cursor placed in the first item" \_

& CR2 & "of the references list?", vbQuestion \_

+ vbYesNoCancel, "CitationAlyse")

If myResponse <> vbYes Then Exit Sub

On Error GoTo ReportIt

Application.ScreenUpdating = False

' copy references right to the end

Selection.Expand wdParagraph

Selection.Collapse wdCollapseStart

startRefs = Selection.start

Set rngOld = ActiveDocument.Content

rngOld.start = startRefs

rngOld.Copy

rngOld.Collapse wdCollapseStart

rngOld.start = 0

' create new file of the text (not refs) + notes

Set mainDoc = ActiveDocument

Documents.Add

Selection.Text = rngOld.Text

' Copy footnotes and endnotes, text only

gotFoots = (mainDoc.Footnotes.Count > 0)

gotEnds = (mainDoc.Endnotes.Count > 0)

DoEvents

If gotFoots = True And includeNotes = True Then

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr

Selection.Text = mainDoc.StoryRanges(wdFootnotesStory).Text

End If

If gotEnds = True And includeNotes = True Then

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr

Selection.Text = mainDoc.StoryRanges(wdEndnotesStory).Text

End If

DoEvents

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ",([0-9]{4})"

.Wrap = wdFindStop

.Replacement.Text = ", \1"

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute Replace:=wdReplaceAll

End With

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[a-z] \([A-Z]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True

endNow = rng.End

rng.MoveStart wdWord, -1

rng.MoveEnd wdCharacter, -1

rng.Select

rng.Text = LCase(rng.Text)

rng.MoveStart wdWord, -1

rng.Select

myMidWord = Left(rng, 3)

If myMidWord = "und" Or myMidWord = "and" Then

rng.MoveStart wdWord, -1

rng.Text = LCase(rng.Text)

End If

rng.Select

rng.start = endNow + 2

rng.End = endNow + 2

rng.Find.Execute

DoEvents

Loop

' Find and replace to create easy-to-search text

myList = "#ü|uu#û|qcq#~<([A-Z]{2}).|\1# in press| 2999#/| #" & \_

"~<([A-Z]).|\1#(| #)| #:| #;| #, | #" & \_

"~<([A-Z])> ([A-Z])|\1ü\2#~<([A-Z]{2})> ([A-Z])|\1ü\2#" & \_

"#.|ü#(|ü#)|ü#[|ü#]|ü#" & \_

" et alii|üetüal# et al|üetüal# & |üandü# und |üandü#-|û#" & \_

"~<de la |DeüLaü#~<De la |DeüLaü#~<De La |DeüLaü#" & \_

" e |üeü# y |üyü#~<van |Vanü#~<de |Deü#" & \_

" and |üandü#~<von |Vonü#~<van |Vanü#~<de |Deü#" & \_

"~<El |Elü#~<del |Delü#~<Del |Delü#~<la |Laü#~<La |Laü#" & \_

"~<le |Leü#~<di |Diü#~<dos |Dosü#~<la |Laü#~<le |Leü#" & \_

"~<der |Derü#üvon |üVonü#üvan |üVanü#üde |üDeü#üder |üDerü#" & \_

"üdi |üDiü#üdos |üDosü#üla |üLaü#üle |üLeü# ü| #ü | #" & \_

"~([0-9])ü|\1 #~ü([0-9])| \1# | #"

FandR = Split(myList, "#")

Set rng = ActiveDocument.Content

For i = 1 To UBound(FandR)

myFR = FandR(i)

If Left(myFR, 1) <> ChrW(124) And \_

InStr(myFR, ChrW(124)) > 0 Then

If Left(myFR, 1) = "~" Then

myWild = True

myFR = Mid(myFR, 2)

Else

myWild = False

End If

barpos = InStr(myFR, ChrW(124))

myFind = Left(myFR, barpos - 1)

myRep = Mid(myFR, barpos + 1)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = myRep

.Replacement.Highlight = True

.MatchWildcards = myWild

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

Next i

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[12][0-9]{3}[!0-9]"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

' Find & Do to find all years

Do While rng.Find.Found = True

myEnd = rng.End

myYear = Replace(rng, "ü", "")

myYear = Replace(myYear, ",", "")

myYear = Replace(myYear, ";", "")

myYear = Trim(myYear)

rng.Collapse wdCollapseStart

rng.MoveStart wdWord, -1

myInit = Left(rng, 1)

If UCase(myInit) <> myInit Or UCase(myInit) \_

= LCase(myInit) Then GoTo getNext

wd1 = Trim(rng)

rng.Collapse wdCollapseStart

rng.MoveStart wdWord, -1

wd2 = Trim(rng)

myInit = Left(wd2, 1)

gotTwo = (UCase(myInit) = myInit) And \_

(UCase(myInit) <> LCase(myInit))

' Find surname one and add to citeList1

citeList1 = citeList1 & wd1 & " " & myYear & CR

' Find surname two and add to citeList2

If gotTwo Then citeList2 = citeList2 & wd2 & \_

" " & wd1 & " " & myYear & CR

' Check if another date follows

rng.start = myEnd

rng.End = myEnd

Do

rng.MoveEnd wdWord, 1

If Len(rng) < 4 Then

rng.Collapse wdCollapseEnd

rng.MoveEnd wdWord, 1

End If

myYearNumber = Val(rng)

If Val(rng) > 1000 Then

myYear = Trim(rng)

citeList1 = citeList1 & wd1 & " " & myYear & CR

If gotTwo Then citeList2 = citeList2 & wd2 & \_

" " & wd1 & " " & myYear & CR

End If

rng.MoveStart wdWord, 1

myEnd = rng.start

DoEvents

Loop Until myYearNumber < 1000

getNext:

rng.start = myEnd

rng.End = myEnd

rng.Find.Execute

Loop

Set rng = ActiveDocument.Content

rng.Delete

Selection.TypeText Text:=citeList1

myEnd = ActiveDocument.Content.End

Selection.TypeText Text:=citeList2

Set rng = ActiveDocument.Content

rng.start = myEnd

' rng.Font.Italic = True

Set rng = ActiveDocument.Content

rng.Font.Color = myFontColour

' Change by F&R û to -, ü to space, uu to ü, qcq to û

myList = "#ü| #uu|ü#û|-#qcq|û# 2999| in press#"

FandR = Split(myList, "#")

Set rng = ActiveDocument.Content

For i = 1 To UBound(FandR)

myFR = FandR(i)

If Left(myFR, 1) <> ChrW(124) And \_

InStr(myFR, ChrW(124)) > 0 Then

barpos = InStr(myFR, ChrW(124))

myFind = Left(myFR, barpos - 1)

myRep = Mid(myFR, barpos + 1)

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = myFind

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = myRep

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

End With

DoEvents

End If

Next i

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

' Remove duplicates

numPars = ActiveDocument.Paragraphs.Count

For j = numPars To 2 Step -1

Set rng1 = ActiveDocument.Paragraphs(j).Range

Set rng2 = ActiveDocument.Paragraphs(j - 1).Range

If rng1 = rng2 Then rng1.Delete

DoEvents

Next j

' Highlight multiple part citations

numPars = ActiveDocument.Paragraphs.Count

Set rng = ActiveDocument.Content

allText = rng

myStrike = False

For i = 1 To numPars

myText = ActiveDocument.Paragraphs(i).Range.Text

numTimes = Len(Replace(allText, myText, myText & "!")) - Len(allText)

If numTimes > 1 And Len(myText) > 2 Then

addCol = Int(5 \* Rnd())

baseCol = Int(5 \* Rnd())

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = Replace(myText, vbCr, "")

.Wrap = wdFindStop

.Replacement.Text = ""

.MatchWildcards = False

.Execute

DoEvents

End With

Do While rng.Find.Found

rng.HighlightColorIndex = myCol(baseCol)

For j = 1 To rng.Characters.Count

If (j Mod 5) < 2 Then

rng.Characters(j).HighlightColorIndex = myCol(addCol)

End If

Next j

DoEvents

rng.Collapse wdCollapseEnd

rng.Find.Execute

Loop

End If

DoEvents

Next i

Options.DefaultHighlightColorIndex = oldColour

Selection.EndKey Unit:=wdStory

Selection.TypeText Text:=vbCr

wasEnd = Selection.End

Selection.Paste

Set rng = ActiveDocument.Content

rng.start = wasEnd

rng.Fields.Unlink

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p "

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = "^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceAll

DoEvents

End With

' Add author name to "ditto" type references

myPrevAuthor = ""

For i = 1 To rng.Paragraphs.Count

Set myRef = rng.Paragraphs(i).Range

myWord = Trim(myRef.Words(1))

If Len(myRef.Text) > 1 And Len(myWord) = 1 Then

myRef.Select

myWord = myRef.Words(2)

End If

mySample = Left(myRef.Text, 2)

If Len(myRef.Text) > 1 And LCase(mySample) = \_

UCase(mySample) Then

myRef.Words(1) = myPrevAuthor & " "

Else

myPrevAuthor = myWord

End If

DoEvents

Next i

Set rng = ActiveDocument.Content

rng.Sort SortOrder:=wdSortOrderAscending, \_

SortFieldType:=wdSortFieldAlphanumeric

DoEvents

Set rng = ActiveDocument.Content

rng.InsertAfter Text:=CR2

Set rng = ActiveDocument.Content

With rng.ParagraphFormat

.LineSpacingRule = wdLineSpaceSingle

.SpaceBefore = 12

.LeftIndent = CentimetersToPoints(2.5)

.FirstLineIndent = CentimetersToPoints(-2.5)

End With

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

' Do F&R

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<[0-9]{4}>"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.Replacement.Highlight = True

.MatchWildcards = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

Options.DefaultHighlightColorIndex = oldColour

Beep

Application.ScreenUpdating = True

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub UKUSAlyse()**

**'** Paul Beverley - Version 01.11.24

' Analyses variant spelling between UK and US English

minLengthSpell = 4

doCount = True

showCase = True

UKcount = 0

UScount = 0

CR = vbCr

UKwords = CR

USwords = CR

Selection.HomeKey Unit:=wdStory

i = ActiveDocument.Words.Count

iStart = i

StatusBar = "Spellchecking. To go: 100%"

On Error GoTo ReportIt

Application.ScreenUpdating = False

For Each wd In ActiveDocument.Words

myWrd = Replace(Trim(wd), ChrW(8217), "")

If Len(myWrd) >= minLengthSpell And wd.Font.StrikeThrough = False Then

UKok = Application.CheckSpelling(myWrd, \_

MainDictionary:=Languages(wdEnglishUK).NameLocal)

USok = Application.CheckSpelling(myWrd, \_

MainDictionary:=Languages(wdEnglishUS).NameLocal)

If UKok <> USok Then

If UKok Then

UKcount = UKcount + 1

If showCase = True Then

If InStr(UKwords, "!" & myWrd & CR) = 0 Then \_

UKwords = UKwords & "!" & myWrd & CR

Else

If InStr(UKwords, "!" & LCase(myWrd) & CR) = 0 Then \_

UKwords = UKwords & "!" & LCase(myWrd) & CR

End If

Else

UScount = UScount + 1

If showCase = True Then

If InStr(USwords, "$" & myWrd & CR) = 0 Then \_

USwords = USwords & "$" & myWrd & CR

Else

If InStr(USwords, "$" & LCase(myWrd) & CR) = 0 Then \_

USwords = USwords & "$" & LCase(myWrd) & CR

End If

End If

StatusBar = "Spellchecking. To go: " & Trim(Str(Int((i / iStart) \_

\* 100))) & "% UK: " & UKcount & \_

" US: " & UScount

Debug.Print "Spellchecking. To go: " & Trim(Str(Int((i / iStart) \_

\* 100))) & "% UK: " & UKcount & \_

" US: " & UScount

End If

End If

i = i - 1

If i Mod 1000 = 0 Then

StatusBar = "Spellchecking. To go: " & \_

Trim(Str(Int((i / iStart) \* 100))) & \_

"% UK: " & \_

UKcount & " US: " & UScount

DoEvents

End If

Next wd

Application.ScreenUpdating = True

Selection.HomeKey Unit:=wdStory

Set myDoc = ActiveDocument

Documents.Add

Set myResults = ActiveDocument

Selection.TypeText Text:=UKwords & CR & USwords

Set rng = ActiveDocument.Content

rng.Sort

rng.InsertAfter Text:=CR

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[^13]{1,}\!"

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = ""

.MatchWildcards = True

.Execute Replace:=wdReplaceOne

DoEvents

.Text = "$"

.Replacement.Text = "^p^p"

.MatchWildcards = False

.Execute Replace:=wdReplaceOne

Set rng2 = ActiveDocument.Content

lenOne = Len(rng2)

.Text = "$"

.Replacement.Text = ""

.Execute Replace:=wdReplaceAll

lenTwo = Len(rng2)

.Text = "!"

.Execute Replace:=wdReplaceAll

lenThree = Len(rng2)

End With

Selection.TypeText Text:=CR & "US: " & Trim(Str(lenOne - lenTwo + 1)) \_

& " (" & UScount & ")" & CR

Selection.HomeKey Unit:=wdStory

Selection.TypeText Text:="UK: " & Trim(Str(lenTwo - lenThree + 1)) \_

& " (" & UKcount & ")" & CR

Set rng = myDoc.Content

totLen = rng.Characters.Count

Selection.HomeKey Unit:=wdStory

For Each pr In myResults.Paragraphs

testWord = Trim(pr.Range.Words(1))

If InStr("Distinct UK US", testWord) = 0 \_

And Len(testWord) > 2 Then

StatusBar = "Counting the occurences: " & testWord

Debug.Print "Counting the occurences: " & testWord

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = testWord

.Wrap = wdFindContinue

.Forward = True

.Replacement.Text = "^&!"

.MatchCase = showCase

.MatchWholeWord = True

.Execute Replace:=wdReplaceAll

DoEvents

End With

lenNow = rng.Characters.Count

numWords = lenNow - totLen

If numWords > 0 Then

myDoc.Activate

WordBasic.EditUndo

Set rng2 = pr.Range.Duplicate

rng2.MoveEnd , -1

rng2.InsertAfter Text:=" . . . " & Trim(Str(numWords))

myResults.Activate

End If

End If

Next pr

myDoc.Activate

Selection.HomeKey Unit:=wdStory

myResults.Activate

numParas = myResults.Paragraphs.Count

myResults.Paragraphs(1).Range.Font.Bold = True

myResults.Paragraphs(numParas - 2).Range.Font.Bold = True

Beep

myTime = Timer

Do

Loop Until Timer > myTime + 0.2

Beep

Exit Sub

ReportIt:

Application.ScreenUpdating = True

On Error GoTo 0

Resume

End Sub

**Sub CopyToListAlphaMenu()**

**'** Paul Beverley - Version 08.12.24

' Copies selected text into various alphabetic lists

' Filename contains (case INsensitive)

' keyWord = "list"

keyWord = "sheet"

Dim myList(10) As String

beepIfAlreadyListed = True

myStyle = "Heading 3"

' Heading words are case SENSitive

numLists = 3

myList(1) = "9 Word list"

myList(2) = "6 Places"

myList(3) = "3 People"

' copyWholePara = True

copyWholePara = False

includeFormatting = False

' includeFormatting = True

wordsToAvoid = "switch"

' wordsToAvoid = "FRedit,switch"

goBackToSource = True

' User inputs a character: myChar

myPrompt = ""

For i = 1 To numLists

myPrompt = myPrompt & myList(i) & vbCr

Next i

myChar = InputBox(myPrompt, "CopyToListAlphaMenu")

If myChar = vbCr Or myChar = "" Then Beep: Exit Sub

Do

followHeading = ""

For i = 1 To numLists

If Left(myList(i), 1) = myChar Then

followHeading = Mid(myList(i), 3)

Exit For

DoEvents

End If

Next i

DoEvents

Loop Until followHeading > ""

Dim sourceText As Range, tgt As Range

Set thisDoc = ActiveDocument

wds = Split("," & LCase(wordsToAvoid), ",")

If Selection.start = Selection.End Then

If LCase(Selection) = UCase(Selection) Then Selection.MoveLeft , 1

If copyWholePara = True Then

Selection.Expand wdParagraph

Else

Set rng = Selection.Range.Duplicate

rng.Expand wdWord

rng.MoveEnd wdWord, 1

chkWd = rng.Words.Last

If chkWd = "-" Then

rng.MoveEnd wdWord, 2

chkWd = rng.Words.Last

If chkWd = "-" Then

rng.MoveEnd wdWord, 1

Else

rng.MoveEnd wdWord, -1

End If

Else

rng.MoveEnd wdWord, -1

End If

DoEvents

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

End If

rng.Select

Else

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseEnd

rng.MoveEnd , -1

rng.Expand wdWord

Do While InStr(ChrW(8217) & "' ", Right(rng.Text, 1)) > 0

rng.MoveEnd , -1

DoEvents

Loop

Selection.Collapse wdCollapseStart

Selection.Expand wdWord

Selection.Collapse wdCollapseStart

rng.start = Selection.start

rng.Select

End If

Set sourceText = Selection.Range.Duplicate

myText = sourceText.Text

numTracks = Selection.Range.Revisions.Count

sourceText.Copy

Selection.Collapse wdCollapseEnd

gottaList = False

For Each myListDoc In Application.Documents

thisName = myListDoc.Name

nm = LCase(thisName)

gottaList = False

If InStr(nm, LCase(keyWord)) > 0 Then gottaList = True

For i = 1 To UBound(wds)

If InStr(nm, wds(i)) > 0 Then gottaList = False

DoEvents

Next i

If gottaList = True Then Exit For

DoEvents

Next myListDoc

If gottaList = False Then

Beep

MsgBox "Can't find a list."

Exit Sub

End If

' Decide where to put the item

Set rng = myListDoc.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = followHeading

.Style = myStyle

.Wrap = wdFindStop

.Forward = True

.Replacement.Text = ""

.MatchCase = True

.MatchWildcards = False

.Execute

DoEvents

End With

rng.Expand wdParagraph

listStart = rng.End

If rng.Find.Found = False Then

Beep

MsgBox "Can't find heading: """ & followHeading & """"

rng.start = 0

rng.End = 0

listStart = 0

Selection.HomeKey Unit:=wdStory

End If

rng.Collapse wdCollapseStart

' Check if it's already in the list

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^p" & myText & "^p"

.Wrap = wdFindStop

.Execute

DoEvents

End With

If rng.Find.Found = True Then

If goBackToSource = False Then

rng.Select

rng.MoveStart , 1

rng.MoveEnd , -1

End If

If beepIfAlreadyListed = True Then Beep

Exit Sub

End If

rng.start = listStart

rng.End = myListDoc.Content.End

If LCase(rng) <> UCase(rng) Then

For i = 1 To rng.Paragraphs.Count

myNextItem = Replace(rng.Paragraphs(i).Range.Text, vbCr, "")

If LCase(myNextItem) > LCase(myText) Or \_

LCase(myNextItem) = UCase(myNextItem) Then

Set rng = rng.Paragraphs(i).Range.Duplicate

Exit For

DoEvents

End If

DoEvents

Next i

End If

rng.Collapse wdCollapseStart

' Paste it in

If includeFormatting = True Then

rng.FormattedText = sourceText.FormattedText

If InStr(myText, vbCr) = 0 Then

rng.InsertAfter vbCr

End If

Else

rng.InsertAfter myText

If InStr(myText, vbCr) = 0 Then rng.InsertAfter vbCr

End If

rng.Revisions.AcceptAll

If goBackToSource = False Then myListDoc.Activate

End Sub

**Sub FigNoteCalloutSwap()**

**'** Paul Beverley - Version 13.11.24

' Moves the note callout to after the figure citation

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "^2 ("

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

myCount = 0

Do While rng.Find.Found = True

rng.MoveStart , 1

rng.MoveEnd wdWord, 7

parPosn = InStr(rng, ").")

If Left(rng, 5) = " (fig" And parPosn > 0 Then

rng.End = rng.start + parPosn + 1

rng.Select

rng.Cut

rng.MoveStart , -1

rng.Collapse wdCollapseStart

rng.Paste

myCount = myCount + 1

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

MsgBox "Changed: " & myCount

End Sub

**Sub MacquariePremiumLogin()**

**'** Paul Beverley - Version 30.11.24

' Logs in to the Macquarie dictionary

myWait = 1

myWaitPW = 0

mySite = "https://app.macquariedictionary.com.au/login"

uName = "my username"

pWord = "my password"

numTabs = 2

ActiveDocument.FollowHyperlink Address:=mySite

' Wait a bit to let the site settle

myTime = Timer

Do

Loop Until Timer > myTime + myWait

For i = 1 To numTabs

SendKeys "{TAB}"

Next i

SendKeys uName

SendKeys "{TAB}"

' Wait another bit, just in case

myTime = Timer

Do

Loop Until Timer > myTime + myWaitPW

SendKeys pWord

SendKeys "{ENTER}"

SendKeys "{NUMLOCK}"

End Sub

**Sub AcronymAddThe()**

**'** Paul Beverley - Version 11.12.24

' Adds 'the' in front of various acronyms or titles

myWords = "BBC ABC SPQR CIEP"

' Text, footnotes, endnotes?

myDo = "TFE"

If ActiveDocument.Footnotes.Count = 0 Then myDo = Replace(myDo, "F", "")

If ActiveDocument.Endnotes.Count = 0 Then myDo = Replace(myDo, "E", "")

myWords = " " & myWords & " "

myWords = Replace(myWords, " ", " ")

acronym = Split(Trim(myWords), " ")

For myStory = 1 To Len(myDo)

doIt = Mid(myDo, myStory, 1)

For i = 0 To UBound(acronym)

Select Case doIt

Case "T": Set rng = ActiveDocument.Content

Case "F": Set rng = ActiveDocument.StoryRanges(wdFootnotesStory)

Case "E": Set rng = ActiveDocument.StoryRanges(wdEndnotesStory)

End Select

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = "<" & acronym(i) & ">"

.Wrap = wdFindContinue

.Replacement.Text = ""

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.Execute

End With

Do While rng.Find.Found = True

rngNow = rng.start

rng.MoveStart , -4

rngTest = rng.start

Debug.Print rngTest, rngNow

myWord = LCase(Left(rng, 3))

ch3 = Mid(rng, 3, 1)

ch4 = Mid(rng, 4, 1)

rng.MoveStart , 4

If rngTest = 0 Then rng.start = 2

If rngNow = 0 Then rng.start = 0

If myWord <> "the" Then

myCount = myCount + 1

If ch4 = vbCr Or rngTest = 0 Or \_

InStr("!?.", ch3) > 0 Then

myArticle = "The "

Else

myArticle = "the "

End If

rng.InsertBefore Text:=myArticle

If myCount Mod 5 = 0 Then rng.Select

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Next i

Next myStory

Beep

MsgBox "Added 'the' to " & Trim(Str(myCount)) & " acronyms."

End Sub

**Sub CorrectDialogueEndGlobal()**

**'** Paul Beverley - Version 29.11.23

' Correct punctuation & capitalisation at dialogue end

' (from a macro by Philip Ridgers)

addHighlight = True

myColour = wdYellow

myVerbs = "said,replied,asked,shouted,stated,surmised,opined,"

myVerbs = myVerbs & "whispered,requested,"

myPNs = "She,He,They,It,"

myNames = "Jane,Harry,Ian,Helen"

myNames = myNames & ",Brian,Brown,Etc"

showAsAllChanged = False

selectOnly = (Selection.Range.Words.Count > 1)

If selectOnly Then

Set rng = Selection.Range.Duplicate

Else

Beep

myResponse = MsgBox("Work on the whole document?!", \_

vbQuestion + vbYesNoCancel, "CorrectDialogueEndGlobal")

If myResponse <> vbYes Then Exit Sub

Set rng = ActiveDocument.Content

End If

oldColour = Options.DefaultHighlightColorIndex

Options.DefaultHighlightColorIndex = myColour

myNames = "," & myNames & ","

myPNs = Replace(myPNs, ",", " ,")

myEnd = rng.End

With rng.Find

.MatchWildcards = True

.ClearFormatting

.Replacement.ClearFormatting

.Text = "[\!\?.]['""" & ChrW(8217) & ChrW(8221) & "] [A-Za-z\-]{1,} [a-z\-]@>"

.Forward = True

.Wrap = wdFindStop

.Execute

End With

Do Until rng.Find.Found = False Or rng.End > myEnd

If InStr(myVerbs, Trim(rng.Words.Last)) > 0 Then

lwStart = InStr(rng, " " & rng.Words.Last)

rng.End = rng.start + lwStart

thisWord = rng.Words.Last

If InStr(myPNs, thisWord) > 0 Then

If rng.Characters.First = "." Then rng.Characters.First = ","

rng.Characters(4) = LCase(rng.Characters(4))

If showAsAllChanged = True Then

rng.Cut

rng.Paste

End If

End If

If InStr(myNames, "," & Trim(thisWord) & ",") > 0 Then

If rng.Characters.First = "." Then rng.Characters.First = ","

End If

If addHighlight = True Then rng.HighlightColorIndex = myColour

n = n + 1

If n Mod 10 = 1 Then rng.Select

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Options.DefaultHighlightColorIndex = oldColour

Beep

Selection.EndKey Unit:=wdStory

MsgBox "All done!"

End Sub

**Sub CompareTwoLists()**

**'** Paul Beverley - Version 19.12.24

' Compares two lists, and highlights and lists the items unique to each

caseSensitive = True

' caseSensitive = False

myColour = wdBrightGreen

Set docA = ActiveDocument

nameA = Replace(docA.Name, ".docx", "")

CR = vbCr: CR2 = CR & CR

MsgBox "Please click in the file to be compared with > " & nameA & " <", vbInformation

Do

newName = Replace(ActiveDocument.Name, ".docx", "")

DoEvents

Loop Until newName <> nameA

nameB = newName

myResponse = MsgBox("Compare > " & nameA & " < with > " & nameB & " < ?", \_

vbQuestion + vbYesNoCancel, "CompareTwoLists")

If myResponse <> vbYes Then Beep: Exit Sub

Set docB = Documents(nameB)

If caseSensitive = True Then

wrdsA = CR & docA.Content.Text & CR

wrdsB = CR & docB.Content.Text & CR

Else

wrdsA = CR & LCase(docA.Content.Text) & CR

wrdsB = CR & LCase(docB.Content.Text) & CR

End If

Set docC = Documents.Add

Selection.TypeText Text:="Unique words in " & nameA & CR2

For Each pa In docA.Paragraphs

If caseSensitive = True Then

wd = pa.Range.Text

Else

wd = LCase(pa.Range.Text)

End If

If InStr(wrdsB, wd) = 0 Then

Selection.TypeText Text:=Mid(wd, 1)

pa.Range.HighlightColorIndex = myColour

End If

Next pa

Set docD = Documents.Add

Selection.TypeText Text:="Unique words in " & nameB & CR2

For Each pa In docB.Paragraphs

If caseSensitive = True Then

wd = pa.Range.Text

Else

wd = LCase(pa.Range.Text)

End If

If InStr(wrdsA, wd) = 0 Then

Selection.TypeText Text:=Mid(wd, 1)

pa.Range.HighlightColorIndex = myColour

End If

Next pa

End Sub

**Sub QuickSort(arr As Variant, ByVal low As Long, ByVal high As Long)**

**'** Paul Beverley - Version 21.12.24

' Sorts an array of text at high speed

Dim i As Long

Dim j As Long

Dim pivot As String

Dim temp As String

i = low

j = high

pivot = arr((low + high) \ 2)

Do While i <= j

Do While arr(i) < pivot

i = i + 1

Loop

Do While arr(j) > pivot

j = j - 1

Loop

If i <= j Then

temp = arr(i)

arr(i) = arr(j)

arr(j) = temp

i = i + 1

j = j - 1

End If

Loop

If low < j Then QuickSort arr, low, j

If i < high Then QuickSort arr, i, high

End Sub

**Sub DuplicateSentenceCountFast()**

**'** Paul Beverley - Version 17.12.24

' Finds duplicate sentences and gives their frequency

minLen = 7

' Max number of words

Dim sentenceArray() As String

timeNow = Timer

' Loop through sentences and add those longer than minLen to the array

CR = vbCr

For i = 1 To ActiveDocument.sentences.count

mySentence = Replace(Trim(ActiveDocument.sentences(i).Text), CR, "")

If Right(mySentence, 1) = ChrW(8221) Then

mySentence = Left(mySentence, Len(mySentence) - 1)

End If

myCount = UBound(Split(mySentence, " ")) + 1

If myCount >= minLen Then

For n = 1 To Len(mySentence)

ch = Mid(mySentence, n, 1)

If UCase(ch) <> LCase(ch) Then Exit For

Next n

mySentence = Mid(mySentence, n)

If Left(mySentence, 1) = "(" Then mySentence = Mid(mySentence, 2)

If Right(mySentence, 1) = ")" Then mySentence = Left(mySentence, Len(mySentence) - 1)

sntCount = sntCount + 1

ReDim Preserve sentenceArray(1 To sntCount)

sentenceArray(sntCount) = mySentence

If sntCount Mod 10 = 0 Then

DoEvents

StatusBar = "Finding sentences: " & sntCount

Debug.Print "Finding sentences: " & sntCount

End If

End If

Next i

If sntCount = 0 Then

Beep

MsgBox "No sentences longer than min. words found.", vbInformation

Exit Sub

End If

' Sort the array alphabetically

Call QuickSort(sentenceArray, LBound(sentenceArray), UBound(sentenceArray))

Documents.Add

Set rng = ActiveDocument.Content

' Output the sorted sentences to the debug window

n = 1

For i = LBound(sentenceArray) To UBound(sentenceArray) - 1

If sentenceArray(i) <> sentenceArray(i + 1) Then

If n > 1 Then

rng.InsertAfter Text:=sentenceArray(i) & " [" \_

& Trim(Str(n)) & "]" & CR

n = 1

End If

Else

n = n + 1

End If

Next i

If n > 1 Then rng.InsertAfter Text:=sentenceArray(i) & " [" \_

& Trim(Str(n)) & "]" & CR

newTime = Timer

totTime = newTime - timeNow

MsgBox ((Int(10 \* totTime) / 10) & " Seconds" & vbCr & vbCr \_

& (Int(10 \* totTime / 60) / 10) & " minutes")

End Sub

**Sub FindNumbersInQuotesSingle()**

**'** Paul Beverley - Version 14.12.24

' Finds the next number (numeral) inside single quotes

Set rng = Selection.Range.Duplicate

rng.End = ActiveDocument.Content.End

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8216) & "\*[0-9]{1,}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True

rng.Select

If InStr(rng, ChrW(8217)) = 0 Then

rng.MoveEnd , -1

rng.Collapse wdCollapseEnd

rng.Select

Selection.MoveUp , 1

Selection.MoveDown , 1

Exit Sub

Else

apos = InStr(rng, ChrW(8217) & "s")

rng.MoveStart , apos

sapo = InStr(rng, "s" & ChrW(8217))

rng.MoveStart , sapo

If InStr(rng, ChrW(8217)) = 0 Then

rng.MoveEnd , -1

rng.Collapse wdCollapseEnd

rng.Select

Selection.MoveUp , 1

Selection.MoveDown , 1

Exit Sub

End If

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Beep

Selection.Collapse wdCollapseEnd

MsgBox "No more found"

End Sub

**Sub NumbersInSingleQuotesHighlight()**

**'** Paul Beverley - Version 16.12.24

' Finds numbers (numerals) inside single quotes and highlights them

myColour = wdYellow

Set rng = ActiveDocument.Content

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ChrW(8216) & "\*[0-9]{1,}"

.Wrap = wdFindStop

.Replacement.Text = ""

.Forward = True

.MatchWildcards = True

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True

rng.Select

If InStr(rng, ChrW(8217)) = 0 Then

rng.HighlightColorIndex = myColour

rng.Select

Else

apos = InStr(rng, ChrW(8217) & "s")

rng.MoveStart , apos

rng.Select

sapo = InStr(rng, "s" & ChrW(8217))

rng.MoveStart , sapo + 1

rng.Select

If InStr(rng, ChrW(8217)) = 0 Then

rng.HighlightColorIndex = myColour

rng.Select

End If

End If

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

Beep

End Sub

**Sub TextCollectThisColour()**

**'** Paul Beverley - Version 17.12.24

' Copies any text in the colour at the cursor into a new document

Set rng = Selection.Range.Duplicate

rng.Collapse wdCollapseStart

rng.MoveEnd , 1

myColour = rng.Font.color

If myColour = 0 Then

Beep

MsgBox "Place the cursor in the colour to be collected"

Exit Sub

End If

rng.start = 0

rng.Collapse wdCollapseStart

Documents.Add

With rng.Find

.ClearFormatting

.Replacement.ClearFormatting

.Text = ""

.Wrap = wdFindStop

.Font.color = myColour

.Replacement.Text = ""

.Forward = True

.MatchWildcards = False

.MatchWholeWord = False

.Execute

End With

Do While rng.Find.Found = True

myCount = myCount + 1

rng.Copy

Selection.Paste

Selection.Collapse wdCollapseEnd

Selection.TypeText Text:=vbCr

rng.Collapse wdCollapseEnd

rng.Find.Execute

DoEvents

Loop

End Sub