**Information and Download** [**https://msaccessgurus.com/VBA/WordAutomate.htm**](https://msaccessgurus.com/VBA/WordAutomate.htm)

Unblock files if you have Mark of the Web issues:

<https://msaccessgurus.com/MOTW_Unblock.htm>

# Word Automation Code

Show the Navigation Pane to quickly find what you're looking for in this document

Graphical user interface, diagram, application

Description automatically generated

## Very Simplified Object Model

Diagram

Description automatically generated

## Object Variables

These are the object variables used in these code examples:

Dim db as DAO.Database \_

, rs As DAO.Recordset \_

, oWord As Word.Application \_

, oDoc As Word.Document \_

, oTable As Word.Table \_

, oRange As Word.Range \_

, oRange2 As Word.Range

Set db = CurrentDb

Set rs = db.OpenRecordset("Table-or-Query-or-SQL")

Set oWord = CreateObject("Word.Application")

Set oDoc = oWord.Documents.Add

Set oRange = oDoc.Content

Set oRange = oDoc.Paragraphs(oDoc.Paragraphs.Count).Range

Set oRange = oDoc.Paragraphs(2).Range

Set oRange = oDoc.Range(Start:=oDoc.Paragraphs(1).Range.Start,   
End:=oDoc.Paragraphs(4).Range.End)

Set oRange = ActiveDocument.Range(Start:=0, End:=99)

Set oRange2 = oDoc.Range(oRange.Start, oRange.Start + 1)

Set oRange = oDoc.Range

Set oRange = oTable.Cell(iRow, iColumn).Range

Set oRange = oTable.Rows(1).Range

## Create and Save Document

Dim oWord As Word.Application

'Initialize Word

Set oWord = CreateObject("Word.Application")

' make Word visible

oWord.Visible = True

'make a new Word document

Set oDoc = oWord.Documents.Add

'save document as specified name

oDoc.SaveAs sPathFile

'get full path and filename if extension wasn't specified

sPathFile = oDoc.Path & "\" & oDoc.Name

'... do other stuff

'update saved document – prefer to explicitely do this

oDoc.Save

'close without saving

oDoc.Close SaveChanges:=False

'release Word document object

Set oDoc = Nothing

'Quit Word

oWord.Quit

'release Word application object

Set oWord = Nothing

### Get Desktop Path

sPath = Environ("USERPROFILE") & "\Desktop\

But maybe that doesn't work right for remote users? Here's another way:

Function GetDesktopPath( \_

Optional pbAddTrailBackslash As Boolean = False \_

) As String

'strive4peace

With CreateObject("WScript.Shell")

GetDesktopPath = .SpecialFolders("Desktop") \_

& IIf(pbAddTrailBackslash, "\", "")

End With

End Function

### Public Function MakeAPath

This can be useful to make a specified path.

Public Function MakeAPath( \_

psPath As String) As Boolean

'crystal (strive4peace) ...190204

'set up error handler

On Error GoTo Proc\_Err

'initialize return value to be False for not successful

MakeAPath = False

'if directory is already there, return True and exit

If Len(Dir(psPath, vbDirectory)) > 0 Then

MakeAPath = True

GoTo Proc\_Exit

End If

'dimension variables

Dim i As Integer \_

, iPos As Integer \_

, sPath As String

'add backslash to end of path if necessary

iPos = 1

If Right(psPath, 1) <> "\" Then psPath = psPath & "\"

'get position of first backslash

iPos = InStr(iPos, psPath, "\")

'loop through directories of path and make folders

Do While iPos > 0

sPath = Left(psPath, iPos)

If Len(Dir(sPath, vbDirectory)) = 0 Then

MkDir sPath

DoEvents

End If

'set start search position to be 1+ position of last backslash found

iPos = InStr(iPos + 1, psPath, "\")

Loop

'if folder exists, then return True and exit

If Len(Dir(psPath, vbDirectory)) > 0 Then

MakeAPath = True

GoTo Proc\_Exit

End If

'exit code

Proc\_Exit:

On Error Resume Next

Exit Function

'if there is an error, then resume with exit code

Proc\_Err:

Resume Proc\_Exit

End Function

## Styles

Styles enable you to apply multiple formatting attributes to characters and paragraphs. Styles can also be used to identify text to be included in a Table of Centents, Lists of Figures and Tables, and anywhere you can insert the contents of a field. If a style exists in multiple places, Word will first use styles defined in the document, then in custom templates, then in the Normal.dotm template.

When using styles, it's helpful to turn on the Styles Pane. Click the little launcher icon in the lower right corner of the Styles group on the Home Ribbon tab.

Figure . Launcher icon to open Styles pane

Graphical user interface, application, Word

Description automatically generated

### help: Style object

<https://docs.microsoft.com/en-us/office/vba/api/word.style>

### help: WdStyleType enumeration

<https://docs.microsoft.com/en-us/office/vba/api/word.wdstyletype>

When changing styles, it's a good idea to change for the *current document only*. Otherwise all other documents using the same template, even those that have already been created, will be affected.

### Redefine Built-in or Existing Style

With oDoc.Styles(“aStyleName”)

With .Font

.Name = psFontName

.Size = piFontSize

.Color = pnFontColor

.Italic = pbooFontItalic

.Bold = pbooFontBold

End With

With .ParagraphFormat

.KeepWithNext = pbooKeepWithNext

.KeepTogether = pbooKeepTogether

.SpaceBefore = piSpaceBefore

.SpaceAfter = piSpaceAfter

.LeftIndent = piLeftIndent

.PageBreakBefore = pbooPageBreakBefore

.LineSpacingRule = 0 ' wdLineSpaceSingle

.Alignment = piAlignParagraph

End With

End With

### Create New Style

oDoc.Styles.Add Name:=psStyle, Type:=1 '1=wdStyleTypeParagraph

### Call Word\_DefineStyle

Call function to define a new paragraph style. Return 0 if Style was modified, or 1 if new style was created.

Select Case Word\_DefineStyle(oDoc, "aNormal" \_

, "Normal", "aNormal" \_

, "Calibri", 10, 0 \_

, 0, 0 \_

, 0, -1 \_

, 8, 0, 0 \_

)

Case 0: iStyleModified = iStyleModified + 1

Case 1: iStyleAdded = iStyleAdded + 1

End Select

'aAnchor

Select Case Word\_DefineStyle(oDoc, "aAnchor" \_

, "aNormal", "aNormal" \_

, , 1, RGB(255, 255, 255) \_

, 0, 0 \_

, -1, -1 \_

, 0, 1, 0 \_

)

Case 0: iStyleModified = iStyleModified + 1

Case 1: iStyleAdded = iStyleAdded + 1

End Select

### Public Function Word\_DefineStyle

This creates a new paragraph style

Public Function Word\_DefineStyle( \_

oDoc As Object \_

, psStyle As String \_

, Optional psBaseStyle As String = "aNormal" \_

, Optional psNextParagraphStyle As String = "aNormal" \_

, Optional psFontName As String = "" \_

, Optional piFontSize As Integer = -99 \_

, Optional pnFontColor As Long = -99 \_

, Optional pbooFontItalic As Integer = -99 \_

, Optional pbooFontBold As Integer = -99 \_

, Optional pbooKeepWithNext As Integer = -99 \_

, Optional pbooKeepTogether As Integer = -99 \_

, Optional piSpaceBefore As Integer = -99 \_

, Optional piSpaceAfter As Integer = -99 \_

, Optional piLeftIndent As Integer = -99 \_

, Optional pbooPageBreakBefore As Integer = -99 \_

, Optional piAlignParagraph As Integer = -99 \_

) As Integer

' strive4peace 170520 ... 170809

On Error GoTo Proc\_Err

Word\_DefineStyle = 0 'assume style already there

Dim sStr As String

'if there is not already a aTblDescription tag, then make one

On Error Resume Next

sStr = oDoc.Styles(psStyle).NameLocal

If Err.Number > 0 Then

oDoc.Styles.Add Name:=psStyle, Type:=1 'wdStyleTypeParagraph '-- set Description? Type?

Word\_DefineStyle = 1 'style added

End If

On Error GoTo Proc\_Err

With oDoc.Styles(psStyle) 'new -- or modify -- style

.AutomaticallyUpdate = False 'Shauna: False = don't update global Normal template -- lots of problems

'https://msdn.microsoft.com/en-us/library/office/microsoft.office.interop.word.style\_members.aspx

'True if the style is automatically redefined based on the selection. False if Microsoft Word prompts for confirmation before redefining the style based on the selection.

If psBaseStyle <> "" Then .BaseStyle = psBaseStyle

If psNextParagraphStyle <> "" Then .NextParagraphStyle = psNextParagraphStyle

.NoSpaceBetweenParagraphsOfSameStyle = False

' .Type = 1 'wdStyleTypeParagraph

With .Font

If psFontName <> "" Then

.Name = psFontName

End If

If piFontSize > 0 Then

.Size = piFontSize

End If

If pnFontColor <> -99 Then

.Color = pnFontColor

End If

If pbooFontItalic <> -99 Then

.Italic = pbooFontItalic

End If

If pbooFontBold <> -99 Then

.Bold = pbooFontBold

End If

End With

With .ParagraphFormat

If pbooKeepWithNext <> -99 Then

.KeepWithNext = pbooKeepWithNext

End If

If pbooKeepTogether <> -99 Then

.KeepTogether = pbooKeepTogether

End If

If piSpaceBefore <> -99 Then

.SpaceBefore = piSpaceBefore

End If

If piSpaceAfter <> -99 Then

.SpaceAfter = piSpaceAfter

End If

If piLeftIndent <> -99 Then

.LeftIndent = piLeftIndent

End If

If pbooPageBreakBefore <> -99 Then

.PageBreakBefore = pbooPageBreakBefore

End If

.LineSpacingRule = 0 ' wdLineSpaceSingle

'2 = wdAlignParagraphRight

If piAlignParagraph <> -99 Then

.Alignment = piAlignParagraph

End If

End With 'ParagraphFormat

End With

Proc\_Exit:

On Error Resume Next

Exit Function

Proc\_Err:

MsgBox Err.Description, , \_

"ERROR " & Err.Number \_

& " Word\_DefineStyle"

Word\_DefineStyle = -1 'error

Resume Proc\_Exit

Resume

End Function

## Write Text

With oDoc.Content

.InsertAfter "some text "

.InsertParagraphAfter

End With 'oDoc.Content

### Assign Style to Paragraph

'assign a style to the last paragraph in the document.

oDoc.Paragraphs(oDoc.Paragraphs.Count).Style = oDoc.Styles("aStyleName")

## Ranges

A range identifies a portion or all of a document. For examples of setting ranges, goto [SetRange](#SetRange)

#### Help for Range

##### Range object

<https://docs.microsoft.com/en-us/office/vba/api/word.range>

##### Range.Start property

<https://docs.microsoft.com/en-us/office/vba/api/word.range.start>

##### Range.End property

<https://docs.microsoft.com/en-us/office/vba/api/word.range.end>

## Find/Replace

in the ActiveDocument:

* Find 0 and replace with ChrW(9744) 'Checkbox open ☐
* Find -1 and replace with ChrW(9745) 'Checkbox checked ☑

Sub ReplaceWithCheckboxes()

With ActiveDocument.Range.Find

.Text = "0"

.Replacement.Text = ChrW(9744)

.MatchWholeWord = True

.Execute Replace:=2 'wdReplaceAll

.Text = "-1"

.Replacement.Text = ChrW(9745)

.MatchWholeWord = True

.Execute Replace:=2 'wdReplaceAll

End With

MsgBox "done"

End Sub

## Find/Replace, double paragraphs

Sub ReplaceDoubleParagraphs()

With ActiveDocument.Range.Find

.Text = "^w^p"

.Replacement.Text = "^p"

.MatchWholeWord = False

.Execute Replace:=2 'wdReplaceAll

.Text = "^p^p"

.Replacement.Text = "^p"

.MatchWholeWord = True

.Execute Replace:=2 'wdReplaceAll

End With

MsgBox "done"

End Sub

## Find/Replace, Format Text

Format parts of text in each cell of a table

Public Sub Word\_Table\_FormatCharacter

'early binding: oTable As Word.Table

Sub Word\_Table\_FormatCharacter(oTable As Object \_

, psCharacterStyle As String \_

, Optional psDeli1 As String = "(" \_

, Optional psDeli2 As String = "(" \_

)

'strive4peace

'format anything in parentheses as specified character style

'future -- make delimiters use parameters psDeli1 and psDeli2

With oTable.Range.Find

.Replacement.Style = psCharacterStyle

'wdReplaceAll = 2

'FindText is \(\*\) \means literally what's next

' search for literal (, then anything, then literal )

' then it's enclosed in parentheses to reference \1

'ReplaceWith:="\1" is everything delimited by the first and only set of parentheses in FindText

.Execute \_

Replace:=2 \_

, FindText:="(\(\*\))" \_

, ReplaceWith:="\1" \_

, MatchWildcards:=True

.Replacement.ClearFormatting

End With

End Sub

#### Help for Find

##### Find object (Word)

<https://docs.microsoft.com/en-us/office/vba/api/word.find>

##### Find.Execute method

<https://docs.microsoft.com/en-us/office/vba/api/word.find.execute>

##### Find.MatchWildcards property

<https://docs.microsoft.com/en-us/office/vba/api/word.find.matchwildcards>

## Move Around

| WdUnits | |
| --- | --- |
| 1 | wdCharacter |
| 2 | wdWord |
| 3 | wdSentence |
| 4 | wdParagraph |
| 5 | wdLine |
| 6 | wdStory |
| 7 | wdScreen |
| 8 | wdSection |
| 9 | wdColumn |
| 10 | wdRow |
| 11 | wdWindow |
| 12 | wdCell |
| 15 | wdTable |

### Range or Selection object

'Move end of selection/range back 1 character

oRange.MoveEnd Unit:=1, Count:=-1 '1=wdCharacter

'Move end forward 1 paragraph

oRange.MoveEnd Unit:=4, Count:=1 '4=wdParagraph

'Move start back 1 word

oRange.MoveStart Unit:=2, Count:=-1 '2=wdWord

'Move end until letter "A" is found and then Extend selection

With Selection

.MoveEndUntil Cset:="A", Count:= 1073741823 '1073741823= wdForward

.MoveRight Unit:=1, Count:=1, Extend:=1 '1=wdCharacter, 1=wdExtend

End With

### End of Document

Set oRange = oDoc.Range

oRange.Collapse (0) '0=wdCollapseEnd

### Beginning of Document

Set oRange = oDoc.Range

oRange.Collapse (1) '0=wdCollapseStart

### End of Row

Set oRange = .Cell(nRow, iCols).Range

'prepare for next after this

oRange.Collapse 0

oRange.Select

## Tables

### Insert Table

'goto end of document

Set oRange = oDoc.Range

oRange.Collapse (0) '0=wdCollapseEnd

oRange.Select

With oDoc

Set oTable = .Tables.Add( \_

Range:=oRange \_

, NumRows:=nRows \_

, NumColumns:=iCols \_

)

End With

### Add Caption to Table

'add a caption to a table

oWord.Selection.InsertCaption Label:="MyTableCaption" \_

, title:=sCaption \_

, Position:=0 \_

, ExcludeLabel:=0

### Write Text in cell of table

.Cell(1, 1).Range.Text = "#" & Chr(160) 'no-break space

.Cell(1, 1).Range.ParagraphFormat.Style = oDoc.Styles("aMyStyle")

### Call WordMakeTable

Set oTable = WordMakeTable(oDoc, oRange, nRows, iCols2 \_

, sTemp, "#", "This -> Foreign Field", "Foreign Table", "Info")

With oTable

'----------------------- Column widths

.Columns(1).PreferredWidth = CInt(0.35 \* 72) '#

.Columns(2).PreferredWidth = CInt(4 \* 72) 'Fields

.Columns(3).PreferredWidth = CInt(1.65 \* 72) 'Foreign Table

.Columns(4).PreferredWidth = CInt(1.25 \* 72) 'Attributes

End oTable

### Public Function WordMakeTable

Public Function WordMakeTable(oDoc As Object \_

, oRange As Object \_

, ByVal pnRows As Long \_

, ByVal pnCols As Long \_

, Optional ByVal psCaption As String = "" \_

, Optional ByVal psHead1 As String = "" \_

, Optional ByVal psHead2 As String = "" \_

, Optional ByVal psHead3 As String = "" \_

, Optional ByVal psHead4 As String = "" \_

) As Object 'As Word.Table

'strive4peace 170811, 20202

Dim oTable As Object 'late binding

' Dim oTable As Word.Table 'early binding

'insert table

With oDoc

Set oTable = .Tables.Add( \_

Range:=oRange \_

, NumRows:=pnRows \_

, NumColumns:=pnCols \_

)

End With

If (psCaption <> "") Then

'insert caption

oDoc.Application.Selection.InsertCaption Label:="Table" \_

, title:=psCaption \_

, Position:=0 \_

, ExcludeLabel:=0

End If

With oTable

'Position - wdCaptionPositionAbove=0

' .ApplyStyleHeadingRows = True

.TopPadding = 0

.BottomPadding = 0

.LeftPadding = 2 'points

.RightPadding = 2

.Spacing = 0 'Auto

.AllowPageBreaks = True

.AllowAutoFit = False

'mark heading row

.Rows(1).HeadingFormat = True

'dont allow rows to break

.Rows.AllowBreakAcrossPages = False

'no space above text between paragraphs

.Range.Paragraphs.SpaceBefore = 0

'Vertical Alignment is Center

.Range.Cells.VerticalAlignment = 1 ' 1=wdCellAlignVerticalCenter

'Heading Row

.Cell(1, 1).Range.Text = psHead1 ' rs1!Head1

.Cell(1, 1).Range.ParagraphFormat.Style = "aColumnOne"

.Cell(1, 2).Range.Text = psHead2 ' rs1!Head2 '"Index Name"

.Cell(1, 2).Range.ParagraphFormat.Style = "aColumnRest"

If (psHead3 <> "") Then

.Cell(1, 3).Range.Text = psHead3 ' "Fields"

.Cell(1, 3).Range.ParagraphFormat.Style = "aColumnRest"

End If

If (psHead4 <> "") Then

.Cell(1, 4).Range.Text = psHead4 ' rs1!Head4 '"Attributes"

.Cell(1, 4).Range.ParagraphFormat.Style = "aColumnRest"

End If

End With

Call DoTableBorders(oTable)

Set WordMakeTable = oTable

End Function

### Table Borders

### Public Sub DoTableBorders

Public Sub DoTableBorders(oTable As Object) 'Word.Table

's4p 170811

Dim i As Integer

With oTable

For i = 1 To 6

'wdBorderTop =-1

'wdBorderLeft = -2

'wdBorderBottom =-3

'wdBorderRight= -4

'wdBorderHorizontal = -5

'wdBorderVertical = -6

With .Borders(-i)

.LineStyle = 1 'wdLineStyleSingle=1

.LineWidth = 8 'wdLineWidth100pt=8. wdLineWidth150pt=12

.Color = RGB(200, 200, 200) 'medium-light gray

End With

Next i

End With

'change borders to black for first row

With oTable.Rows(1)

For i = 1 To 4

With .Borders(-i)

.Color = 0 'wdColorBlack = 0

End With

Next i

'Shading for header row

.Shading.BackgroundPatternColor = RGB(232, 232, 232)

End With 'first row

'Not used:

' 'wdLineStyleNone = 0

' .Borders(-7).LineStyle = 0 'wdBorderDiagonalDown =-7

' .Borders(-8).LineStyle = 0 'wdBorderDiagonalUp =-8

End Sub

## Sort

Sort Selected paragraphs

Selection.Sort

sort a table with a header row by column 1 and then column 3

.Sort ExcludeHeader:=True \_

, FieldNumber:=1 \_

, FieldNumber2:=3

## Bookmarks

set a bookmark to be any range in your document

Set oRange = oDoc.Paragraphs(oDoc.Paragraphs.Count).Range

oRange.Style = oDoc.Styles("aAnchor") 'aAnchor is a custom style

Set oRange2 = oDoc.Range(oRange.Start, oRange.Start + 1)

'bookmark

sBookmarkName = GetRemoveBadChars(sTablename)

oDoc.Bookmarks.Add sBookmarkName, oRange2

## Hyperlinks

insert a hyperlink to a bookmark in the current document

oDoc.Hyperlinks.Add Anchor:=oRange \_

, Address:="" \_

, SubAddress:=sBookmarkName

, ScreenTip:="" \_

, TextToDisplay:="Whatever text you want to appear to click on"

### Public Function GetRemoveBadChars

Public Function GetRemoveBadChars( \_

ByVal pString As String \_

) As String

'171005 crystal

Dim i As Integer \_

, sString As String \_

, sChar As String \* 1 \_

, iPos As Integer

'remove bad characters from name

'PARAMETERS

'pString is the string you want to correct

'EXAMPLE USEAGE

' on the AfterUpdate event of a control

' =GetRemoveBadChars([controlname])

'

'in a query:

'field --> GetRemoveBadChars: GetRemoveBadChars([strFieldname])

'EXAMPLE

' ? GetRemoveBadChars("as(,48209j@##@!")

' --> as48209j

GetRemoveBadChars = ""

If Len(Nz(pString)) < 1 Then Exit Function

pString = LTrim(Trim(pString))

For i = 1 To Len(pString)

sChar = Mid(pString, i, 1)

If Not InStr("`!@#$%^&\*()+=|\:;""'<>,.?/ ", sChar) > 0 Then

sString = sString & sChar

End If

Next i

GetRemoveBadChars = sString

End Function

## Images

### Insert image from file

Insert an image from file as an Inline image where the cursor is

Dim sPathFile As String

sPathFile = "c:\path\filename.png"

Selection.InlineShapes.AddPicture \_

FileName:=sPathFile \_

, LinkToFile:=False \_

, SaveWithDocument:=True

insert an inline picture at a specific location

'set range to end of document – or wherever you want, see [SetRange](#SetRange)

Set oRange = oDoc.Content

oRange.Collapse (0) '0=wdCollapseEnd

oRange.InlineShapes.AddPicture \_

FileName:=sPathFile \_

, LinkToFile:=False \_

, SaveWithDocument:=True

## Save Attachments to Files

Saving attachments to external files is a good way to go -- not just for transferring, but as a way to store. My research shows that only about 30% of embedded attachments are actually unique, unless they are pictures of a specific person or product ... so space will be saved and that will be more efficient. The database will also have better performance without being burdened by external files.

This code loops through an attachment field in a table and saves all the files externally.

Sub run\_SaveAttachmentsToFiles()

'130117 strive4peace

SaveAttachmentsToFiles "Props", "pScrShot", "PropID" \_

,, "propAtt", "propFile"

End Sub

Sub SaveAttachmentsToFiles( \_

ByVal sTableName As String \_

,ByVal sFieldName\_Att As String \_

,ByVal sFieldName\_ID As String \_

,Optional ByVal sPath As String = "" \_

,Optional ByVal sTableNameChild As String = "" \_

,Optional ByVal sFilenameField As String = "" \_

)

'130117 Crystal strive4peace

'PARAMETERS

' sTableName = name of table with attachment field

' sFieldName\_Att = name of attachment field

' sFieldName\_ID = name of PK

' sPath - optional. If not specified, written to c:\CurrentDbPath\Attachments

' WRITE CHILD RECORDS

' assumption: FK = sFieldName\_ID (same as parent table)

' sTableNameChild - optional. name of related table

' sFilenameField - fieldname in child table for Filename

On Error GoTo Proc\_Err

Dim db As DAO.Database \_

,rs As DAO.Recordset \_

,rs2 As DAO.Recordset2 \_

,fld2 As DAO.Field2

Dim sPathFile As String \_

,nNum As Long \_

,sSQL As String

nNum = 0

If sPath = "" Then

sPath = CurrentProject.Path & "\Attachments\"

If Dir(sPath,vbDirectory) = "" Then

MkDir sPath

DoEvents

End If

Else

If Right(sPath,1) <> "\" Then sPath = sPath & "\"

End If

Set db = CurrentDb

Set rs = db.OpenRecordset(sTableName,dbOpenDynaset)

Do While Not rs.EOF

Set rs2 = rs.Fields(sFieldName\_Att).Value

With rs2

Do While Not .EOF

sPathFile = sPath \_

& sTableName & "\_" \_

& Replace( \_

Replace(rs2.Fields( "FileName").Value \_

, ".jpg", "\_" & rs(sFieldName\_ID).Value & ".jpg") \_

, ".png",rs(sFieldName\_ID).Value & ".png")

If Dir(sPathFile) <> "" Then

' set attribute to Normal in case it is ReadOnly

' VBA.SetAttr sPathFile, vbNormal

Kill sPathFile

End If

Set fld2 = rs2.Fields( "FileData")

fld2.SaveToFile sPathFile

nNum = nNum + 1

If sTableNameChild <> "" And sFilenameField <> "" Then

'current database directory is stripped from path

'if path starts with \ then it is relative to database directory

sSQL = "INSERT INTO " & sTableNameChild \_

& "(" & sFieldName\_ID & ", " & sFilenameField & ")" \_

& " SELECT " & rs(sFieldName\_ID).Value \_

& ", """ & Replace(sPathFile,CurrentProject.Path, "") & """;"

With db

.Execute sSQL

If Not .RecordsAffected > 0 Then

If MsgBox( "Error creating Child Record for " \_

& sPathFile,vbOKCancel, "Error -- continue anyway") = vbCancel Then

GoTo Proc\_Exit

End If

End If

End With

End If

.MoveNext

Loop 'rs2

.Close

End With 'rs2

rs.MoveNext

Loop 'rs

MsgBox "Created " & nNum & " Files from Attachments" \_

,, "Done"

Proc\_Exit:

On Error Resume Next

'release object variables

If Not rs Is Nothing Then

rs.Close

Set rs = Nothing

End If

If Not rs2 Is Nothing Then

rs2.Close

Set rs2 = Nothing

End If

Set db = Nothing

Exit Sub

Proc\_Err:

MsgBox Err.Description,,\_

"ERROR " & Err.Number \_

& " SaveAttachmentsToFiles"

Resume Proc\_Exit

Resume

End Sub

## Page Setup

Page Setup has lots more, but these are the basics that I usually set.

' Page Setup

With oDoc.PageSetup

.TopMargin = CInt(0.5 \* 72) 'InchesToPoints

.BottomMargin = CInt(0.5 \* 72) 'InchesToPoints

.LeftMargin = CInt(0.6 \* 72) 'InchesToPoints

.RightMargin = CInt(0.5 \* 72) 'InchesToPoints

End With

## Table of Contents

Insert a Table of Contents (TOC) on the second paragraph of the document – replace what is there, if anything. Include up to 4 Heading levels or whatever is specified by LowerHeadingLevel.

Create the TOC after the document is written.

With oDoc

Set oRange = .Paragraphs(2).Range

.TablesOfContents.Add Range:=oRange \_

, RightAlignPageNumbers:=True \_

, UseHeadingStyles:=True \_

, UpperHeadingLevel:=1 \_

, LowerHeadingLevel:=4 \_

, IncludePageNumbers:=True \_

, AddedStyles:="" \_

, UseHyperlinks:=True \_

, HidePageNumbersInWeb:=True \_

, UseOutlineLevels:=False

.TablesOfContents(1).TabLeader = 1 'wdTabLeaderDots

.TablesOfContents.Format = 0 'wdIndexIndent

End With 'oDoc

## Lists

Create a Table of Figures at the end of the document

With oDoc

Set oRange =.Range

oRange.Collapse 0

.TablesOfFigures.Add \_

Range:=oRange \_

, Caption:="Table" \_

, IncludeLabel:=True \_

, RightAlignPageNumbers:=True \_

, UseHeadingStyles:=False \_

, UpperHeadingLevel:=1 \_

, LowerHeadingLevel:=3 \_

, IncludePageNumbers:=True \_

, AddedStyles:="" \_

, UseHyperlinks:=True \_

, HidePageNumbersInWeb:=True

.TablesOfFigures(1).TabLeader = 1 'wdTabLeaderDots

.TablesOfFigures.Format = 0 'wdIndexIndent

End With

## Headers

'---------------------------- header – NOT GOOD

'Macro recorder gets you something like this – not the best way

With oWord.ActiveWindow.ActivePane.View

If .Type = wdReadingView Then

.Type = wdPrintView

End If

'make current page header active

.SeekView = 9 'wdSeekCurrentPageHeader

'do stuff and then go back to the main document

'make main document active

.SeekView = 0 'wdSeekMainDocument

End With

### Better way to do a header than the macro recorder does it!

' Word.Document

Sub WriteHeader(oDoc As Object \_

, psTitle As String \_

, Optional pbHeadings As Boolean = False)

'220530 strive4peace

'tip from Doug Robbins so range doesn't have to be selected

Dim sgTabStop As Single

With oDoc.PageSetup

sgTabStop = .PageWidth - .LeftMargin - .RightMargin

End With

Dim oRange As As Object 'Word.Range

'1= wdHeaderFooterPrimary

Set oRange = oDoc.Sections(1).Headers(1).Range

With oDoc

If pbHeadings = True Then

'reference to Heading 1

' -1=wdFieldEmpty, False= Don't PreserveFormatting

.Fields.Add oRange, -1 \_

, "STYLEREF " & Chr(34) & "Heading 1" & Chr(34), False

Set oRange = .Sections(1).Headers(1).Range

'position cursor after field just added

oRange.Collapse 0 'wdCollapseEnd

' add comma space

oRange.InsertAfter ", "

'collapse to end

oRange.Collapse Direction:=0 'wdCollapseEnd

'reference to Heading 2

' -1=wdFieldEmpty

.Fields.Add oRange, -1 \_

, "STYLEREF " & Chr(34) & "Heading 2" & Chr(34), False

Set oRange = .Sections(1).Headers(1).Range

oRange.Collapse Direction:=0

End If

'add TABs and text to align on right

oRange.InsertAfter vbTab & psTitle & ", " \_

& "strive4peace, page "

oRange.Collapse Direction:=0

'reference to Page number

.Fields.Add oRange, -1, "Page", False

Set oRange = .Sections(1).Headers(1).Range

'collapse to end

oRange.Collapse Direction:=0

'insert /

oRange.InsertAfter "/"

oRange.Collapse 0

'reference to total pages

oRange.Parent.Fields.Add oRange, -1, "NumPages", False

Set oRange = .Sections(1).Headers(1).Range

oRange.Collapse 0

'update fields

.Sections(1).Headers(1).Range.Fields.Update

'border line below paragraph

With oRange

With .ParagraphFormat

'6 point space after paragraph

.SpaceAfter = 6

'clear default tab stops

.TabStops.ClearAll

'right tab stop at 6.5 inches

'wdAlignTabRight=2

'wdTabLeaderSpaces=0

.TabStops.Add Position:=sgTabStop \_

, Alignment:=2 \_

, Leader:=0

End With 'ParagraphFormat

With .Borders(-3) 'wdBorderBottom =-3

.LineStyle = 1 'wdLineStyleSingle=1

.LineWidth = 8 'wdLineWidth100pt=8

.Color = RGB(75, 75, 75) 'dark gray

End With 'Borders

End With

.Range.Collapse 1 'goto beginning of document

End With

End Sub

'---------------------------- end header

## Headers and Footers

Code for manipulating the various Headers and Footers in a document contributed by Doug Robbins, an Australian MVP for Word.

|  |  |
| --- | --- |
| wdHeaderFooterIndex | |
| 1 | wdHeaderFooterPrimary |
| 2 | wdHeaderFooterFirstPage |
| 3 | wdHeaderFooterEvenPages |

' Doug Robbins

Dim DocTarget As Document

Dim i As Long, j As Long

Set DocTarget = Documents.Add("path\name of template.dotx")

With DocTarget

    With .Sections(1)

        .PageSetup.OddAndEvenPagesHeaderFooter = True

        .PageSetup.DifferentFirstPageHeaderFooter = True

        With .Headers(wdHeaderFooterFirstPage).Range

            'Do something with the range of the header on the First Page

        End With

        With .Footers(wdHeaderFooterFirstPage).Range

            'Do something with the range of the footer on the First Page

        End With

        With .Headers(wdHeaderFooterPrimary).Range

            'Do something with the range of the header

            'that will appear on the Odd numbered pages

        End With

        With .Footers(wdHeaderFooterPrimary).Range

            'Do something with the range of the footer

            'that will appear on the Odd numbered pages

        End With

        With .Headers(wdHeaderFooterEvenPages).Range

            'Do something with the range of the header

            'that will appear on the even numbered pages

        End With

        With .Footers(wdHeaderFooterEvenPages).Range

            'Do something with the range of the footer

            'that will appear on the even numbered pages

        End With

    End With

    For i = 2 To .Sections.Count

        With .Sections(i)

            .PageSetup.OddAndEvenPagesHeaderFooter = True

            .PageSetup.DifferentFirstPageHeaderFooter = True

            ' by default, the headers and footers in a new Section

            ' added to a document (by the insertion of a Section Break)

            ' are linked to those in the previous Section

            ' so the code to set up that linking is perhaps redundant

            For j = 2 To .Headers.Count

                .Headers(j).LinkToPrevious = True

                .Footers(j).LinkToPrevious = True

            Next j

        End With

    Next i

End With

## Favorite Word Shortcut Keys

Some of my favorite shortcut keys when working with Word

|  |  |
| --- | --- |
|  | Cut |
|  | Copy |
|  | Paste |
|  | Expand AutoText (saved in template or BuildingBlocks.dotx) |
|  | Switch Case |
|  | Heading 1 |
|  | Heading 2 |
|  | Heading 3 |
|  | Bold |
|  | Italic |
|  | Underline |
|  | Yes, do it again (repeat formatting) |
|  | paste format again |
|  | remove formatting of selection |
|  | Goto last position |
|  | Go back from following hyperlink |
|  | Go to beginning |
|  | Go to end |
|  | Thesaurus |
|  | Spell Checker |
|  | Undo |

### Custom Shortcut keys for Styles

I like to define more shortcut keys for Styles – particularly Heading levels past 3. Steps:

1. Modify from down-arrow of style in Styles pane  
   Graphical user interface, application

   Description automatically generated  
   OR right-click Modify on style in ribbon  
   Graphical user interface, application

   Description automatically generated
2. from the Modify Style dialog box, drop choices under Format in lower left  
   Graphical user interface, text, application

   Description automatically generated
3. from Customize Keyboard, click in 'Press new shortcut key' and press what you want
4. Graphical user interface, text, application

   Description automatically generated
5. if you don't want shortcut to apply globally in Normal.dotm, then change where to save changes  
   Graphical user interface, text, application

   Description automatically generated
6. and then click Assign or press 

### additional Shortcut keys I like to define

|  |  |
| --- | --- |
|  | Heading 4 |
|  | Heading 5 |
|  | Heading 6 |

## About Crystal

Microsoft MVP for Access since 2006.

### Presentation coming up for me that is free to join

#### Access Pacific (hosted by George Hepworth) - Drawing an Access Report

*how to draw and write on an Access report without using controls. Example database is chemical elements.*

|  |  |
| --- | --- |
| *Periodic Table of the Elements* | Table  Description automatically generated |
| *Oxygen atom* | Diagram  Description automatically generated |

<https://accessusergroups.org/pacific/event/257-2022-06-02/>

Thursday June 2, 2022  
6:30 pm Pacific, 8:30 pm central

### Free Tools, Code, Videos, and Articles

On MsAccessGurus, you'll find lots of useful matter to help you become a better developer – videos, articles, VBA code, and tools with open source code so you can learn!

<https://MsAccessGurus.com>

### YouTube

My YouTube channel is <https://www.youtube.com/LearnAccessByCrystal>

### One-on-one Training

Do you have a project you need guidance with? We connect and I show you how to *do it yourself*. As we work together, I explain things, give you links for learning, and share code from the thousands of modules and procedures in my libraries.

I can teach you Access, Excel, Word, PowerPoint, Camtasia (or other video production tool – soon will be learning Microsoft ClipChamp), and more.

Special price for first one to take advantage of this offer: 4 anytime hours for 200 (1 hour minimum per session and also provide quick answers via email).

Also teach in virtual classrooms.

[training@MsAccessGurus.com](mailto:training@MsAccessGurus.com)

kind regards,   
crystal (*strive4peace*)