Scientia

May 02, 2021

Contents

CHAPTER 1

Introduction

This is a work in progress

Hello World

CHAPTER 2

Outlook VBA Scripts

2.1 Introduction

This VBA Module provides a feature to write a selection of emails to disk, such that the .msg file is named with date_sent – sender – subject and the visible attachments are saved alongside the file with the same date_sent as a prefix.

The .msg file and the associated attachments have their modified timestamp set to the sent timestamp.

The email in the Outlook Mailbox has the category "saved" added to it, so that messages which have already been saved can be kept track of.

Repository on GitHub: outlook-vba.

With thanks to Chip Pearson, www.cpearson.com, chip@cpearson.com for his file timestamp get/set VBA functions.

2.2 Source

```
Attribute VB_Name = "modEmailExport"
Option Explicit
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' modEmailExport
' By Simon Hoffe, https://github.com/simon-hoffe
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''


'---------------------------------------------------------------------
''' SaveSelectedEmails()
'''
''' Call this sub when one or more emails are selected in the navigator.
''' Each email will be saved to the folder "\Documents\Emails\" in the user's
''' home directory.
'''
```

(continues on next page)

(continued from previous page)

```
''' Each email will be saved to a file of the form:
'''    E yyyymmdd-hhmm - INITIAL@domain - Subject
'''    where
'''        yyyymmdd-hhmm is the time of sending, in the timezone of the user
'''        INITIAL@domain is a the Initials @ the domain of the sender
'''        Subject is the subject of the email. Repeating "RE" and "FWD" prefixes are reduced to one.
'''
'''    Then all visible attachments, and/or all non image attachments will also be saved
'''    to the \Documents\Emails\ folder in the form:
'''
'''        E yyymmdd-hhmm ___ filename.ext
'''        where
'''            yyyymmdd-hhmm is the same timestamp from the email.
'''            filename.ext is the original filename of the attachment.
'''
'''    There is some rudimentary checking for the existance of the folder, and
'''    for existing files with the same name. There is the option to overwrite the original .msg files.
'''    Attachments never overwrite and will be issued with unique names by appending a number "-1" or "-2" and so on.


Public Sub SaveSelectedEmails()
    Dim oMail As Outlook.MailItem
    Dim objItem As Object
    Dim enviro As String
    Dim sPath As String
    Dim nMsgResult As Long

    Dim oRegEx As Object
    Set oRegEx = CreateObject("vbscript.regexp")


    enviro = CStr(Environ("USERPROFILE"))
    sPath = enviro & "\Documents\Emails\"

    ' Prompt if the directory doesn't exist.
    If Not TestDirExist(sPath) Then
        nMsgResult = MsgBox( _
            Prompt:=sPath & " does not exist.", _
            Title:="Critical Error", _
            Buttons:=vbCritical + vbOKOnly)
        Exit Sub
    End If

    ' Iterate through all the selected emails
    For Each objItem In ActiveExplorer.Selection
        If objItem.MessageClass = "IPM.Note" Then
            Set oMail = objItem

            SaveOneMessageAsMsg oMail
        End If
    Next
End Sub

```

(continues on next page)

(continued from previous page)

```
'--------------------------------------------------------------------------
''' SaveOpenEmail()
'''
''' Call this sub from a window with the one email open and display.
''' Behaviour is as for SaveSelectedEmails() above.
'''
Public Sub SaveOpenEmail()
    Dim oMail As Outlook.MailItem
    Dim objItem As Object
    Dim enviro As String
    Dim sPath As String
    Dim nMsgResult As Long

    Dim oRegEx As Object
    Set oRegEx = CreateObject("vbscript.regexp")


    enviro = CStr(Environ("USERPROFILE"))
    sPath = enviro & "\Documents\Emails\"

    If Not TestDirExist(sPath) Then
        nMsgResult = MsgBox( _
            Prompt:=sPath & " does not exist.", _
            Title:="Critical Error", _
            Buttons:=vbCritical + vbOKOnly)
        Exit Sub
    End If

    Set objItem = ActiveInspector.CurrentItem

    If objItem.MessageClass = "IPM.Note" Then
        Set oMail = objItem

        SaveOneMessageAsMsg oMail
    End If
End Sub

'--------------------------------------------------------------------------
''' ShowSenders()
'''
''' Call from the navigator with one or more emails selected.
''' This is a development tool to display the information pulled from the
''' MailItem object for each of the emails.
''' It doesn't write anything

Public Sub ShowSenders()
    Dim oMail As Outlook.MailItem
    Dim objItem As Object
    Dim sPath As String
    Dim dtDate As Date
    Dim sName As String
    Dim enviro As String
    Dim sSender As String
    Dim sSubject As String
    Dim sDateTime As String

    enviro = CStr(Environ("USERPROFILE"))
```

(continues on next page)

(continued from previous page)

```
    For Each objItem In ActiveExplorer.Selection
        If objItem.MessageClass = "IPM.Note" Then
            Set oMail = objItem

            GetDateString oMail, sDateTime
            GetSenderString oMail, sSender
            GetSubjectString oMail, sSubject

            sName = "E " & sDateTime & " - " & sSender & " - " & sSubject & ".msg"

            sPath = enviro & "\Documents\Emails"

            MsgBox "Subject: " & oMail.Subject & vbCrLf & _
                "SentOn: " & oMail.SentOn & vbCrLf & _
                "SenderName: " & oMail.SenderName & vbCrLf & _
                "SenderAddress: " & oMail.SenderEmailAddress & vbCrLf & _
                "SenderEmailType: " & oMail.SenderEmailType & vbCrLf & _
                vbCrLf & _
                "Folder: " & sPath & vbCrLf & _
                "Filename: " & sName

        End If
    Next

End Sub

'---------------------------------------------------------------------------
''' SaveOneMessageAsMsg(oMail As Outlook.MailItem
'''
''' The workhorse called by SaveSelectedEmails() and SaveOpenEmail()

Private Sub SaveOneMessageAsMsg(ByRef oMail As Outlook.MailItem)
    Dim oAttachment As Outlook.Attachment
    Dim objItem As Object
    Dim sPath As String
    Dim dtDate As Date
    Dim sName As String
    Dim sAttName As String
    Dim sAttExt As String
    Dim sExt As String
    Dim enviro As String
    Dim sSender As String
    Dim sSubject As String
    Dim sDateTime As String
    Dim bSkipAttachment As Boolean
    Dim iSize As Long
    Dim iSizeLimit As Long
    Dim nMsgResult As Long
    Dim bSkipEmail As Boolean
    Dim vTemp As Variant
    Dim vPropNames() As Variant
    Dim bHiddenProp As Boolean
    Dim Result As Boolean


    Dim oRegEx As Object
    Set oRegEx = CreateObject("vbscript.regexp")
```

(continues on next page)

(continued from previous page)

```
183
184
185     enviro = CStr(Environ("USERPROFILE"))
186     sPath = enviro & "\Documents\Emails\"
187
188     If Not TestDirExist(sPath) Then
189         nMsgResult = MsgBox( _
190             Prompt:=sPath & " does not exist.", _
191             Title:="Critical Error", _
192             Buttons:=vbCritical + vbOKOnly)
193         Exit Sub
194     End If
195
196     GetDateString oMail, sDateTime
197     GetSenderString oMail, sSender
198     GetSubjectString oMail, sSubject
199
200     sName = "E " & sDateTime & " - " & sSender & " - " & sSubject
201
202
203     sExt = ".msg"
204
205 ' Some code for debugging
206
207 '            MsgBox "Subject: " & oMail.Subject & vbCrLf & _
208 '                "SentOn: " & oMail.SentOn & vbCrLf & _
209 '                "SenderName: " & oMail.SenderName & vbCrLf & _
210 '                "SenderAddress: " & oMail.SenderEmailAddress & vbCrLf & _
211 '                "SenderEmailType: " & oMail.SenderEmailType & vbCrLf & _
212 '                vbCrLf & _
213 '                "Folder: " & sPath & vbCrLf & _
214 '                "Filename: " & sName & sExt
215 '
216 '            Debug.Print sPath & sName
217
218     bSkipEmail = False
219     If TestFileExist(sPath & sName & sExt) Then
220         nMsgResult = MsgBox( _
221             Prompt:="This file already exists:" & vbCrLf & vbCrLf & _
222                     sName & sExt & vbCrLf & vbCrLf & _
223                     "Overwrite?", _
224             Title:="Warning: File Exists", _
225             Buttons:=vbQuestion + vbYesNoCancel)
226
227         If nMsgResult = vbCancel Then
228             Exit Sub
229         End If
230
231         If nMsgResult = vbYes Then
232             bSkipEmail = False
233         Else
234             bSkipEmail = True
235         End If
236     End If
237
238     If Not bSkipEmail Then
239         ' Save the email to file on disk
```

(continues on next page)

(continued from previous page)

```
        oMail.SaveAs sPath & sName & sExt, olMSG

        ' Set the Modified Timestamp of the file to the Sent Timestamp
        Result = SetFileDateTime(FileName:=sPath & sName & sExt, _
            FileDateTime:=oMail.SentOn, _
            WhichDateToChange:=FileDateLastModified, _
            NoGMTConvert:=False)

        If Result = False Then
            Debug.Print "An error occurred with SetFileDateTime."
        Else
            ' If all is well, add a category called "Saved" to the email in the
            ' Outlook Mailbox to flag that it's been saved
            AddCategoryToEmail oMail, "Saved"
            oMail.Save
        End If

        For Each oAttachment In oMail.Attachments
            bSkipAttachment = False
            oRegEx.Pattern = "^(.*?)(\.?[^.]*)$"
            sAttName = oRegEx.Replace(oAttachment.FileName, "$1")
            sAttExt = LCase(oRegEx.Replace(oAttachment.FileName, "$2"))
            iSize = oAttachment.Size


            ' Differentiate between attachments which are embedded inline, and
↪explicit attachments
            Const PR_ATTACHMENT_HIDDEN = "http://schemas.microsoft.com/mapi/proptag/
↪0x7FFE000B"

            vPropNames = Array(PR_ATTACHMENT_HIDDEN)
            vTemp = oAttachment.PropertyAccessor.GetProperties(vPropNames)

            bHiddenProp = Not IsError(vTemp(0))

            bSkipAttachment = False

            If bHiddenProp Then
                Select Case sAttExt
                    Case ""
                        bSkipAttachment = True
                    Case "."
                        bSkipAttachment = True
                    Case ".png"
                        bSkipAttachment = True
                    Case ".jpg"
                        bSkipAttachment = True
                    Case ".gif"
                        bSkipAttachment = True
                    Case Else
                        ' Don't skip hidden attachments which aren't in the above list
                        bSkipAttachment = False
                End Select
            End If

            ' Ignore files which look like: F4F90AC9@CEDD0C72.18bc1a5d
            ' regardless of whether tagged as hidden or not
```

(continues on next page)

(continued from previous page)

```
            oRegEx.Pattern = "^\.[0-9a-fA-F]{5,}$"
            If oRegEx.Test(sAttExt) Then
                bSkipAttachment = True
            End If

            ' Ignore files which have no extension
            ' regardless of whether tagged as hidden or not
            oRegEx.Pattern = "^\.{0,1}$"
            If oRegEx.Test(sAttExt) Then
                bSkipAttachment = True
            End If

            If Not bSkipAttachment Then
                sName = "E " & sDateTime & " ___ " & sAttName

                MakeFileNameUnique sPath, sName, sAttExt

                oAttachment.SaveAsFile sPath & sName & sAttExt

                Result = SetFileDateTime(FileName:=sPath & sName & sAttExt, _
                    FileDateTime:=oMail.SentOn, _
                    WhichDateToChange:=FileDateLastModified, _
                    NoGMTConvert:=False)

                If Result = False Then
                    Debug.Print "An error occurred with SetFileDateTime."
                End If
            End If
        Next
    End If
End Sub


'---------------------------------------------------------------------------
Private Sub ReplaceCharsForFileName(sName As String, _
    sChr As String _
)
    sName = Replace(sName, "'", sChr)
    sName = Replace(sName, "*", sChr)
    sName = Replace(sName, "/", sChr)
    sName = Replace(sName, "\", sChr)
    sName = Replace(sName, ":", sChr)
    sName = Replace(sName, "?", sChr)
    sName = Replace(sName, Chr(34), sChr)
    sName = Replace(sName, "<", sChr)
    sName = Replace(sName, ">", sChr)
    sName = Replace(sName, "|", sChr)
End Sub

'---------------------------------------------------------------------------
Private Sub GetDateString(oMail As Outlook.MailItem, _
sDateString As String _
)
    Dim dtDate As Date

    dtDate = oMail.SentOn
    sDateString = Format(dtDate, "yyyymmdd", vbUseSystemDayOfWeek, vbUseSystem) & _
```

(continues on next page)

(continued from previous page)

```
        Format(dtDate, "-hhnn", vbUseSystemDayOfWeek, vbUseSystem)
End Sub

'---------------------------------------------------------------------------
Private Sub AddCategoryToEmail(ByRef oMail As Outlook.MailItem, _
   sNewCategory As String _
)
    Dim sCategorySeparator As String
    Dim sCategories() As String
    Dim bNewCatExist As Boolean
    Dim sOne As Variant

    sCategorySeparator = RegKeyRead("HKEY_CURRENT_USER\Control Panel\International\sList")
    If Len(sCategorySeparator) <> 1 Then
        sCategorySeparator = ","
    End If

    sCategories = Split(oMail.Categories, sCategorySeparator)

    bNewCatExist = False
    For Each sOne In sCategories
        If StrComp(UCase(sOne), UCase(sNewCategory), vbTextCompare) = 0 Then
            bNewCatExist = True
            Exit For
        End If
    Next sOne

    If Not bNewCatExist Then
        ReDim Preserve sCategories(UBound(sCategories) + 1)
        sCategories(UBound(sCategories)) = sNewCategory

        oMail.Categories = Join(sCategories, sCategorySeparator)
    End If
End Sub

'---------------------------------------------------------------------------
Private Sub GetSenderString(oMail As Outlook.MailItem, _
sSenderString As String _
)
    Dim objRegEx As Object
    Dim oMatches As Object
    Dim oMatch As Object
    Dim sSenderName As String
    Dim sInitials As String
    Dim sSenderAddress As String
    Dim sDomain As String
    Dim sTemp As String

    Set objRegEx = CreateObject("vbscript.regexp")

'    MsgBox "Subject: " & oMail.Subject & vbCrLf & _
'        "SentOn: " & oMail.SentOn & vbCrLf & _
'        "SenderName: " & oMail.SenderName & vbCrLf & _
'        "SenderAddress: " & oMail.SenderEmailAddress & vbCrLf & _
'        "SenderEmailType: " & oMail.SenderEmailType

```

(continues on next page)

(continued from previous page)

```
    ' Process the Sender Name

    With objRegEx
        .Global = True
        .MultiLine = True
        .IgnoreCase = True
    End With

    sSenderName = oMail.SenderName
    sSenderAddress = oMail.SenderEmailAddress

    ' Remove any email address add on in the sender name
    objRegEx.Pattern = "\S+@[^\.]\S*\.\w{2,}"
    If objRegEx.Test(sSenderName) Then
        sTemp = objRegEx.Replace(sSenderName, "")
    End If

    ' Trim any leading and/or trailing white space
    objRegEx.Pattern = "^\s*(.*?)\s*$"
    sTemp = objRegEx.Replace(sTemp, "$1")

    If Len(sTemp) = 0 Then
        objRegEx.Pattern = "(\S+)@[^\.]\S*\.\w{2,}"
        sTemp = objRegEx.Replace(sSenderName, "$1")
    End If

    sSenderName = sTemp

    ' Remove anything between brackets
    objRegEx.Pattern = "(\([^()]*\)|\[[^\[\]]*\]|\{[^{}]*\}|<[^<>]*>)"
    If objRegEx.Test(sSenderName) Then
        sSenderName = objRegEx.Replace(sSenderName, "")
    End If

    ' If the Sender name is in "Surname, Name" format, then switch it around
    objRegEx.Pattern = "^([^,]+),([^,]+)$"
    If objRegEx.Test(sSenderName) Then
        sSenderName = objRegEx.Replace(sSenderName, "$2 $1")
    End If

    objRegEx.Pattern = "\b\w"
    Set oMatches = objRegEx.Execute(sSenderName)

    sInitials = ""
    For Each oMatch In oMatches
        sInitials = sInitials & UCase(oMatch.Value)
    Next

    sDomain = ""
    If oMail.SenderEmailType = "SMTP" Then
        objRegEx.Pattern = "@[^.]*"
        Set oMatches = objRegEx.Execute(sSenderAddress)
        For Each oMatch In oMatches
            sDomain = LCase(oMatch.Value)
            Exit For
        Next
    Else
```

(continues on next page)

(continued from previous page)

```
        sDomain = "@local"
    End If

    sSenderString = sInitials & sDomain
End Sub

'--------------------------------------------------------------------------
Private Sub GetSubjectString(oMail As Outlook.MailItem, _
sSubjectString As String _
)
    Dim objRegEx As Object
    Dim sName As String

    sName = oMail.Subject

    Set objRegEx = CreateObject("vbscript.regexp")

    With objRegEx
        .Global = True
        .MultiLine = True
        .IgnoreCase = True
    End With

    ' Remove duplicate "RE" strings
    objRegEx.Pattern = "(re\W+)(re\W+)+"
    sName = objRegEx.Replace(sName, "$1")

    ' Remove duplicate "FW" strings
    objRegEx.Pattern = "(fwd?\W+)(fwd?\W+)+"
    sName = objRegEx.Replace(sName, "$1")

    ReplaceCharsForFileName sName, " "

    ' Trim any leading and/or trailing white space
    objRegEx.Pattern = "^\s*(.*?)\.*\s*$"
    sName = objRegEx.Replace(sName, "$1")

    ' Trim any double spaces
    objRegEx.Pattern = "\s+"
    sName = objRegEx.Replace(sName, " ")

    ' Limit the length to 100 characters
    sName = Left(sName, 100)

    If Len(sName) = 0 Then
        sName = "No Subject"
    End If

    sSubjectString = sName
End Sub

'--------------------------------------------------------------------------
Private Sub MakeFileNameUnique(sPath As String, sFileName As String, sExt As String _
)
    Dim oRegEx As Object
    Dim sIntExt As String
    Dim sIntPath As String
```

(continues on next page)

(continued from previous page)

```
    Dim sIntFileName As String
    Dim sTestName As String
    Dim i As Integer

    Set oRegEx = CreateObject("vbscript.regexp")

    If Len(sExt) <> 0 Then
        oRegEx.Pattern = "^\.*"
        sIntExt = oRegEx.Replace(sExt, ".") ' Make sure the ext has a "."
    Else
        sIntExt = ""
    End If


    oRegEx.Pattern = "\\*$"
    sIntPath = oRegEx.Replace(sPath, "") ' Remove the trailing backslash, for now

    If Len(Dir(sIntPath, vbDirectory)) = 0 Then
        ' Bigger problems, the directory doesn't exist!
        Exit Sub
    End If

    sIntPath = sIntPath + "\"
    sIntFileName = sFileName
    sTestName = Dir(sIntPath & sIntFileName & sIntExt)

    i = 0
    Do While sTestName <> ""
        i = i + 1
        sIntFileName = sFileName & "-" & i
        sTestName = Dir(sIntPath & sIntFileName & sIntExt)
    Loop
    sFileName = sIntFileName
End Sub

'--------------------------------------------------------------------------
Private Function TestDirExist(sPath As String) As Boolean
    Dim sIntPath As String
    Dim oRegEx As Object
    Set oRegEx = CreateObject("vbscript.regexp")

    oRegEx.Pattern = "\\*$"
    sIntPath = oRegEx.Replace(sPath, "") ' Remove any trailing backslash
    If Len(Dir(sIntPath, vbDirectory)) = 0 Then
        TestDirExist = False
    Else
        TestDirExist = True
    End If
End Function

'--------------------------------------------------------------------------
Private Function TestFileExist(sPath As String) As Boolean
    If Len(Dir(sPath)) = 0 Then
        TestFileExist = False
    Else
        TestFileExist = True
    End If
```

(continues on next page)

(continued from previous page)

```
End Function


'--------------------------------------------------------------------------
Function BrowseForFolder(Optional OpenAt As Variant) As Variant
  Dim ShellApp As Object
  Set ShellApp = CreateObject("Shell.Application"). _
 BrowseForFolder(0, "Please choose a folder", 0, OpenAt)

 On Error Resume Next
    BrowseForFolder = ShellApp.self.Path
 On Error GoTo 0

 Set ShellApp = Nothing
    Select Case Mid(BrowseForFolder, 2, 1)
        Case Is = ":"
            If Left(BrowseForFolder, 1) = ":" Then GoTo Invalid
        Case Is = "\"
            If Not Left(BrowseForFolder, 1) = "\" Then GoTo Invalid
        Case Else
            GoTo Invalid
    End Select
 Exit Function

Invalid:
 BrowseForFolder = False
End Function

'reads the value for the registry key i_RegKey
'if the key cannot be found, the return value is ""
' https://vba-corner.livejournal.com/3054.html
Function RegKeyRead(i_RegKey As String) As String
Dim myWS As Object

  On Error Resume Next
  'access Windows scripting
  Set myWS = CreateObject("WScript.Shell")
  'read key from registry
  RegKeyRead = myWS.RegRead(i_RegKey)
End Function
```

CHAPTER 3

VIM Scripts

3.1 Introduction

Personal reference for my own VIM environment setup and scripts.

Repository on GitHub: vimfiles.

3.2 GVim RC

```
colorscheme darkblue
set guifont=Lucida_Console:h9:cANSI:qDRAFT
set hidden
set fileencodings=latin1,utf-8,ucs-bom
set encoding=latin1
set lines=40 columns=128
set number
set visualbell

command Edgvimrc e $HOME/Dropbox/_VIM/_gvimrc

source $HOME/Dropbox/_VIM/_gvim_simon
```

3.3 GVim Personal Setup

```
" vim: set syntax=vim:set encoding=latin1

source $HOME/SJH/scientia/vimfiles/_vim_common_simon

set listchars=eol:¶,tab:»,trail:·,nbsp:␣,extends:,precedes:
```

(continues on next page)

(continued from previous page)

```
set list
set showbreak=¦
```

CHAPTER 4

Metaquotes MT4 and MT5

4.1 Introduction

These pages are firstly "notes to self" on setting things up, and will reference private git repositories.

Perhaps, as this matures, I will clean it up for wider consumption.

4.2 MT4 Setup

- Install MT4 from the install file provided by the broker

  Modify the destination directory with a (n) suffix to install multiple copies on the same machine

- Git clone the MT4 base repo into a new folder on the machine, then manually copy it across to overlay onto the MQL4 folder in Terminal datadirectory.

```
git clone https://github.com/simon-hoffe/mql5-mt4-base.git

git clone git@github.com-personal:simon-hoffe/mql5-mt4-base.git
```

Then need to initialise the sub-modules and clone them as well.

```
git submodule update --init --recursive
```

- genindex

Index