



MemoryKnots.xlsm
www.github.com/alexofrhodes

--- Table Of Contents ---

(Document Module)	ThisWorkbook
(Document Module)	SETTINGS - Sheet3
(Document Module)	o NOTES - Sheet1
(Document Module)	o RESOLVED - Sheet2
(Document Module)	README - Sheet22
(Code Module)	modules
(Code Module)	UpdateAddin
(Code Module)	VoiceToWav
(Code Module)	StringToWav
(Code Module)	ZipProc
(Code Module)	JPG
(Code Module)	Home
(UserForm)	uMemoryKnots
(UserForm)	uDEV

--- ThisWorkbook ---

' Author	Anastasiou Alex
' Website	https://github.com/alexofrhodes

--- modules ---

```
Function OutlookCheck() As Boolean
    Dim xOLApp As Object
    ' On Error GoTo L1
    Set xOLApp = CreateObject("Outlook.Application")
    If Not xOLApp Is Nothing Then
        OutlookCheck = True
        ' MsgBox "Outlook " & xOL
        ' App.Version & " installed", vbExclamation
        Set xOLApp = Nothing
        Exit Function
    End If
    OutlookCheck = False
    'L1: MsgBox "Outlook not installed
    '", vbExclamation, "Kutools for Outlook"
End Function
```

```
Function Clipboard(Optional StoreText As String) As String
    'PURPOSE: Read/Write to Clipboard
    'Source: ExcelHero.com (Daniel Ferry)

    Dim X As Variant

    'Store as variant for 64-bit VBA support
    X = StoreText

    'Create HTMLFile Object
    With CreateObject("htmlfile")
        With .parentWindow.clipboardData
            Select Case True
            Case Len(StoreText)
                'Write to the clipboard
                .SetData "text", X
            Case Else
                'Read from the clipboard (no variable passed through)
                Clipboard = .GetData("text")
            End Select
        End With
    End With
End Function
```

```
Sub ListboxSortAZ(myListBox As MSForms.ListBox, Optional resetMacro As _
String)
    'Create variables
    Dim j As Long
    Dim i As Long
    Dim TEMP As Variant
    'Reset the listBox into standard order
    If resetMacro <> "" Then
        Run resetMacro, myListBox
    End If
End Sub
```



```

'Use Bubble sort method to put listBox in A-Z order
With myListBox
    For j = 0 To .ListCount - 2
        For i = 0 To .ListCount - 2
            If LCase(.List(i)) > LCase(.List(i + 1)) Then
                TEMP = .List(i)
                .List(i) = .List(i + 1)
                .List(i + 1) = TEMP
            End If
        Next i
    Next j
End With
End Sub

```

```

Sub ListboxSortZA(myListBox As MSForms.ListBox, Optional resetMacro As _
String)

```

```

'Create variables

```

```

Dim j As Long

```

```

Dim i As Long

```

```

Dim TEMP As Variant

```

```

'Reset the listBox into standard order

```

```

If resetMacro <> "" Then

```

```

    Run resetMacro, myListBox

```

```

End If

```

```

'Use Bubble sort method to put listBox in Z-A order

```

```

With myListBox

```

```

    For j = 0 To .ListCount - 2

```

```

        For i = 0 To .ListCount - 2

```

```

            If LCase(.List(i)) < LCase(.List(i + 1)) Then

```

```

                TEMP = .List(i)

```

```

                .List(i) = .List(i + 1)

```

```

                .List(i + 1) = TEMP

```

```

            End If

```

```

        Next i

```

```

    Next j

```

```

End With

```

```

End Sub

```

```

Function WorksheetExists(shtName As String, Optional wb As Workbook) As _
Boolean

```

```

    Dim sht As Worksheet

```

```

    If wb Is Nothing Then Set wb = ThisWorkbook

```

```

    On Error Resume Next

```

```

    Set sht = wb.Sheets(shtName)

```

```

    On Error GoTo 0

```

```

    WorksheetExists = Not sht Is Nothing

```

```

End Function

```

```

Function SelectionValues(link As String)

```

```

    Dim c As Range
    If TypeName(Selection) = "Range" _
    And Selection.Cells.count = 1 Then
        SelectionValues = Selection.Value
    ElseIf TypeName(Selection) = "Range" Then
        For Each c In Selection.SpecialCells(xlCellTypeVisible)
            If Len(c.Value) <> 0 Then
                If SelectionValues = "" Then
                    SelectionValues = c.Value
                Else
                    SelectionValues = SelectionValues & link & c.Value
                End If
            End If
        Next c
    End If
End Function

```

```

Function Listbox_Selected(LBox As MSForms.ListBox, Count_Indexes_Values _
As Integer)

```

```

    Dim SelectedIndexes As String
    Dim SelectedValues As String
    Dim SelectedCount As Integer
    Dim i As Long
    With LBox
        For i = 0 To .ListCount - 1
            If .Selected(i) Then
                'items count
                SelectedCount = SelectedCount + 1
                'items' indexes
                SelectedIndexes = SelectedIndexes & i & ","
                'items' values
                SelectedValues = SelectedValues & .List(i) & ","
            End If
        Next i
    End With

```

```

    If SelectedCount = 0 Then
        Listbox_Selected = 0
        Exit Function
    End If

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

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

```

```

    Select Case Count_Indexes_Values
    Case Is = 1
        Listbox_Selected = SelectedCount
    Case Is = 2
        Listbox_Selected = SelectedIndexes
    Case Is = 3
        Listbox_Selected = SelectedValues
    End Select

```

```
End Select
```

```
End Function
```

```
Sub FoldersCreate(folderPath As String)
```

```
    'Create all the folders in a folder path
```

```
    Dim individualFolders() As String
```

```
    Dim tempFolderPath As String
```

```
    Dim arrayElement As Variant
```

```
    'Split the folder path into individual folder names
```

```
    individualFolders = Split(folderPath, "\")
```

```
    'Loop though each individual folder name
```

```
    On Error Resume Next
```

```
    For Each arrayElement In individualFolders
```

```
        'Build string of folder path
```

```
        tempFolderPath = tempFolderPath & arrayElement & "\"
```

```
        'If folder does not exist, then create it
```

```
        If Dir(tempFolderPath, vbDirectory) = "" Then
```

```
            MkDir tempFolderPath
```

```
        End If
```

```
    Next arrayElement
```

```
End Sub
```

```
Sub ListboxClearSelection(LBox As MSForms.ListBox)
```

```
    On Error Resume Next
```

```
    For i = 0 To LBox.ListCount
```

```
        LBox.Selected(i) = False
```

```
    Next i
```

```
End Sub
```

```
Sub ListboxSelectValue(LBox As MSForms.ListBox, str As String, Optional _  
clr As Boolean = True)
```

```
    If Not clr Is Nothing Then
```

```
        Call ListboxClearSelection(LBox)
```

```
    End If
```

```
    For i = 0 To LBox.ListCount - 1
```

```
        If LBox.List(i) = str Then
```

```
            LBox.Selected(i) = True
```

```
            Exit Sub
```

```
        End If
```

```
    Next i
```

```
End Sub
```

--- UpdateAddin ---

```
'
'Sub ExportNotes()
'  Dim fso As Object
'  Set fso = VBA.CreateObject("Scripting.FileSystemObject")
'
'  Application.ScreenUpdating = False
'  Application.DisplayAlerts = False
'  Dim strPath As String
'
'  strPath = memoPath                                'PickFolder
'  If strPath = "" Then Exit Sub
'
'  On Error Resume Next
'  '  Kill strPath & "MemoryKnots.xlam"
'  fso.CopyFile Workbooks("MemoryKnots.xlam").FullName, strPath
'
'  Workbooks("MemoryKnots.xlam").Sheets.Copy
'
'  'Workbooks("MemoryKnots.xlam").Sheets(Array("SE
'TTINGS", "o NOTES", "o RESOLVED")).Copy  'or .delete
'
'  ActiveWorkbook.Sheets("SETTINGS").Delete
'  ActiveWorkbook.SaveAs Filename:=strPath & "MemoryKnots.xlsx", _
'                        FileFormat:
'xlOpenXMLWorkbook, CreateBackup:=False
'  ActiveWorkbook.Saved = True
'  ActiveWorkbook.Close
'  CreateObject("WScript.Shell").PopUp "Suc
'cessfully exported to " & Chr(10) & strPath, 1
'  'MsgBox "Successfully exported to " & Chr(10) & strPath
'  ActiveWindow.Close
'  Application.DisplayAlerts = True
'  Application.ScreenUpdating = True
'
'End Sub

Function PickFolder()
  With Application.FileDialog(msoFileDialogFolderPicker)
    .Title=
    .ButtonName=
    .InitialFileName = Environ("USERprofile") & "\Desktop\"
    'ThisWorkbook.Path 'Left(ThisWorkbook.Path, InStrRev(ThisWorkbook
    '.Path, "\"))
    If .Show = -1 Then          ' if OK is pressed
      PickFolder = .SelectedItems(1) & "\"
    Else
      Exit Function
    End If
  End With
End Function
```

```

'
'Sub ImportNotes()
'   Dim strADMIN As String
'   Dim answer As Integer
'   answer = MsgBox("ATTENTION!" & Chr(10) & Chr(10) & _
'                   "Present Notebooks will be DELETED
'and REPLACED from IMPORT file" & Chr(10) & Chr(10) & _
'                   "Proceed? (YES) or Cancel import? (NO)", _
'                   vbYesNo)
'   If answer = vbYes Then
'       '       KeepLoading = True
'   Else
'       '       'stop update to clear list(0)= update
'       '       strADMIN = InputBox("Non ADMIN ignore this")
'       '       If UCase(strADMIN) = "FREEZE" Then
'           '       KeepLoading = True
'           '       '       Workbooks("MemoryKno
'ts").Sheets("o NOTES").Cells(2, 2).EntireRow.Delete
'           '       Exit Sub
'       '       End If
'       '       'boolean to force unload userform
'       '       KeepLoading = False
'       Exit Sub
'   End If

'   Dim AddinWorkbook As Workbook
'   Set AddinWorkbook = Workbooks("MemoryKnots.xlam")

'   Dim ImportWorkbook As Workbook
'   If IsWorkBookOpen("MemoryKnots.xlsx") Then
'       Set ImportWorkbook = Workbooks("MemoryKnots.xlsx")
'   Else
'       If Dir(memoPath & "MemoryKnots.xlsx") > Len(memoPath) Then
'           Set ImportWorkbook = Workb
'ooks.Open(memoPath & "MemoryKnots.xlsx")
'       Else
'           CreateObject("WScript.Shell").PopUp
'"MemoryKnots.xlsx not found. Use EXPORT first.", 1
'           '           MsgBox "Memor
'yKnots.xlsx not found. Use EXPORT first."
'           GoTo cleanup
'       End If
'   End If

'   ImportWorkbook.Save

'   'import procedure

```

```

' Application.ScreenUpdating = False
'
' AddinWorkbook.IsAddin = False
' Dim ws As Worksheet
'
'     Set ws = AddinWorkbook.Sheets.Add(before:=Sheets(1))
'     ws.Name = "tmp"
'     Set ws = Nothing
'
' Application.DisplayAlerts = False
'
' For Each ws In AddinWorkbook.Worksheets
'     If ws.Name <> "SETTINGS" Then ws.Delete
' Next ws
'
' 'format sheets to be imported
' Dim cell As Range
' For Each ws In ImportWorkbook.Worksheets
'     If Left(ws.Name, 1) = "o" Then
'         For Each cell In ws.Columns(2)
'             If cell.Value = "" Then Exit For
'             If cell.Offset(0, -1) = "" Then
'                 cell.Offset(0, -1) = Now()
'             End If
'         Next cell
'     End If
' Next ws
'
'
' 'import sheets
' For Each ws In ImportWorkbook.Worksheets
'     If ws.Name <> "SETTINGS" Then
'         ws.Copy After:=AddinWork
'book.Sheets(AddinWorkbook.Sheets.count)
'     End If
' Next ws
'
'     AddinWorkbook.Worksheets("tmp").Delete
'
'cleanup:
' Application.DisplayAlerts = True
'
' AddinWorkbook.IsAddin = True
' Set ws = Nothing
' Set ImportWorkbook = Nothing
' Set AddinWorkbook = Nothing
'
' Application.ScreenUpdating = True
'End Sub
'
'Sub testgetfile()
'     Debug.Print GetFileToImport("xlsx", False)

```

```

'End Sub
'
'Function GetFileToImport(Optional fileType As S
'tring, Optional multiSelect As Boolean) As String
'
'    Dim blArray As Boolean
'    Dim strErrMsg As String, strTitle As String
'    strTitle = "Import Notebooks"
'
'    'check whether the file type parameter was passed
'    If IsMissing(fileType) Then
'        Exit Function
'    End If
'    'proceed
'    If strErrMsg = vbNullString Then
'        ' set title of dialog box
'        With Application.FileDialog(msoFileDialogFilePicker)
'            .InitialFileName = "MemoryKnots.xlsx" 'Environ("USERprofile") & "\D
'sktop\" 'ThisWorkbook.Path 'Left(ThisWorkbook.Path, InStrRev(ThisWorkbook.Path, "\
'))
'            .AllowMultiSelect = multiSelect
'            .Filters.Clear
'            If blArray Then .Filters.Add "File type", "*. " & fileType
'            .Title = strTitle
'
'            ' show the file picker dialog box
'            If .Show <> 0 Then
'                GetFilePath = .SelectedItems(1)
'            End If
'        End With
'        ' error message
'    Else
'        MsgBox strErrMsg, vbCritical, "Error!"
'    End If
'End Function

Function IsWorkBookOpen(Name As String) As Boolean
    Dim xWb As Workbook
    On Error Resume Next
    Set xWb = Application.Workbooks.Item(Name)
    IsWorkBookOpen = (Not xWb Is Nothing)
End Function

```

--- VoiceToWav ---

'Option Compare Database

Option Explicit

Rem found at <http://www.vbarchiv.net>

```
Public Declare PtrSafe Function sndPlaySound Lib "winmm.dll" Alias _  
"sndPlaySoundA" _  
(ByVal lpzSoundName As String, ByVal uFlags As Long) As Long
```

Const SND_SYNC = &H0

Const SND_ASYNC = &H1

Const SND_FILENAME = &H20000

```
Public Declare PtrSafe Function mciSendString Lib "winmm.dll" _  
Alias "mciSendStringA" ( _  
ByVal lpstrCommand As String, _  
ByVal lpstrReturnString As String, _  
ByVal uReturnLength As Long, _  
ByVal hwndCallback As Long) As Long
```

Public Enum BitsPerSec

Bits16 = 16

Bits8 = 8

End Enum

Public Enum SampelsPerSec

Sampels8000 = 8000

Sampels11025 = 11025

Sampels12000 = 12000

Sampels16000 = 16000

Sampels22050 = 22050

Sampels24000 = 24000

Sampels32000 = 32000

Sampels44100 = 44100

Sampels48000 = 48000

End Enum

Public Enum Channels

Mono = 1

Stereo = 2

End Enum

```
Public Sub play(file As String)  
    Dim wavefile  
    wavefile = file  
    Call sndPlaySound(wavefile, SND_ASYNC Or SND_FILENAME)  
End Sub
```

```
Public Sub StartRecord(ByVal BPS As BitsPerSec, _  
ByVal SPS As SampelsPerSec, ByVal Mode As Channels)
```



```

Dim retStr As String
Dim cBack As Long
Dim BytesPerSec As Long

retStr = Space$(128)
BytesPerSec = (Mode * BPS * SPS) / 8
mciSendString "open new type waveaudio alias capture", retStr, 128, _
cBack
mciSendString "set capture time format milliseconds" & _
" bitspersample " & CStr(BPS) & _
" samplespersec " & CStr(SPS) & _
" channels " & CStr(Mode) & _
" bytespersec " & CStr(BytesPerSec) & _
" alignment 4", retStr, 128, cBack
mciSendString "record capture", retStr, 128, cBack
End Sub

```

```

Public Sub SaveRecord(strFile)
Dim retStr As String
Dim TempName As String
Dim cBack As Long
Dim fs, F

TempName = strFile & "Left$(strFile, 3) & "Temp.wav"
retStr = Space$(128)
mciSendString "stop capture", retStr, 128, cBack
mciSendString "save capture " & TempName, retStr, 128, cBack
mciSendString "close capture", retStr, 128, cBack
End Sub

```

```

Public Sub StartRecord_Click()
VoiceToWav.StartRecord Bits16, Sampels32000, Mono
End Sub

```

```

Public Sub EndRecord_Click()
VoiceToWav.SaveRecord Environ("USERPROFILE") & "\Desktop\test.wav"
End Sub

```

```

Public Sub Play_Click()
VoiceToWav.play Environ("USERPROFILE") & "\Desktop\test.wav"
End Sub

```

--- StringToWav ---

'needs reference Microsoft Speech **Object** Library

Option Explicit

Sub TestStringToWavFile()

'run this to make a wav file from a **text** input

Dim sP **As** String, sFN **As** String, sStr **As** String, sFP **As** String

'set parameter values - insert your own profile name first

'paths

sP = Environ("USERPROFILE") & "\\Desktop\\" 'for example

sFN = "Mytest.wav" 'overwrites **if** file name same

sFP = sP & sFN

'string to use for the recording

sStr = "This **is** a short test **string** to be spoken in a user's wave _
file."

'make voice wav file from **string**

StringToWavFile sStr, sFP

End Sub

Function StringToWavFile(sIn **As** String, sPath **As** String) **As** Boolean

'makes a spoken wav file from parameter **text** **string**

'sPath parameter needs full path and file name to new wav file

'If wave file does **not** initially exist it will be made

'If wave file does initially exist it will be overwritten

'Needs reference **set** to Microsoft Speech **Object** Library

Dim fs **As** New SpFileStream

Dim Voice **As** New SpVoice

'set the audio format

fs.Format.Type = SAFT22kHz16BitMono

'create wav file for writing without events

fs.Open sPath, SSFMCreatForWrite, **False**

'Set wav file stream as output for Voice **object**

Set Voice.AudioOutputStream = fs

'send output to default wav file "SimpTTS.wav" and wait till done

Voice.Speak sIn, SVSFDefault

'Close file

fs.Close

'wait

Voice.WaitUntilDone (6000)

```
'release object variables
```

```
Set fs = Nothing
```

```
Set Voice.AudioOutputStream = Nothing
```

```
'transfers
```

```
StringToWavFile = True
```

```
End Function
```

--- ZipProc ---

```
' Zip a file or a folder to a zip file/folder using Windows Explorer.
' Default behaviour is similar to right
' clicking a file/folder and selecting:
'   Send to zip file.
'
' Parameters:
'   Path:
'       Valid (UNC) path to the file or folder to zip.
'   Destination:
'       (Optional) Valid (UNC) path to file with zip extension or other extension.
'   Overwrite:
'       (Optional) Leave (default) or overwrite an existing zip file.
'       If False, the created zip file will be versioned: Example.zip, Example (2).zip, etc.
'       If True, an existing zip file will first be deleted, then recreated.
'
'   Path and Destination can be relative paths. If so, the current path is used.
'
'   If success, 0 is returned, and Destination holds the full path of the created zip file.
'   If error, error code is returned, and Destination will be zero length string.
'
' Early binding requires references to:
'
'   Shell:
'       Microsoft Shell Controls And Automation
'
'   Scripting.FileSystemObject:
'       Microsoft Scripting Runtime
'
' 2017-10-22. Gustav Brock. Cactus Data ApS, CPH.
```

```
Public Function Zip( _
ByVal Path As String, _
Optional ByVal Destination As String, _
Optional ByVal Overwrite As Boolean) _
As Long

    #If EarlyBinding Then
        ' Microsoft Scripting Runtime.
        Dim FileSystemObject As Scripting.FileSystemObject
        ' Microsoft Shell Controls And Automation.
        Dim ShellApplication As Shell

        Set FileSystemObject = New Scripting.FileSystemObject
        Set ShellApplication = New Shell
    #Else
```

```

Dim FileSystemObject As Object
Dim ShellApplication As Object
Set FileSystemObject = CreateObject("Scripting.FileSystemObject")
Set ShellApplication = CreateObject("Shell.Application")
#End If

```

```

' Mandatory extension of zip file.
Const ZipExtensionName As String = "zip"
Const ZipExtension As String = "." & ZipExtensionName
' Custom error values.
Const ErrorPathFile As Long = 75
Const ErrorOther As Long = -1
Const ErrorNone As Long = 0
' Maximum (arbitrary) allowed count of created zip versions.
Const MaxZipVersion As Integer = 1000

```

```

Dim ZipHeader As String
Dim ZipPath As String
Dim ZipName As String
Dim ZipFile As String
Dim ZipBase As String
Dim ZipTemp As String
Dim Version As Integer
Dim result As Long

```

```

If FileSystemObject.FileExists(Path) Then
    ' The source is an existing file.
    ZipName = FileSystemObject.GetBaseName(Path) & ZipExtension
    ZipPath = FileSystemObject.GetFile(Path).ParentFolder
ElseIf FileSystemObject.FolderExists(Path) Then
    ' The source is an existing folder.
    ZipName = FileSystemObject.GetBaseName(Path) & ZipExtension
    ZipPath = FileSystemObject.GetFolder(Path).ParentFolder
Else
    ' The source does not exist.
End If

```

```

If ZipName = "" Then
    ' Nothing to zip. Exit.
    Destination = ""
Else
    If Destination <> "" Then
        If FileSystemObject.GetExtensionName(Destination) = "" _
            Then
            ' Destination is a folder.
            ZipPath = Destination
        Else
            ' Destination is a file.
            ZipName = FileSystemObject.GetFileName(Destination)
            ZipPath = FileSystemObject._
                GetParentFolderName(Destination)
        End If
    End If

```

```

Else
    ' Use the already found folder of the source.
End If

ZipFile = FileSystemObject.BuildPath(ZipPath, ZipName)
If FileSystemObject.FileExists(ZipFile) Then
    If Overwrite = True Then
        ' Delete an existing file.
        FileSystemObject.DeleteFile ZipFile, True
        ' At this point either
        'the file is deleted or an error is raised.
    Else
        ZipBase = FileSystemObject.GetBaseName(ZipFile)
        ' Modify name of the zip f
        'ile to be created to preserve an existing file:
        ' "Example.zip" -> "Example (2).zip", etc.
        Version = Version + 1
        Do
            Version = Version + 1
            ZipFile = FileSystemObject.BuildPath(ZipPath, _
            ZipBase & Format(Version, " \(\0\)") & _
            ZipExtension)
        Loop Until FileSystemObject.FileExists(ZipFile) = _
        False Or Version > MaxZipVersion
        If Version > MaxZipVersion Then
            ' Give up.
            err.Raise ErrorPathFile, "Zip Create", "File _
            could not be created."
        End If
    End If
End If

' Create a temporary zip name to allow for
' a final destination file with another extension than zip.
ZipTemp = FileSystemObject.BuildPath(ZipPath, _
FileSystemObject.GetBaseName(FileSystemObject.GetTempName()) _
& ZipExtension)
' Create empty zip folder.
' Header string provided by S
'tuart McLachlan <stuart@lexacorp.com.pg>.
ZipHeader = Chr(80) & Chr(75) & Chr(5) & Chr(6) & String(18, _
vbNullChar)
With FileSystemObject.OpenTextFile(ZipTemp, ForWriting, True)
    .Write ZipHeader
    .Close
End With

' Resolve relative paths.
ZipTemp = FileSystemObject.GetAbsolutePathName(ZipTemp)
Path = FileSystemObject.GetAbsolutePathName(Path)
' Copy the source file/folder into the zip file.
With ShellApplication
    Debug.Print Timer, "Zipping started . ";

```

```

        .Namespace(CVar(ZipTemp)).CopyHere CVar(Path)
        ' Ignore error while looking
        'up the zipped file before is has been added.
        On Error Resume Next
        ' Wait for the file to created.
        Do Until .Namespace(CVar(ZipTemp)).Items.count = 1
            ' Wait a little ...
            Application.Wait (Now + TimeValue("0:00:01"))
            Debug.Print ".";
        Loop
        Debug.Print
        ' Resume normal error handling.
        On Error GoTo 0
        Debug.Print Timer, "Zipping finished."
    End With
    ' Rename the temporary zip to its final name.
    FileSystemObject.MoveFile ZipTemp, ZipFile
End If

Set ShellApplication = Nothing
Set FileSystemObject = Nothing

If err.Number <> ErrorNone Then
    Destination = ""
    result = err.Number
ElseIf Destination = "" Then
    result = ErrorOther
End If

Zip = result
End Function

```

--- JPG ---

```
' Procedure : ExportRangeAsImage
' Author    : Daniel Pineault, CARDA Consultants Inc.
' Website   : http://www.cardaconsultants.com
' Purpose   : Capture a picture of a worksheet range and save it to disk
'            Returns True if the operation is successful
' Note      : *** Overwrites files, if
'already exists, without any warning! ***
' Copyright : The following is release as
'Attribution-ShareAlike 4.0 International
'            (CC BY-SA 4.0) - https://
'creativecommons.org/licenses/by-sa/4.0/
' Req'd Refs: Uses Late Binding, so none required
'
' Input Variables:
' ~~~~~
' ws          : Worksheet to capture the image of the range from
' rng         : Range to capture an image of
' sPath       : Fully qualified path where to export the image to
' sFileName   : filename to save the im
' age to WITHOUT the extension, just the name
' sImgExtension : The image file extension, commonly: JPG, GIF, PNG, BMP
'            If omitted will be JPG format
'
' Usage:
' ~~~~~
' ? ExportRangeAsImage(Sheets("Sheet1"), Range("A1"), "C:\Temp\Charts\", "test01". "JPG")
' ? ExportRangeAsImage(Sheets("Products"), Range("D5:F23"), "C:\Temp\Charts", "test02")
' ? ExportRangeAsImage(Sheets("Sheet1"), Range("A1"), "C:\Temp\Charts\", "test01", "PNG")
'
' Revision History:
' Rev      Date(yyyy/mm/dd)      Description
' *****
' *****
' 1         2020-04-06            Initial Release
' -----
' -----
- Function ExportRangeAsImage(ws As Worksheet, _
  rng As Range, _
  sPath As String, _
  sFileName As String, _
  Optional sImgExtension As String = "JPG") As Boolean
  Dim oChart As ChartObject

  On Error GoTo Error_Handler

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

  Application.ScreenUpdating = False
```



```

ws.Activate
rng.CopyPicture xlScreen, xlPicture      'Copy Range Content
Set oChart = ws.ChartObjects.Add(0, 0, rng.Width, rng.Height) _
'Add chart
oChart.Activate
With oChart.Chart
    .Paste      'Paste our Range
    .Export sPath & sFileName & "." & LCase(sImgExtension), _
    sImgExtension      'Export the chart as an image
End With
oChart.Delete      'Delete the chart
ExportRangeAsImage = True

Error_Handler_Exit:
    On Error Resume Next
    Application.ScreenUpdating = True
    If Not oChart Is Nothing Then Set oChart = Nothing
    Exit Function

Error_Handler:
    '76 - Path not found
    MsgBox "The following error has occurred" & vbCrLf & vbCrLf & _
    "Error Number: " & err.Number & vbCrLf & _
    "Error Source: ExportRangeAsImage" & vbCrLf & _
    "Error Description: " & err.Description & _
    Switch(Erl = 0, "", Erl <> 0, vbCrLf & "Line No: " & Erl) _
    , vbOKOnly + vbCritical, "An Error has Occurred!"
    Resume Error_Handler_Exit
End Function

```

--- Home ---

Sub Main()

uMemoryKnots.Show

End Sub

Sub MemoryKnotsButtonClicked(control **As** IRibbonControl)

Main

End Sub

--- uMemoryKnots ---

```
Public WithEvents WorksheetSelectionChangeCheck As Excel.Worksheet
```

```
Dim memoPath As String
```

```
Dim FolderToZip As String
```

```
Dim MemoryKnotsWB As Workbook
```

```
Dim MemoryKnotsWS As Worksheet
```

```
Dim cell As Range
```

```
Dim i As Long
```

```
Dim str As String
```

```
Dim strTMP As String
```

```
Dim tmpWS As Worksheet
```

```
Dim KeepLoading As Boolean
```

```
Dim RestoreTo As Worksheet
```

```
Private Sub AddBook_MouseDown(ByVal Button As Integer, ByVal Shift As _  
Integer, ByVal X As Single, ByVal Y As Single)
```

```
str = InputBox("New NoteBook name")
```

```
If str = "" Then Exit Sub
```

```
Set MemoryKnotsWS = MemoryKnotsWB.Sheets.Add(After:=MemoryKnotsWB. _  
Sheets(MemoryKnotsWB.Sheets.count))
```

```
noteBOOKS.AddItem ("o" & UCase(str))
```

```
With MemoryKnotsWS
```

```
    .Name = "o" & UCase(str)
```

```
    .[A1] = "DATE"
```

```
    .[B1] = "NOTES"
```

```
    .[A1:B1].Font.Bold = True
```

```
    ' .Visible = xlSheetHidden
```

```
End With
```

```
End Sub
```

```
Sub AddNotes()
```

```
    ' noteLIST.ListIndex = -1
```

```
    Call ListboxClearSelection(noteLIST)
```

```
    noteBOX.Text = ""
```

```
    Me.Width = 400
```

```
    noteBOX.SetFocus
```

```
End Sub
```

```
Private Sub AddNote_MouseDown(ByVal Button As Integer, ByVal Shift As _  
Integer, ByVal X As Single, ByVal Y As Single)
```

```
Call AddNotes
```

```
End Sub
```

```
Private Sub Clear_FilterNoteBooks_MouseDown(ByVal Button As Integer, _  
ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
```

```
On Error Resume Next
```

```
Dim tmp1 As String
```

```
tmp1 = noteBOOKS.List(noteBOOKS.ListIndex)
```

```
FilterNoteBooks.Text = ""
```

```
For i = 0 To noteBOOKS.ListCount - 1
```

```
    If noteBOOKS.List(i) = tmp1 Then
```

```

        noteBOOKS.Selected(i) = True
    End If
Next i
End Sub

```

```

Private Sub Clear_FilterNoteList_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
FilterNoteList.Text = ""
noteLIST.SetFocus

'    For i = 0 To noteLIST.ListCount - 1
'        If noteLIST.List(i) = tmp1 Then
'            noteLIST.Selected(i) = True
'        End If
'    Next i
End Sub

```

```

Private Sub CloseNoteBook_MouseDown(ByVal Button As Integer, ByVal Shift _
As Integer, ByVal X As Single, ByVal Y As Single)
If noteBOOKS.ListIndex < 0 Then
    MsgBox "No selection"
    Exit Sub
End If

```

```

MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS.ListIndex)).Visible = False
If Not tmpWS Is Nothing Then
    tmpWS.Activate
End If
End Sub

```

```

Private Sub cmdCopyAll_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)
If Listbox_Selected(noteLIST, 1) = 0 Or noteLIST.ListCount = 0 Then
    MsgBox "List empty or no item selected"
    Exit Sub
End If

```

```

Dim msg As String
Dim i As Long, j As Long
With noteLIST
    For i = 0 To .ListCount - 1
        msg = msg & .List(i) & vbCrLf
    Next i
End With
msg = Left(msg, Len(msg) - 2)
Call Clipboard(msg)
MsgBox "Selection copied"

End Sub

```

```

Private Sub cmdCopyAllLinked_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)

```

```

If Listbox_Selected(noteLIST, 1) = 0 Or noteLIST.ListCount = 0 Then
    MsgBox "List empty or no item selected"
    Exit Sub
End If

```

```

If Listbox_Selected(noteLIST, 1) < 2 Then
    MsgBox "What's the sound of one hand clapping?" & Chr(10) & Chr(10) & _
    _
    Space(50) & "~Fortune Cookie"
    Exit Sub
End If

```

```

Dim link As String
link = InputBox("Choose link")
If link = "" Then
    MsgBox "No input"
    Exit Sub
End If

```

```

Dim msg As String
Dim i As Long, j As Long
With noteLIST
    For i = 0 To .ListCount - 1
        msg = msg & .List(i) & link
    Next i
End With
msg = Left(msg, Len(msg) - 1)
Call Clipboard(msg)
End Sub

```

```

Private Sub cmdCopySelected_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
If Listbox_Selected(noteLIST, 1) = 0 Or noteLIST.ListCount = 0 Then
    MsgBox "List empty or no item selected"
    Exit Sub
End If

```

```

Dim answer As Long
answer = MsgBox("(YES) link with line break" & Chr(10) & _
"(NO) link with your choice", vbYesNoCancel)

```

```

If answer = vbCancel Then Exit Sub

```

```

Dim msg As String
Dim i As Long, j As Long
With noteLIST
    For i = 0 To .ListCount - 1
        If .Selected(i) Then
            If answer = vbYes Then
                msg = msg & .List(i) & vbCrLf
            Else
                msg = msg & .List(i) & link
            End If
        End If
    Next i
End With

```

```

    End If
  End If
Next i
End With

```

```

If answer = vbYes Then
    msg = Left(msg, Len(msg) - 2)
Else
    msg = Left(msg, Len(msg) - 1)
End If

```

```

Call Clipboard(msg)
MsgBox "Selection copied"
End Sub

```

```

Private Sub cmdExport_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)
Call ExportNotes
End Sub

```

```

Private Sub cmdExportAsImage_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
Call ExportAsImage
End Sub

```

```

Private Sub cmdExportAsImageMini_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
Call ExportAsImage
End Sub

```

```

Sub ExportAsImage()
If Not TypeName(Selection) = "Range" Then Exit Sub

```

```

    Dim JPGfolder As String
    JPGfolder = memoPath & "JPG"
    'On Error Resume Next
    'MkDir JPGfolder
    'On Error GoTo 0

```

```

    Dim action As Long

```

```

    'If Selection.Areas.count > 1 Then
    action = MsgBox("(YES) = for each area in selection" & Chr(10) & _
"(NO) = for each cell in selection", vbYesNoCancel)
    If action = vbCancel Then Exit Sub
    'Else
    '    action = vbNo
    'End If

```

```

    On Error Resume Next 'goto 0
    Application.DisplayAlerts = False

```

```

Dim JPGcell As Range
Dim result As String
Dim ImageExtension As String
ImageExtension = InputBox("Choose extension" & Chr(10) & _
"(1) = jpg" & Chr(10) & _
"(2) = bmp" & Chr(10) & _
"(3) = gib" & Chr(10) & _
"(4) = ico" & Chr(10) & _
"(5) = cur" & Chr(10) & _
"(6) = wmf", Default:=2)
If Not IsNumeric(ImageExtension) Or ImageExtension < 1 Or _
ImageExtension > 6 Then
    Exit Sub
End If

Select Case ImageExtension
Case Is = 1
    ImageExtension = "jpg"
Case Is = 2
    ImageExtension = "bmp"
Case Is = 3
    ImageExtension = "gib"
Case Is = 4
    ImageExtension = "ico"
Case Is = 5
    ImageExtension = "cur"
Case Is = 6
    ImageExtension = "wmf"
End Select

Select Case action
Case Is = vbNo
    For Each JPGcell In Selection
        Call ExportRangeAsImage(ActiveSheet, JPGcell, JPGfolder, _
JPGcell.Value, ImageExtension)
        Application.Wait (Now + TimeValue("0:00:01"))

        Set MemoryKnotsWS = MemoryKnotsWB.Sheets(noteBOOKS. _
List(noteBOOKS.ListIndex))
        Set cell = MemoryKnotsWS.Cells(Rows.count, 1).End(xlUp). _
Offset(1, 0)
        cell = Now()
        cell.Offset(0, 1) = JPGcell.Value & "." & ImageExtension
        noteLIST.AddItem JPGcell.Value & "." & ImageExtension
    Next JPGcell
Case Is = vbYes
    For i = 1 To Selection.Areas.count

```

```

        result = ""
        result = InputBox("name for image of area: " & Selection. _
        Areas(i).Address)

        Call ExportRangeAsImage(ActiveSheet, Selection.Areas(i), _
        JPGfolder, result, ImageExtension)
        Application.Wait (Now + TimeValue("0:00:01"))

        Set MemoryKnotsWS = MemoryKnotsWB.Sheets(noteBOOKS. _
        List(noteBOOKS.ListIndex))
        Set cell = MemoryKnotsWS.Cells(Rows.count, 1).End(xlUp). _
        Offset(1, 0)
        cell = Now()
        cell.Offset(0, 1) = result & "." & ImageExtension
        noteLIST.AddItem result & "." & ImageExtension
    Next i

```

```
End Select
```

```
Application.DisplayAlerts = True
```

```
End Sub
```

```

Private Sub cmdImport_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)
    Call ImportNotes
End Sub

```

```

Private Sub cmdInsertComment_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
    If Listbox_Selected(noteLIST, 1) = 0 Or noteLIST.ListCount = 0 Then
        MsgBox "List empty or no item selected"
        Exit Sub
    End If

```

```

    If Selection.Cells.count > 1 Then
        MsgBox ("choose 1 cell to insert comment")
        Exit Sub
    End If

```

```
Dim answer As Long
```

```

answer = MsgBox("(YES) link with line break" & Chr(10) & _
"(NO) link with your choice", vbYesNoCancel)

```

```
If answer = vbCancel Then Exit Sub
```

```

If answer = vbNo Then
    Dim link As String
    link = InputBox("Choose link")
End If

```



```
Dim msg As String
```

```
With noteLIST
```

```
For i = 0 To .ListCount - 1
```

```
    If .Selected(i) Then
```

```
        If answer = vbYes Then
```

```
            msg = msg & .List(i) & vbCrLf
```

```
        Else
```

```
            msg = msg & .List(i) & link
```

```
        End If
```

```
    End If
```

```
Next i
```

```
End With
```

```
If answer = vbYes Then
```

```
    msg = Left(msg, Len(msg) - 2)
```

```
Else
```

```
    msg = Left(msg, Len(msg) - 1)
```

```
End If
```

```
Dim action As Long
```

```
If ActiveCell.Comment Is Nothing Then
```

```
    ActiveCell.AddComment Format(Now(), "yymmdd hhmm") & " " & msg
```

```
Else
```

```
    action = MsgBox("(YES) pretend comment" & Chr(10) & _  
        "(NO) replace comment", vbYesNoCancel)
```

```
    If action = vbCancel Then Exit Sub
```

```
    If action = vbYes Then
```

```
        msg = Format(Now(), "yy/mm/dd hh:mm") & Chr(10) & Chr(10) & msg & _  
            Chr(10) & Chr(10) & ActiveCell.Comment.Text
```

```
        ActiveCell.Comment.Delete
```

```
        ActiveCell.AddComment msg
```

```
    Else
```

```
        ActiveCell.Comment.Delete
```

```
        ActiveCell.AddComment Format(Now(), "yy/mm/dd hh:mm") & Chr(10) & _  
            Chr(10) & msg
```

```
    End If
```

```
End If
```

```
ActiveCell.Comment.Shape.TextFrame.AutoSize = True
```

```
'    Cells(ActiveCell.Row + 1, ActiveCell.Column).Activate
```

```
End Sub
```

```
Private Sub cmdInsertSelectedToCells_MouseDown(ByVal Button As Integer, _
```

```
    ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
```

```
If Not TypeName(Selection) = "Range" Then Exit Sub
```

```
If Listbox_Selected(noteLIST, 1) = 0 Or noteLIST.ListCount = 0 Then
```

```
    MsgBox "List empty or no item selected"
```

```
Exit Sub
```

```
End If
```

```
Dim j As Long
```

```
Dim answer As Long
```

```
answer = MsgBox("(YES) insert vertically" & Chr(10) & _  
"(NO) insert horizontally", vbYesNoCancel)
```

```
If answer = vbCancel Then Exit Sub
```

```
Dim result As Long
```

```
If answer = vbYes Then
```

```
    If Selection.Cells.count = 1 Then
```

```
        j = 0
```

```
        With noteLIST
```

```
            For i = 0 To .ListCount - 1
```

```
                If .Selected(i) Then
```

```
                    Cells(ActiveCell.Row + j, ActiveCell.Column).Value = . _
```

```
                    List(i)
```

```
                    j = j + 1
```

```
                End If
```

```
            Next i
```

```
        End With
```

```
    Else
```

```
        result = MsgBox("You've selected " & Selection.Cells.count & " _
```

```
        cells." & Chr(10) & Chr(10) & _
```

```
        "Proceed to insert selected items vertically for each cell?", _
```

```
        vbYesNo)
```

```
        If result = vbNo Then Exit Sub
```

```
        If result = vbYes Then
```

```
            For Each cell In Selection.Cells
```

```
                j = 0
```

```
                With noteLIST
```

```
                    For i = 0 To .ListCount - 1
```

```
                        If .Selected(i) Then
```

```
                            Cells(cell.Row + j, cell.Column).Value = . _
```

```
                            List(i)
```

```
                            j = j + 1
```

```
                        End If
```

```
                    Next i
```

```
                End With
```

```
            Next cell
```

```
        End If
```

```
    End If
```

```
Else
```

```
    If Selection.Cells.count = 1 Then
```

```
        j = 0
```

```
        With noteLIST
```

```
            For i = 0 To .ListCount - 1
```

```
                If .Selected(i) Then
```

```
                    Cells(ActiveCell.Row, ActiveCell.Column + j).Value = . _
```

```

        List(i)
        j = j + 1
    End If
Next i
End With
Else
    result = MsgBox("You've selected " & Selection.Cells.count & " _
cells." & Chr(10) & Chr(10) & _
"Proceed to insert selected items vertically for each cell?", _
vbYesNo)
    If result = vbYes Then
        For Each cell In Selection.Cells
            j = 0
            With noteLIST
                For i = 0 To .ListCount - 1
                    If .Selected(i) Then
                        Cells(cell.Row, cell.Column + j).Value = . _
List(i)
                        j = j + 1
                    End If
                Next i
            End With
        Next cell
    End If
End If
End Sub

```

```

Private Sub cmdInsertSelectedMerged_MouseDown(ByVal Button As Integer, _
ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
    If TypeName(Selection) <> "Range" Then Exit Sub

```

```

    If Listbox_Selected(noteLIST, 1) = 0 Or noteLIST.ListCount = 0 Then
        MsgBox "List empty or no item selected"
        Exit Sub
    End If

```

```

Dim answer As Long
answer = MsgBox("(YES) link with line break" & Chr(10) & _
"(NO) link with your choice", vbYesNoCancel)

```

```

If answer = vbCancel Then Exit Sub

```

```

If answer = vbNo Then
    Dim link As String
    link = InputBox("Choose link")
End If

```

```

Dim msg As String
With noteLIST
    For i = 0 To .ListCount - 1
        If .Selected(i) Then

```

```

    If answer = vbYes Then
        msg = msg & .List(i) & vbCrLf
    Else
        msg = msg & .List(i) & link
    End If
End If
Next i
End With

```

```

If answer = vbYes Then
    ActiveCell.Value = Left(msg, Len(msg) - 2)
Else
    ActiveCell.Value = Left(msg, Len(msg) - 1)
End If
'    Cells(ActiveCell.Row + 1, ActiveCell.Column).Activate

```

End Sub

```

Private Sub cmdMail_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)
If Listbox_Selected(noteLIST, 1) = 0 Or noteLIST.ListCount = 0 Then
    MsgBox "List empty or no item selected"
    Exit Sub
End If

```

```

Dim WshShell As Object
Set WshShell = CreateObject("WScript.Shell")
'    Dim FolderToZip As String
FolderToZip = WshShell.SpecialFolders("MyDocuments")
FolderToZip = FolderToZip & "\FolderToZip\"

```

Call FoldersCreate("FolderToZip")

```

Dim AttachPath As String
AttachPath = MailAttachments

```

```

On Error Resume Next
Kill memoPath & "zipNotes.zip"
On Error GoTo 0

```

```

Dim msg As String
With noteLIST
    For i = 0 To .ListCount - 1
        If .Selected(i) Then
            msg = msg & .List(i) & vbCrLf
        End If
    Next i
End With

```

```

If OutlookCheck = True Then
    'For Tips see: http://www.rondebruin.nl/win/winmail/Outlook/tips.htm
    'Working in Office 2000-2016

```

```
Dim OutApp As Object
Dim OutMail As Object
```

```
Set OutApp = CreateObject("Outlook.Application")
Set OutMail = OutApp.CreateItem(0)
```

```
On Error Resume Next
```

```
With OutMail
    .To = ""
    .CC = ""
    .BCC = ""
    .Subject = "MemoryKnots"
    .body = msg
    If AttachPath <> "" Then
        .Attachments.Add (MailAttachments)
    End If
    .Display
    ' .Send
End With
```

```
Set OutMail = Nothing
Set OutApp = Nothing
```

```
Else
```

```
Call Clipboard(msg)
MsgBox "Outlook was not found" & Chr(10) & _
"Notes have been COPIED" & Chr(10) & _
"Please go to your mail and PASTE" & Chr(10) & _
"If you've included wav/image notes, they have been zipped at " & _
Chr(10) & _
memoPath & "zipNotes.zip"
```

```
End If
```

```
' On Error GoTo 0
```

```
On Error Resume Next
```

```
Kill FolderToZip & "*.*)"
Rmdir FolderToZip
```

```
End Sub
```

```
Function MailAttachments()
```

```
Dim fso As Object
Set fso = VBA.CreateObject("Scripting.FileSystemObject")
```

```
Dim FileToCopy As String
```

```
With noteLIST
```

```
For i = 0 To .ListCount - 1
    If .Selected(i) = True Then
        If (Right(.List(i), 3) = "wav" _
        Or Right(.List(i), 3) = "jpg" Or _
        Right(.List(i), 3) = "bmp" Or _
        Right(.List(i), 3) = "gib" Or _
```

```

        Right(.List(i), 3) = "ico" Or _
        Right(.List(i), 3) = "cur" Or _
        Right(.List(i), 3) = "wmf") Then
            FileToCopy = memoPath & .List(i)
            fso.CopyFile FileToCopy, FolderToZip
        End If
    End If
Next i
End With

If MailAttachmentsCount > 0 Then
    Call Zip(FolderToZip, memoPath & "zipNotes.zip")
    '        Call ZipFolder(FolderToZip, MemoPath & "wav.zip")
    MailAttachments = memoPath & "zipNotes.zip"
Else
    MailAttachments = ""
End If

End Function

Function MailAttachmentsCount()
    Dim Path As String, count As Integer

    Path = FolderToZip & ".*"

    Filename = Dir(Path)

    Do While Filename <> ""
        count = count + 1
        Filename = Dir()
    Loop
    MailAttachmentsCount = count
End Function

Private Sub cmdMoveNote_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)
If Listbox_Selected(noteLIST, 1) = 0 Then Exit Sub
'    Set cell = MemoryKnotsWB.Sheets(noteBOOKS.
'List(noteBOOKS.ListIndex)).Columns("B:B").Find( _
'        What:=noteLIST.List(noteLIST.ListIndex), _
'        LookIn:=xlFormulas, _
'        LookAt:=xlWhole, _
'        SearchOrder:=xlByRows, _
'        SearchDirection:=xlNext, _
'        MatchCase:=False, _
'        SearchFormat:=False)

noteSELECT.Clear
For i = 0 To noteBOOKS.ListCount - 1
    noteSELECT.AddItem noteBOOKS.List(i)
Next i
noteSELECT.ListIndex = -1

```

```

FrameSelect.Width = 84
FrameSelect.ZOrder (0)
FrameSelect.Visible = True

End Sub

Private Sub cmdMoveOK_Click()

'If noteSELECT.ListIndex <> -1 Then
'    strTMP = noteSELECT.List(noteSELECT.ListIndex)
'    cell.EntireRow.Copy MemoryKnotsWB.Sheets(str
'TMP).Range("A" & Rows.Count).End(xlUp).Offset(1, 0)
'    cell.EntireRow.Delete
'    noteBOX.Value = ""
'    noteLIST.RemoveItem (noteLIST.ListIndex)
'
'    noteBOOKS.ListIndex = -1
'    For i = 0 To noteBOOKS.ListCount - 1
'        If noteBOOKS.List(i) = strTMP Then
'            noteBOOKS.Selected(i) = True
'        Exit For
'    End If
'    Next i
'
'    noteSELECT.Clear
'    FrameSelect.Visible = False
'Else
'    strTMP = ""
'    noteSELECT.Clear
'    FrameSelect.Visible = False
'End If

```

```

If noteSELECT.ListIndex = -1 Then
    MsgBox "List empty or no item selected"
    noteSELECT.Clear
    FrameSelect.Visible = False
    Exit Sub
Else

    Dim moveWhat As Variant
    moveWhat = Split(Listbox_Selected(noteLIST, 2), ",")
    Dim moveTo As String
    moveTo = noteSELECT.List(noteSELECT.ListIndex)

    For i = UBound(moveWhat) To LBound(moveWhat) Step -1

        Set cell = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
ListIndex)).Columns("B:B").Find( _
What:=noteLIST.List(moveWhat(i)), _
LookIn:=xlFormulas, _
LookAt:=xlWhole, _
SearchOrder:=xlByRows, _

```

```

SearchDirection:=xlNext, _
MatchCase:=True, _
SearchFormat:=False)

cell.EntireRow.Copy MemoryKnotsWB.Sheets(moveTo).Range("A" & Rows. _
count).End(xlUp).Offset(1, 0)
cell.EntireRow.Delete

```

```

- Next i
noteSELECT.Clear
FrameSelect.Visible = False
End If

```

```

For i = UBound(moveWhat) To LBound(moveWhat) Step -1
noteLIST.RemoveItem (i)
Next i

```

```

If ToggleExtra.Value = False Then
FrameExtra.Width = 5
FrameExtra.Visible = False
End If
End Sub

```

```

Private Sub cmdMoveX_Click()
' strTMP = ""
noteSELECT.Clear
FrameSelect.Visible = False
End Sub

```

```

Private Sub cmdNewNoteFromSelection_MouseDown(ByVal Button As Integer, _
ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
Call ListboxClearSelection(noteLIST)
noteBOX.Text = SelectionValues(Chr(10))
Me.Width = 400
noteBOX.SetFocus
End Sub

```

```

Private Sub cmdNewNoteFromSelectionMini_MouseDown(ByVal Button As Integer, _
ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
Call ListboxClearSelection(noteLIST)
noteBOXmini.Text = SelectionValues(Chr(10))
noteBOXmini.SetFocus
End Sub

```

```

Private Sub cmdNewNotesFromSelection_MouseDown(ByVal Button As Integer, _
ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
Dim i As Long
Dim varr As Variant
varr = Split(SelectionValues("``"), "``")
Call ListboxClearSelection(noteLIST)
noteBOX.Text = ""

```



```

For i = 0 To UBound(varr)
    noteBOX.Text = varr(i)
    Call NoteSave
Next
End Sub

Private Sub cmdNewNotesFromSelectionMini_MouseDown(ByVal Button As _
Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
    Dim i As Long
    Dim varr As Variant
    varr = Split(SelectionValues("``"), "``")
    Call ListboxClearSelection(noteLIST)
    noteBOXmini.Text = ""
    For i = 0 To UBound(varr)
        noteBOXmini.Text = varr(i)
        Call NoteSaveMini
    Next
End Sub

Private Sub cmdPlayWAV_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)
    If Listbox_Selected(noteLIST, 1) = 1 Then
        - If Right(noteLIST.List(noteLIST.ListIndex), 3) = "wav" Then
            VoiceToWav.play memoPath & noteLIST.List(noteLIST.ListIndex)
        Else
            MsgBox "Not wav file, or file was manually deleted"
        - End If
    End If
End Sub

Private Sub cmdSettingsNotebooks_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
    Dim result As String
    result = InputBox("Change noteBOOKS font size", Default:=Me.noteBOOKS. _
Font.Size)
    If result = "" Then Exit Sub
    If Not IsNumeric(result) Then
        MsgBox "Font size must be numeric"
        Exit Sub
    End If
    ThisWorkbook.Sheets("SETTINGS").Range("noteBooksFontSize").Value = result
    Me.noteBOOKS.Font.Size = result
End Sub

Private Sub cmdSettingsNotebox_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
    Dim result As String
    result = InputBox("Change noteLIST font size", Default:=Me.noteBOX.Font. _
Size)
    If result = "" Then Exit Sub
    If Not IsNumeric(result) Then
        MsgBox "Font size must be numeric"

```

```

Exit Sub
End If
ThisWorkbook.Sheets("SETTINGS").Range("noteBoxFontSize").Value = result
Me.noteBOX.Font.Size = result
End Sub

Private Sub cmdSettingsNotelist_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
Dim result As String
result = InputBox("Change noteLIST font size", Default:=Me.noteLIST.Font. _
Size)
If result = "" Then Exit Sub
If Not IsNumeric(result) Then
MsgBox "Font size must be numeric"
Exit Sub
End If
ThisWorkbook.Sheets("SETTINGS").Range("noteListFontSize").Value = result
Me.noteLIST.Font.Size = result
End Sub

Private Sub cmdSpeechToWavStartRecording_MouseDown(ByVal Button As _
Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
Call VoiceNoteRecord
End Sub

Sub VoiceNoteRecord()
VoiceToWav.StartRecord Bits16, Sampels32000, Mono
cmdSpeechToWavStopRecording.Visible = True
cmdSpeechToWavStartRecording.Visible = False
End Sub

Private Sub cmdSpeechToWavStartRecordingMini_MouseDown(ByVal Button As _
Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
Call VoiceNoteRecord
End Sub

Private Sub cmdSpeechToWavStopRecording_MouseDown(ByVal Button As Integer, _
ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
Call VoiceNoteSave
End Sub

Sub VoiceNoteSave()
VoiceToWav.SaveRecord Environ("USERPROFILE") & "\Desktop\" & "tmp.wav"

result = ""
result = InputBox("Filename")
If result = "" Then
Exit Sub
End If
result = result & ".wav"

Name Environ("USERPROFILE") & "\Desktop\" & "tmp.wav" As memoPath & _

```

```
result
```

```
Set MemoryKnotsWS = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _  
ListIndex))  
Set cell = MemoryKnotsWS.Cells(Rows.count, 1).End(xlUp).Offset(1, 0)  
cell = Now()  
cell.Offset(0, 1) = result  
noteLIST.AddItem result
```

```
cmdSpeechToWavStopRecording.Visible = False  
cmdSpeechToWavStartRecording.Visible = True
```

```
End Sub
```

```
Private Sub cmdSpeechToWavStopRecordingMini_MouseDown(ByVal Button As _  
Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)  
Call VoiceNoteSave  
End Sub
```

```
Private Sub cmdTextToWav_MouseDown(ByVal Button As Integer, ByVal Shift _  
As Integer, ByVal X As Single, ByVal Y As Single)  
Call SaveTextToWav(1)  
End Sub
```

```
Sub SaveTextToWav(NormalOrMini As Long)
```

```
'run this to make a wav file from a text input
```

```
Dim sP As String, sFN As String, sStr As String, sFP As String
```

```
Dim result As String
```

```
'set parameter values - insert your own profile name first
```

```
'paths
```

```
result = InputBox("Record title?" & Chr(10) & Chr(10) & "Overwrites _  
if file name same")
```

```
- If result = "" Then
```

```
Exit Sub
```

```
- End If
```

```
sFN = result & ".wav"
```

```
- Select Case NormalOrMini
```

```
Case Is = 1
```

```
result = noteBOX.Text
```

```
Case Is = 2
```

```
result = noteBOXmini.Text
```

```
- End Select
```

```
- If result = "" Then
```

```
' Kill ThisWorkbook.Path & "\" & result
```

```
Exit Sub
```

```
- End If
```

```
'string to use for the recording
```

```
sStr = result
```

```

'make voice wav file from string
'"My.wav" 'overwrites if file name same
SP = memoPath
sFP = sP & sFN
StringToWavFile sStr, sFP

Application.ScreenUpdating = False
Dim tmpSheet As Worksheet
Set tmpSheet = ActiveSheet
Dim i As Integer
Dim j As Variant

Set MemoryKnotsWS = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
ListIndex))
Set cell = MemoryKnotsWS.Cells(Rows.count, 1).End(xlUp).Offset(1, 0)
cell = Now()
cell.Offset(0, 1) = sFN
noteLIST.AddItem sFN

```

```

Call ListboxClearSelection(noteLIST)
- Select Case NormalOrMini
Case Is = 1
    noteBOX.Text = ""
Case Is = 2
    noteBOXmini.Text = ""
- End Select

```

```

End Sub

```

```

Private Sub cmdTextToWavMini_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
Call SaveTextToWav(2)
End Sub

```

```

Private Sub DeleteBook_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)
If noteBOOKS.ListIndex < 0 Then
    MsgBox "Notebooks list empty or none selected"
Exit Sub

```

```

End If
If noteBOOKS.List(noteBOOKS.ListIndex) = "o NOTES" _
Or noteBOOKS.List(noteBOOKS.ListIndex) = "o RESOLVED" Then
Exit Sub
End If

```

```

Application.DisplayAlerts = False
MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS.ListIndex)).Delete
noteBOOKS.RemoveItem (noteBOOKS.ListIndex)
Application.DisplayAlerts = True
End Sub

```

```

Private Sub DeleteNote_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)

```

```

' If noteLIST.ListIndex < 0 Then
'     MsgBox "Note list empty or none selected"
'     Exit Sub
' End If
' Set cell = MemoryKnotsWB.Sheets(noteBOOKS.
' List(noteBOOKS.ListIndex)).Columns("B:B").Find( _
'     What:=noteLIST.List(noteLIST.ListIndex), _
'     LookIn:=xlFormulas, _
'     LookAt:=xlWhole, _
'     SearchOrder:=xlByRows, _
'     SearchDirection:=xlNext, _
'     MatchCase:=False, _
'     SearchFormat:=False)
' cell.EntireRow.Delete
' noteBOX.Value = ""
' noteLIST.RemoveItem (noteLIST.ListIndex)

If Listbox_Selected(noteLIST, 1) = 0 Or noteLIST.ListCount = 0 Then
    MsgBox "List empty or no item selected"
    Exit Sub
End If

'remove item
Dim deleteWhat As Variant
deleteWhat = Split(Listbox_Selected(noteLIST, 2), ",")

'move backwards when deleting
For i = UBound(deleteWhat) To LBound(deleteWhat) Step -1

    Set cell = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS.ListIndex)). _
    Columns("B:B").Find( _
    What:=noteLIST.List(deleteWhat(i)), _
    LookIn:=xlFormulas, _
    LookAt:=xlWhole, _
    SearchOrder:=xlByRows, _
    SearchDirection:=xlNext, _
    MatchCase:=False, _
    SearchFormat:=False)

    cell.EntireRow.Delete
    On Error Resume Next
    With noteLIST
        If Right(.List(i), 3) = "wav" Or Right(.List(i), 3) = "jpg" Or _
        Right(.List(i), 3) = "bmp" Or Right(.List(i), 3) = "gib" _
        Or Right(.List(i), 3) = "ico" Or Right(.List(i), 3) = "cur" Or _
        Right(.List(i), 3) = "wmf" Then
            Kill memoPath & noteLIST.List(deleteWhat(i))
        End If
    End With
    noteLIST.RemoveItem (deleteWhat(i))

Next i

```

```

noteBOX.Text = ""
If ToggleExtra.Value = False Then
    FrameExtra.Width = 5
    FrameExtra.Visible = False
End If

Me.Width = 174
End Sub

Private Sub DynamicImage_MouseDown(ByVal Button As Integer, ByVal Shift _
As Integer, ByVal X As Single, ByVal Y As Single)
Call Shell("explorer.exe" & " " & memoPath & noteLIST.List(noteLIST. _
ListIndex), vbNormalFocus)
End Sub

Private Sub FilterNoteBooks_Change()
On Error GoTo eh
'Reload list so if you type and delete you'll get the items back
Call LoadNoteBooks

Dim i           As Long
Dim n           As Long
Dim str         As String
Dim sTemp       As String

'Equals is always case sensitive
'Remove LCase if you want it to be case sensitive
str = LCase(FilterNoteBooks.Text)

n = noteBOOKS.ListCount

For i = n - 1 To 0 Step -1      'Work backwards when deleting items
    'Equals is always case sensitive
    'Remove LCase if you want it to be case sensitive
    sTemp = LCase(noteBOOKS.List(i))

    If InStr(sTemp, str) = 0 Then
        noteBOOKS.RemoveItem (i)
        'Exit Sub 'Uncomment to Exit if value found
    End If
Next i

noteBOOKS.Selected(0) = True
Exit Sub
eh:
End Sub

Private Sub FilterNoteList_Change()
On Error GoTo eh
'Reload list so if you type and delete you'll get the items back
Call LoadNoteList

```

```

Dim i As Long
Dim n As Long
Dim str As String
Dim sTemp As String

'Equals is always case sensitive
'Remove LCase if you want it to be case sensitive
str = LCase(FilterNoteList.Text)

n = noteLIST.ListCount

For i = n - 1 To 0 Step -1 'Work backwards when deleting items
    'Equals is always case sensitive
    'Remove LCase if you want it to be case sensitive
    sTemp = LCase(noteLIST.List(i))

    If InStr(sTemp, str) = 0 Then
        noteLIST.RemoveItem (i)
        'Exit Sub 'Uncomment to Exit if value found
    End If
Next i

noteLIST.Selected(0) = True
Exit Sub
eh:
End Sub

Private Sub cmdResolved_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)

If noteBOOKS.List(noteBOOKS.ListIndex) = "o RESOLVED" Then Exit Sub

If Listbox_Selected(noteLIST, 1) = 0 Or noteLIST.ListCount = 0 Then
    MsgBox "List empty or no item selected"
    Exit Sub
End If

Dim resolveWhat As Variant
resolveWhat = Split(Listbox_Selected(noteLIST, 2), ",")

For i = LBound(resolveWhat) To UBound(resolveWhat)

    Set cell = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS.ListIndex)). _
Columns("B:B").Find( _
What:=noteLIST.List(resolveWhat(i)), _
LookIn:=xlFormulas, _
LookAt:=xlWhole, _
SearchOrder:=xlByRows, _
SearchDirection:=xlNext, _
MatchCase:=False, _
SearchFormat:=False)

```

```

cell.EntireRow.Copy MemoryKnotsWB.Sheets("o RESOLVED").Range("A" & _
Rows.count).End(xlUp).Offset(1, 0)
cell.EntireRow.Delete

Next i
For i = UBound(resolveWhat) To LBound(resolveWhat) Step -1
    noteLIST.RemoveItem resolveWhat(i)
Next i

noteBOX.Value = ""

If ToggleExtra.Value = False Then
    FrameExtra.Width = 5
    FrameExtra.Visible = False
End If

End Sub

Private Sub Info_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)
    uDEV.Show
End Sub

Private Sub noteBOOKS_Change()
    On Error GoTo eh
    Set MemoryKnotsWS = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
ListIndex))
    noteLIST.Clear
    Call LoadNoteList
eh:
End Sub

Private Sub noteBOOKS_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
    If noteBOOKS.ListIndex = -1 Then Exit Sub
    If noteBOOKS.List(noteBOOKS.ListIndex) = "o NOTES" Or _
noteBOOKS.List(noteBOOKS.ListIndex) = "o RESOLVED" Then
        MsgBox "Can't touch this" & Chr(10) & _
        " ~mc Hammer"
    End If
Exit Sub
End If

Dim result As String
result = InputBox("New NoteBook name")
If result = "" Then Exit Sub
If WorksheetExists("o" & UCase(result)) Then
    MsgBox "Name taken"
Exit Sub
End If

Set MemoryKnotsWS = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
ListIndex))

```



```

With MemoryKnotsWS
    .Name = "o" & UCase(result)
End With

noteBOOKS.List(noteBOOKS.ListIndex) = "o" & UCase(result)
Set MemoryKnotsWS = Nothing
End Sub

Private Sub noteBOX_Exit(ByVal Cancel As MSForms.ReturnBoolean)
Me.Width = 174
End Sub

Private Sub noteLIST_Change()
DynamicImage.Visible = False

If Listbox_Selected(noteLIST, 1) <= 1 Then           'noteLIST.ListCount > 0 _
And noteLIST.ListIndex >= 0 And
noteBOX.Text = noteLIST.List(noteLIST.ListIndex)
If ToggleExtra.Value = False Then
    FrameExtra.Width = 5
    FrameExtra.Visible = False
End If
Else
    noteBOX.Text = ""
    FrameExtra.Width = 23
    FrameExtra.ZOrder (0)
    FrameExtra.Visible = True
End If

If Listbox_Selected(noteLIST, 1) = 1 Then
    - If Right(noteLIST.List(Listbox_Selected(noteLIST, 2)), 3) = "wav" Then
        cmdPlayWAV.Visible = True
        cmdPlayWAV.ZOrder (0)
        DynamicImage.Visible = False
        Me.Width = 174
    ElseIf Right(noteLIST.List(noteLIST.ListIndex), 3) = "jpg" Or _
        Right(noteLIST.List(noteLIST.ListIndex), 3) = "bmp" Or _
        Right(noteLIST.List(noteLIST.ListIndex), 3) = "gib" Or _
        Right(noteLIST.List(noteLIST.ListIndex), 3) = "ico" Or _
        Right(noteLIST.List(noteLIST.ListIndex), 3) = "cur" Or _
        Right(noteLIST.List(noteLIST.ListIndex), 3) = "wmf" Then
        cmdPlayWAV.Visible = False
        DynamicImage.Visible = True
        '
        '           On Error Resume Next
        '           DynamicImage.Picture = LoadPi
        '           ctture(MemoPath & noteLIST.List(noteLIST.ListIndex))
        '           DynamicImage.ZOrder (0)
        '           Me.Width = 400
    Else
        cmdPlayWAV.Visible = False
        DynamicImage.Visible = False

```

```

L End If
Else
    cmdPlayWAV.Visible = False
    DynamicImage.Visible = False
    Me.Width = 174
End If
End Sub

Private Sub noteLIST_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
If noteLIST.ListCount = 0 Then Exit Sub
If Right(noteLIST.List(noteLIST.ListIndex), 3) = "wav" Then
    str = InputBox("Rename WAV file" & Chr(10) & Chr(10) & "Will replace _
    file with same name.")
    - If str = "" Then
        Exit Sub
    Else
        On Error Resume Next
        Name memoPath & noteLIST.List(noteLIST.ListIndex) As _
        memoPath & str & ".wav"

        Set MemoryKnotsWS = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
        ListIndex))
        Set cell = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
        ListIndex)).Columns("B:B").Find( _
        What:=noteLIST.List(noteLIST.ListIndex), _
        LookIn:=xlFormulas, _
        LookAt:=xlWhole, _
        SearchOrder:=xlByRows, _
        SearchDirection:=xlNext, _
        MatchCase:=False, _
        SearchFormat:=False)

        cell.Offset(0, -1) = Now()
        cell = str & ".wav"

        noteLIST.List(noteLIST.ListIndex) = str & ".wav"

    - End If
ElseIf Right(noteLIST.List(noteLIST.ListIndex), 3) = "jpg" Or _
Right(noteLIST.List(noteLIST.ListIndex), 3) = "bmp" Or Right(noteLIST. _
List(noteLIST.ListIndex), 3) = "gib" _
Or Right(noteLIST.List(noteLIST.ListIndex), 3) = "ico" Or Right(noteLIST. _
List(noteLIST.ListIndex), 3) = "cur" Or Right(noteLIST.List(noteLIST. _
ListIndex), 3) = "wmf" Then

    str = InputBox("Rename file" & Chr(10) & Chr(10) & "Will replace file _
    with same name.")
    - If str = "" Then
        Exit Sub
    Else
        On Error Resume Next
        Name memoPath & noteLIST.List(noteLIST.ListIndex) As _

```

```

memoPath & str & Right(noteLIST.List(noteLIST.ListIndex), 3)

Set MemoryKnotsWS = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
ListIndex))
Set cell = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS.ListIndex)). _
Columns("B:B").Find( _
What:=noteLIST.List(noteLIST.ListIndex), _
LookIn:=xlFormulas, _
LookAt:=xlWhole, _
SearchOrder:=xlByRows, _
SearchDirection:=xlNext, _
MatchCase:=False, _
SearchFormat:=False)

cell.Offset(0, -1) = Now()
cell = str & Right(noteLIST.List(noteLIST.ListIndex), 3)

noteLIST.List(noteLIST.ListIndex) = str & Right(noteLIST. _
List(noteLIST.ListIndex), 3)
noteBOX.Text = str & Right(noteLIST.List(noteLIST.ListIndex), 3)
End If
Else
    Me.Width = 400
    noteBOX.SetFocus
End If
End Sub

Private Sub OpenNoteBook_MouseDown(ByVal Button As Integer, ByVal Shift _
As Integer, ByVal X As Single, ByVal Y As Single)
If noteBOOKS.ListIndex < 0 Then
    MsgBox "No selection"
    Exit Sub
End If

Set tmpWS = ActiveSheet
With MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS.ListIndex))
    .Visible = True
    .Activate
End With

End Sub

Private Sub OpenBackupFolder_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
Call Shell("explorer.exe" & " " & memoPath, vbNormalFocus)
End Sub

Private Sub cmdNoteSave_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)
Call NoteSave
End Sub

```

```

Sub NoteSave()
    If noteBOX.Text = "" Then
        MsgBox "Write a note first"
        Exit Sub
    End If

    Dim msg As String
    If ToggleRangeNote.Value = True And Selection.Cells.count = 1 Then
        msg = ActiveCell.Address(False, False) & " " & noteBOX.Text
    Else
        msg = noteBOX.Text
    End If

    Set MemoryKnotsWS = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
ListIndex))

    If Listbox_Selected(noteLIST, 1) = 0 Then
        Set cell = MemoryKnotsWS.Cells(Rows.count, 1).End(xlUp).Offset(1, _
0)
        cell = Now()
        cell.Offset(0, 1) = msg
        noteLIST.AddItem (msg)
    Else
        Set cell = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
ListIndex)).Columns("B:B").Find( _
What:=noteLIST.List(noteLIST.ListIndex), _
LookIn:=xlFormulas, _
LookAt:=xlWhole, _
SearchOrder:=xlByRows, _
SearchDirection:=xlNext, _
MatchCase:=False, _
SearchFormat:=False)

        cell.Offset(0, -1) = Now()
        cell = msg

        noteLIST.List(noteLIST.ListIndex) = msg
    End If

    Call ListboxClearSelection(noteLIST)
    noteBOX.Text = ""
End Sub

Private Sub cmdNoteSaveMini_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
    Call NoteSaveMini
End Sub

Sub NoteSaveMini()
    If noteBOXmini.Text = "" Then
        MsgBox "Write a note first"
        Exit Sub
    End If

```

```

- End If

Dim msg As String
- If ToggleRangeNote.Value = True And Selection.Cells.count = 1 Then
    msg = ActiveCell.Address(False, False) & " " & noteBOXmini.Text
Else
    msg = noteBOXmini.Text
- End If

Set MemoryKnotsWS = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
ListIndex))

- If Listbox_Selected(noteLIST, 1) = 0 Then
    Set cell = MemoryKnotsWS.Cells(Rows.count, 1).End(xlUp).Offset(1, _
    0)
    cell = Now()
    cell.Offset(0, 1) = msg
    noteLIST.AddItem (msg)
Else
    Set cell = MemoryKnotsWB.Sheets(noteBOOKS.List(noteBOOKS. _
ListIndex)).Columns("B:B").Find( _
What:=noteLIST.List(noteLIST.ListIndex), _
LookIn:=xlFormulas, _
LookAt:=xlWhole, _
SearchOrder:=xlByRows, _
SearchDirection:=xlNext, _
MatchCase:=False, _
SearchFormat:=False)

    cell.Offset(0, -1) = Now()
    cell = msg

    noteLIST.List(noteLIST.ListIndex) = msg
- End If

noteBOXmini.Text = ""
End Sub

Private Sub cmdSortNoteBooksAZ_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
On Error Resume Next
Dim tmp1 As String
tmp1 = noteBOOKS.List(noteBOOKS.ListIndex)
Call ListboxSortAZ(noteBOOKS)
For i = 0 To noteBOOKS.ListCount - 1
- If noteBOOKS.List(i) = tmp1 Then
    noteBOOKS.Selected(i) = True
- End If
Next i

End Sub

```

```

Private Sub cmdSortNoteBooksZA_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
On Error Resume Next
Dim tmp1 As String
tmp1 = noteBOOKS.List(noteBOOKS.ListIndex)
Call ListboxSortZA(noteBOOKS)
' Call ListboxClearSelection(notebOoks)
For i = 0 To noteBOOKS.ListCount - 1
    If noteBOOKS.List(i) = tmp1 Then
        noteBOOKS.Selected(i) = True
    End If
Next i
End Sub

```

```

Private Sub cmdSortNoteListAZ_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
On Error Resume Next
' Dim tmp1 As String
' tmp1 = noteLIST.List(noteLIST.ListIndex)
Call ListboxSortAZ(noteLIST)
Call ListboxClearSelection(noteLIST)
' For i = 0 To noteLIST.ListCount - 1
'     If noteLIST.List(i) = tmp1 Then
'         noteLIST.Selected(i) = True
'     End If
' Next i
End Sub

```

```

Private Sub cmdSortNoteListZA_MouseDown(ByVal Button As Integer, ByVal _
Shift As Integer, ByVal X As Single, ByVal Y As Single)
On Error Resume Next
' Dim tmp1 As String
' tmp1 = noteLIST.List(noteLIST.ListIndex)
Call ListboxSortZA(noteLIST)
Call ListboxClearSelection(noteLIST)
' For i = 0 To noteLIST.ListCount - 1
'     If noteLIST.List(i) = tmp1 Then
'         noteLIST.Selected(i) = True
'     End If
' Next i
End Sub

```

```

Private Sub Toggle_MouseDown(ByVal Button As Integer, ByVal Shift As _
Integer, ByVal X As Single, ByVal Y As Single)
'If notbooks.ListIndex = -1 Then
'    noteBOOKS.Selected(0) = True
'End If
Call ListboxClearSelection(noteLIST)
If Me.Height > 100 Then
    Me.Height = 64
    Me.Width = 174
    Me.Caption = noteBOOKS.List(noteBOOKS.ListIndex)

```

```

cmdNoteSaveMini.Visible = True
noteBOXmini.Visible = True
cmdSpeechToWavStartRecordingMini.Visible = True
cmdSpeechToWavStopRecordingMini.Visible = True
cmdTextToWavMini.Visible = True
cmdNewNoteFromSelectionMini.Visible = True
cmdNewNotesFromSelectionMini.Visible = True
cmdExportAsImageMini.Visible = True
noteBOXmini.SetFocus

```

Else

```

Me.Height = 336
'Me.Width = 174
Me.Caption = "Notebooks"
cmdNoteSaveMini.Visible = False
noteBOXmini.Visible = False
cmdSpeechToWavStartRecordingMini.Visible = False
cmdSpeechToWavStopRecordingMini.Visible = False
cmdTextToWavMini.Visible = False
cmdNewNoteFromSelectionMini.Visible = False
cmdNewNotesFromSelectionMini.Visible = False
cmdExportAsImageMini.Visible = False
noteBOOKS.SetFocus

```

End If

End Sub

Private Sub ToggleExtra_Click()

If ToggleExtra.Value = **True** **Then**

```

FrameExtra.Width = 23
FrameExtra.ZOrder (0)
FrameExtra.Visible = True

```

Else

```

FrameExtra.Width = 5
FrameExtra.Visible = False

```

End If

End Sub

Private Sub UserForm_Initialize()

'load position

If GetSetting("My Settings Folder", Me.Name, "Left Position") = "" **_**
And GetSetting("My Settings Folder", Me.Name, "Top Position") = "" **Then**

Me.StartUpPosition = 1 ' CenterOwner

Else

```

Me.Left = GetSetting("My Settings Folder", Me.Name, "Left Position")
Me.Top = GetSetting("My Settings Folder", Me.Name, "Top Position")

```

End If

'load position/

Me.Width = 174

Me.noteBOOKS.Font.Size = ThisWorkbook.Sheets("SETTINGS"). _

Range("noteBooksFontSize").Value

Me.noteLIST.Font.Size = ThisWorkbook.Sheets("SETTINGS"). _

Range("noteListFontSize").Value

```

Me.noteBOX.Font.Size = ThisWorkbook.Sheets("SETTINGS"). _
Range("noteBoxFontSize").Value
Me.noteBOXmini.Font.Size = 8          'ThisWorkbook.Sheets("SETTINGS"). _
Range("noteBoxFontSize").Value

Set MemoryKnotsWB = Workbooks(ThisWorkbook.Name)

' Set MemoryKnotsWS = MemoryKnotsWB.Sheets(1)

Call LoadNoteBooks

noteBOOKS.Selected(0) = True

Set WorksheetSelectionChangeCheck = ActiveSheet

memoPath = Environ$("USERPROFILE") & "\My Documents\vbArc\MemoryKnots\"
FoldersCreate memoPath
End Sub

Sub WorksheetSelectionChangeCheck_SelectionChange(ByVal Target As Range)
    If ToggleRangeNote.Value = False Then Exit Sub
    FilterNoteBooks.Text = ""
    FilterNoteList.Text = ""

    Dim noteLISTvalues As String

    For i = 0 To noteBOOKS.ListCount - 1
        noteLISTvalues = noteLISTvalues & " " & noteBOOKS.List(i)
        Debug.Print noteLISTvalues
    Next i

    If InStr(LCase(noteBOOKS.List(noteBOOKS.ListIndex)), _
    LCase(ActiveSheet.Name)) > 0 _
    And Selection.Cells.count = 1 Then
        For i = 0 To noteLIST.ListCount - 1
            noteLISTvalues = noteLISTvalues & " " & noteLIST.List(i)
            Debug.Print noteLISTvalues
        Next i
        If InStr(noteLISTvalues, ActiveCell.Address(False, False)) > 0 Then
            FilterNoteList.Text = ActiveCell.Address(False, False)
        Else
            FilterNoteList.Text = ""
        End If
    End If
End Sub

Sub ListboxValues(LBox As MSForms.ListBox)

End Sub

Sub CheckUpdate()

```



```

    On Error GoTo eh
    If UCase(noteLIST.List(0)) = "UPDATE" Then
        Call ImportNotes
    End If
eh:
End Sub

Private Sub LoadNoteList()
noteLIST.Clear
Dim i As Long
For i = 2 To MemoryKnotsWS.Cells(Rows.count, 1).End(xlUp).Row
    noteLIST.AddItem MemoryKnotsWS.Cells(i, 2)
Next
End Sub

Private Sub LoadNoteBooks()
noteBOOKS.Clear
For i = 1 To MemoryKnotsWB.Sheets.count
    If MemoryKnotsWB.Sheets(i).Name Like "o*" Then
        noteBOOKS.AddItem MemoryKnotsWB.Sheets(i).Name
    End If
Next
End Sub

Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
'/save position
'must have uf position set to manual
SaveSetting "My Settings Folder", Me.Name, "Left Position", Me.Left
SaveSetting "My Settings Folder", Me.Name, "Top Position", Me.Top

End Sub

```

--- uDEV ---

```
Private Sub LFaceBook_Click()
FollowLink ("https://www.facebook.com/VBA-Code-Archive-110295994460212")
End Sub

Private Sub LGitHub_Click()
FollowLink ("https://github.com/alexofrhodes")
End Sub

Private Sub LYouTube_Click()
FollowLink ("https://bit.ly/2QT4wFe")
End Sub

Private Sub LBuyMeACoffee_Click()
FollowLink ("http://paypal.me/alexofrhodes")
End Sub

Private Sub LEmail_Click()
If OutlookCheck = True Then
MailDev
Else
Dim out As String
out = "anastasioualex@gmail.com"
CLIP out
MsgBox ("Outlook not found" & Chr(10) & _
"DEV's email address" & vbNewLine & out & vbNewLine & "copied to _
clipboard")
End If
End Sub

Sub MailDev()
'For Tips see: http://www.rondebruin.nl/win/winmail/Outlook/tips.htm
'Working in Office 2000-2016
Dim OutApp As Object
Dim OutMail As Object
Dim strBody As String
Set OutApp = CreateObject("Outlook.Application")
Set OutMail = OutApp.CreateItem(0)
'    strbody = "Hi there" & vbNewLine & vbNewLine & _
"This is line 1" & vbNewLine & _
"This is line 2" & vbNewLine & _
"This is line 3" & vbNewLine & _
"This is line 4"
On Error Resume Next
With OutMail
.To = "anastasioualex@gmail.com"
.CC = vbNullString
.BCC = vbNullString
.Subject = "DEV REQUEST OR FEEDBACK FOR -CODE ARCHIVE-"
.body = strBody
'You can add a file like this
.Attachments.Add ("C:\test.txt")
End With
End Sub
```

```
        '.Send  
        .Display  
    End With  
    On Error GoTo 0  
    Set OutMail = Nothing  
    Set OutApp = Nothing  
End Sub
```

```
Private Sub UserForm_Click()
```

```
End Sub
```











e
"