MouseRecorder.xlsm

www.github.com/alexofrhodes

--- Table Of Contents ---

--- ThisWorkbook ---

--- uMouseRecorder - Sheet13 ---

--- README - Sheet22 ---

--- Home ---

```
Public Sub Main()
    uMouseRecorder.Show
End Sub

'Callback for buttonMightyMouse onAction
Sub MouseRecorderButtonClicked(control As IRibbonControl)
    Main
End Sub

```

--- mFormParent ---

```
Option Explicit
Option Compare Text
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' modSupport
' By Chip Pearson, chip@cpearson.com www.cpearson.com
'
' This module contains declarations and code that are use
'd in support of the procedures in frmSetParent but aren't
' directly related to the topic of the workbook.
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''
' used by FormatMessage
''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''
Public Const FORMAT_MESSAGE_ALLOCATE_BUFFER = &H100
Public Const FORMAT_MESSAGE_ARGUMENT_ARRAY = &H2000
Public Const FORMAT_MESSAGE_FROM_HMODULE = &H800
Public Const FORMAT_MESSAGE_FROM_STRING = &H400
Public Const FORMAT_MESSAGE_FROM_SYSTEM = &H1000
Public Const FORMAT_MESSAGE_IGNORE_INSERTS = &H200
Public Const FORMAT_MESSAGE_MAX_WIDTH_MASK = &HFF
Public Const FORMAT_MESSAGE_TEXT_LEN = 160        ' from ERRORS.H C++  _
include file.
'''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''
' Various constants
'''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''
Public Const MAX_PATH = 260        ' Windows mandated value
Public Const GWL_HWNDPARENT As Long = -8
Public Const GW_OWNER = 4
'''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''
' Windows API Declares
'''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''
Public Declare PtrSafe Function FormatMessage Lib "kernel32.dll" Alias  _
"FormatMessageA" (ByVal dwFlags As Long, ByRef lpSource As Any, ByVal  _
dwMessageId As Long, ByVal dwLanguageId As Long, ByVal lpBuffer As String, _
ByVal nSize As Long, ByRef Arguments As Long) As Long
Public Declare PtrSafe Function GetWindowLong Lib "user32.dll" Alias  _
"GetWindowLongA" (ByVal hWnd As Long, ByVal nIndex As Long) As Long
Public Declare PtrSafe Function GetClassName Lib "User32" Alias  _
"GetClassNameA" (ByVal hWnd As Long, ByVal lpClassName As String, ByVal  _
nMaxCount As Long) As Long
Public Declare PtrSafe Function GetWindowText Lib "user32.dll" Alias  _
"GetWindowTextA" (ByVal hWnd As Long, ByVal lpString As String, ByVal cch  _
As Long) As Long
```

```
''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''
' frmSetParent
' This form illustrates the SetParent proc
'edure to make a userform the a child window
' of no window, the Excel Application windo
'w, the Excel Desktop Window, and the Active
' Window.
''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Window Class Names
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Public Const C_EXCEL_APP_WINDOWCLASS = "XLMAIN"
Public Const C_EXCEL_DESK_WINDOWCLASS = "XLDESK"
Public Const C_EXCEL_WINDOW_WINDOWCLASS = "EXCEL7"
Public Const C_VBA_USERFORM_WINDOWCLASS = "ThunderDFrame"
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Window HWnds
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Public VBEditorHWnd As Long
Public ApplicationHWnd As Long
Public ExcelDeskHWnd As Long
Public ActiveWindowHWnd As Long
Public UserFormHWnd As Long
Public WindowsDesktopHWnd As Long
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Other Consts
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Public Const GA_ROOT As Long = 2
Public Const GA_ROOTOWNER As Long = 3
Public Const GA_PARENT As Long = 1
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Windows API Functions
''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Public Declare PtrSafe Function FindWindow Lib "User32" Alias  _
"FindWindowA" (ByVal lpClassName As String, ByVal lpWindowName As String)  _
As Long
Public Declare PtrSafe Function FindWindowEx Lib "User32" Alias  _
"FindWindowExA" (ByVal hWnd1 As Long, ByVal hWnd2 As Long, ByVal lpsz1 As  _
String, ByVal lpsz2 As String) As Long
Public Declare PtrSafe Function GetAncestor Lib "user32.dll" (ByVal hWnd  _
As Long, ByVal gaFlags As Long) As Long
Public Declare PtrSafe Function GetDesktopWindow Lib "User32" () As Long
Public Declare PtrSafe Function GetParent Lib "user32.dll" (ByVal hWnd As  _
Long) As Long
Public Declare PtrSafe Function GetWindow Lib "User32" (ByVal hWnd As  _
Long, ByVal wCmd As Long) As Long
Public Declare PtrSafe Function SetForegroundWindow Lib "User32" (ByVal  _
hWnd As Long) As Long
Public Declare PtrSafe Function SetParent Lib "User32" (ByVal hWndChild  _
As Long, ByVal hWndNewParent As Long) As Long
```

```
Public Declare PtrSafe Function SetWindowLong Lib "User32" Alias  _
"SetWindowLongA" (ByVal hWnd As Long, ByVal nIndex As Long, ByVal  _
dwNewLong As Long) As Long
'''
Rem Form on top
Public Const SWP_NOMOVE = &H2
Public Const SWP_NOSIZE = &H1
Public Const HWND_TOP = 0
Public Const HWND_BOTTOM = 1
Public Const HWND_TOPMOST = -1
Public Const HWND_NOTOPMOST = -2
Public Declare PtrSafe Function SetWindowPos Lib "User32" (ByVal hWnd As  _
LongPtr, ByVal hWndInsertAfter As LongPtr, ByVal X As LongPtr, ByVal Y As  _
LongPtr, ByVal cx As LongPtr, ByVal cy As LongPtr, ByVal uFlags As  _
LongPtr) As Long

Public Sub UserformOnTop(Form As Object)
    Const C_VBA6_USERFORM_CLASSNAME = "ThunderDFrame"
    Dim ret As Long
    Dim formHWnd As Long
    'Get window handle of the userform
    formHWnd = CLng(FindWindow(C_VBA6_USERFORM_CLASSNAME, Form.Caption))
    If formHWnd = 0 Then
        Debug.Print Err.LastDllError
    End If
    'Set userform window to 'always on top'
    ret = SetWindowPos(formHWnd, HWND_TOPMOST, 0, 0, 0, 0, SWP_NOMOVE Or  _
    SWP_NOSIZE)
    If ret = 0 Then
        Debug.Print Err.LastDllError
    End If
    'Application.WindowState = xlMinimized
End Sub

'''
Public Sub DisplayLabels(Form As Object)
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' DipsplayLabelText
    ' This gets the various HWnds if they are not already
    ' set and updates the label captions on the form.
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Dim ParentHWnd As Long
    Dim ParentWindowClass As String
    Dim AncestorWindow As Long
    Dim WinLong As Long
    Dim OwnerWindow As Long
    Dim ClassName As String
    Dim s As String
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' Get The HWnds
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

```
    ' HWnd of the VBEditor
    '''''''''''''''''''''''''''''''
    VBEditorHWnd = Application.VBE.mainwindow.hWnd
    '''''''''''''''''''''''''''''''
    ' HWnd of the Excel Application
    '''''''''''''''''''''''''''''''
    ApplicationHWnd = FindWindow(lpClassName:=C_EXCEL_APP_WINDOWCLASS,  _
    lpWindowName:=Application.Caption)
    '''''''''''''''''''''''''''''''
    ' HWnd of the Excel Desktop
    '''''''''''''''''''''''''''''''
    ExcelDeskHWnd = FindWindowEx(hWnd1:=ApplicationHWnd, hWnd2:=0&,  _
    lpsz1:=C_EXCEL_DESK_WINDOWCLASS, lpsz2:=vbNullString)
    '''''''''''''''''''''''''''''''
    ' HWnd of the ActiveWindow
    '''''''''''''''''''''''''''''''
    ActiveWindowHWnd = FindWindowEx(hWnd1:=ExcelDeskHWnd, hWnd2:=0&,  _
    lpsz1:=C_EXCEL_WINDOW_WINDOWCLASS, lpsz2:=Application.ActiveWindow. _
    Caption)
    '''''''''''''''''''''''''''''''
    ' HWnd of the UserForm
    '''''''''''''''''''''''''''''''
    UserFormHWnd = FindWindow(lpClassName:=C_VBA_USERFORM_WINDOWCLASS,  _
    lpWindowName:=Form.Caption)
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' update the Option Button Captions with WindowText
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Debug.Print "Active Window (" & GetHWndWindowText(ActiveWindowHWnd) &  _
    ")"
    Debug.Print "Application (" & GetHWndWindowText(ApplicationHWnd) & ")"
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' Update Labels with HWnds and Parent HWnds.
    ' Use GetWindowLong rather than GetParent to
    ' retrieve the Parent windows.
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ''''''''''''''''''''
    ' VBEditor
    ''''''''''''''''''''
    ClassName = GetWindowClassName(Application.VBE.mainwindow.hWnd)
    Debug.Print "VBEditor -- HWnd: " & CStr(Application.VBE.mainwindow. _
    hWnd) & _
    "  (Window Class: " & ClassName & ")"
    ''''''''''''''''''''
    ' Windows Desktop
    ''''''''''''''''''''
    ClassName = GetWindowClassName(GetDesktopWindow())
    Debug.Print "Windows Desktop -- HWnd: " & CStr(GetDesktopWindow()) & _
    "  (Window Class: " & ClassName & ")"
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' frmSetParent UserForm.   Class Name "ThunderDFrame".
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ParentHWnd = GetWindowLong(UserFormHWnd, GWL_HWNDPARENT)
```

```
    ClassName = GetWindowClassName(UserFormHWnd)
    ParentWindowClass = GetWindowClassName(ParentHWnd)
    Debug.Print "UserForm -- HWnd: " & CStr(UserFormHWnd) & " (Window  _
    Class: " & ClassName & _
    ")   Parent HWnd: " & CStr(ParentHWnd) & "  (Window Class: " &  _
    ParentWindowClass & ")"
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' ActiveWindow. Class Name "EXCEL7".
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ParentHWnd = GetWindowLong(ActiveWindowHWnd, GWL_HWNDPARENT)
    ClassName = GetWindowClassName(ActiveWindowHWnd)
    ParentWindowClass = GetWindowClassName(ParentHWnd)
    Debug.Print "ActiveWindow -- HWnd: " & CStr(ActiveWindowHWnd) & "  _
    (Window Class: " & ClassName & _
    ")   Parent HWnd: " & CStr(ParentHWnd) & "  (Window Class: " &  _
    ParentWindowClass & ")"
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' Information About The Excel Desktop. Class Name "XLDESK"
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ParentHWnd = GetWindowLong(ApplicationHWnd, GWL_HWNDPARENT)
    ParentHWnd = GetWindowLong(ExcelDeskHWnd, GWL_HWNDPARENT)
    ClassName = GetWindowClassName(ExcelDeskHWnd)
    ParentWindowClass = GetWindowClassName(ParentHWnd)
    Debug.Print "Excel Desktop -- HWnd: " & CStr(ExcelDeskHWnd) & "  _
    (Window Class: " & ClassName & _
    ")   Parent HWnd: " & CStr(ParentHWnd) & "  (Window Class: " &  _
    ParentWindowClass & ")"
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' Information About The Application Window. Class Name "XLMAIN"
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ParentHWnd = GetWindowLong(ApplicationHWnd, GWL_HWNDPARENT)
    ClassName = GetWindowClassName(ApplicationHWnd)
    ParentWindowClass = GetWindowClassName(ParentHWnd)
    Debug.Print "Excel Application -- HWnd: " & CStr(ApplicationHWnd) & "  _
    (Window Class: " & ClassName & _
    ")   Parent HWnd: " & CStr(ParentHWnd) & "  (Window Class: " &  _
    ParentWindowClass & ")"
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' Display information about the various Ances
    'tor values: GA_ROOT, GA_ROOTOWNER, and GA_PARENT.
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    AncestorWindow = GetAncestor(UserFormHWnd, GA_ROOT)
    ClassName = GetWindowClassName(AncestorWindow)
    s = "The Ancestor (GA_ROOT) of this UserForm is " &  _
    CStr(AncestorWindow) & "  (Window Class: " & ClassName & ")"
    AncestorWindow = GetAncestor(UserFormHWnd, GA_ROOTOWNER)
    ClassName = GetWindowClassName(AncestorWindow)
    s = s & vbCrLf & "The Ancestor (GA_ROOTOWNER) of this UserForm is " &  _
    CStr(AncestorWindow) & "  (Window Class: " & ClassName & ")"
    AncestorWindow = GetAncestor(UserFormHWnd, GA_PARENT)
```

```vb
    ClassName = GetWindowClassName(AncestorWindow)
    Debug.Print s & vbCrLf & "The Ancestor (GA_PARENT) of this UserForm  _
    is " & CStr(AncestorWindow) & "  (Window Class: " & ClassName & ")"
    ''''''''''''''''''''''''''''''''''''''''''''''''''''
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' Display informationa about this form's owner.
    ''''''''''''''''''''''''''''''''''''''''''''''''''''
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    OwnerWindow = GetWindow(UserFormHWnd, GW_OWNER)
    If OwnerWindow Then
        Debug.Print "The Owner Window of this UserForm is HWnd: " &  _
        CStr(OwnerWindow) _
        & "  (Window Class: " & GetWindowClassName(OwnerWindow) & ")"
    Else
        Debug.Print "There is no owner window of this form."
    End If
    Form.Repaint
End Sub

Public Sub MakeFormChildOfVBEditor(Form As Object)
    '''''''''''''''''''''''''''''''''''''''''''''''
    ''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' MakeFormChildOfVBEditor
    ' This makes the form a child window of the
    'VBEditor window (this is what you see if you have
    ' no workbooks open or all workbooks minimized).
    '''''''''''''''''''''''''''''''''''''''''''''''
    ''''''''''''''''''''''''''''''''''''''''''''''''''''
    Dim Res As Long
    Dim ParentHWnd As Long
    Dim ChildHWnd As Long
    Dim ErrNum As Long
    ''''''''''''''''''''''''''''''''''''''''''''''''
    ' Set Parent HWnd to ActiveWindowHWnd
    ''''''''''''''''''''''''''''''''''''''''''''''''
    ParentHWnd = VBEditorHWnd
    ''''''''''''''''''''''''''''''''''''''''''''''''
    ' Set Child HWnd to UserFormHWnd
    ''''''''''''''''''''''''''''''''''''''''''''''''
    ChildHWnd = UserFormHWnd
    ''''''''''''''''''''''''''''''''''''''''''''''''
    ' Call SetParent to make ChildHWnd a child of
    ' ParentHWnd.
    ''''''''''''''''''''''''''''''''''''''''''''''''
    Res = SetParent(hWndChild:=ChildHWnd, hWndNewParent:=ParentHWnd)
    If Res = 0 Then
        ''''''''''''''''''''''''''''
        ' an error occurred
        ''''''''''''''''''''''''''''
        ErrNum = Err.LastDllError
        DisplayErrorText "Error With SetParent", ErrNum
    Else
```

```
        Debug.Print "The UserForm is a child of the VBEditor window  _
        (Class wndclass_desked_gsk). "
    End If
    SetForegroundWindow UserFormHWnd
    Form.Repaint
End Sub

Public Sub MakeFormChildOfActiveWindow(Form As Object)
    '''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''''
    ' MakeFormChildOfActiveWindow
    ' This makes the form a child window of the ActiveWindow.
    '''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''''
    Dim Res As Long
    Dim ParentHWnd As Long
    Dim ChildHWnd As Long
    Dim ErrNum As Long
    '''''''''''''''''''''''''''''''''''''''''''''''''
    ' Set Parent HWnd to ActiveWindowHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''
    ' Updete the ActiveWindowHWnd
    '''''''''''''''''''''''''''''''
    If Application.ActiveWindow Is Nothing Then
        MsgBox "There is no active window."
        Exit Sub
    End If
    ActiveWindowHWnd = FindWindowEx(hWnd1:=ExcelDeskHWnd, hWnd2:=0&,  _
    lpsz1:=C_EXCEL_WINDOW_WINDOWCLASS, _
    lpsz2:=Application.ActiveWindow.Caption)
    ParentHWnd = ActiveWindowHWnd
    If ParentHWnd = 0 Then
        MsgBox "ParentHWnd Is 0 In MakeFormChildOfActiveWindow"
    End If
    '''''''''''''''''''''''''''''''''''''''''''''''''
    ' Set Child HWnd to UserFormHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''''
    ChildHWnd = UserFormHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''''
    ' Call SetParent to make ChildHWnd a child of
    ' ParentHWnd.
    '''''''''''''''''''''''''''''''''''''''''''''''''
    Res = SetParent(hWndChild:=ChildHWnd, hWndNewParent:=ParentHWnd)
    If Res = 0 Then
        ''''''''''''''''''''''''''''
        ' an error occurred
        ''''''''''''''''''''''''''''
        ErrNum = Err.LastDllError
        DisplayErrorText "Error With SetParent", ErrNum
    Else
        Debug.Print "The UserForm is a child of the ActiveWindow (Class:  _
```

```
        EXCEL7). Note that you cannot move the" & _
        " the form outside of the ActiveWindow, and that the form moves  _
        as you move the ActiveWindow. If" & _
        " you switch to another window such as another workbook, the form  _
        is not be visible until you restore" & _
        " the original window. Note that it is not possible to make a  _
        form a child of an individual worksheet tab."
    End If
    SetForegroundWindow UserFormHWnd
    Form.Repaint
End Sub

Public Sub MakeFormChildOfDesktop(Form As Object)
    '''''''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' MakeFormChildOfDesktop
    ' This makes the form a child window of the
    'Excel Desktop (this is what you see if you have
    ' no workbooks open or all workbooks minimized).
    '''''''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''''''''
    Dim Res As Long
    Dim ParentHWnd As Long
    Dim ChildHWnd As Long
    Dim ErrNum As Long
    '''''''''''''''''''''''''''''''''''''''''''''''
    ' Set Parent HWnd to ActiveWindowHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''
    ParentHWnd = ExcelDeskHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''
    ' Set Child HWnd to UserFormHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''
    ChildHWnd = UserFormHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''
    ' Call SetParent to make ChildHWnd a child of
    ' ParentHWnd.
    '''''''''''''''''''''''''''''''''''''''''''''''
    Res = SetParent(hWndChild:=ChildHWnd, hWndNewParent:=ParentHWnd)
    If Res = 0 Then
        '''''''''''''''''''''''''''''
        ' an error occurred
        '''''''''''''''''''''''''''''
        ErrNum = Err.LastDllError
        DisplayErrorText "Error With SetParent", ErrNum
    Else
        Debug.Print "The UserForm is a child of the Excel Desktop (Class  _
        XLDESK). The window may get lost behind the" & _
        " worksheet windows. In general, you'll never want to make the  _
        form a child of Excel Desktop unless you" & _
        " don't have any open workbooks, in which case it is better to  _
        make a form a child of the Application." & _
        " If the window gets lost, click on the Show Form button on  _
```

```
        Sheet1 to restore the form. The form will" & _
        " still be displayed if you minimize all open windows. Note that  _
        you cannot drag the form outside of the " & _
        " Excel Desktop's window."
    End If
    SetForegroundWindow UserFormHWnd
    Form.Repaint
End Sub

Public Sub MakeFormChildOfApplication(Form As Object)
    '''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''
    ' MakeFormChildOfApplication
    ' This makes the form a child of the main application window.
    '''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''
    Dim Res As Long
    Dim ParentHWnd As Long
    Dim ChildHWnd As Long
    Dim ErrNum As Long
    '''''''''''''''''''''''''''''''''''''''''''''''
    ' Set Parent HWnd to ActiveWindowHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''
    ParentHWnd = ApplicationHWnd
    If ParentHWnd = 0 Then
        MsgBox "ParentHWnd Is 0 In MakeFormChildOfApplication."
    End If
    '''''''''''''''''''''''''''''''''''''''''''''''
    ' Set Child HWnd to UserFormHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''
    ChildHWnd = UserFormHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''
    ' Call SetParent to make ChildHWnd a child of
    ' ParentHWnd.
    '''''''''''''''''''''''''''''''''''''''''''''''
    Res = SetParent(hWndChild:=ChildHWnd, hWndNewParent:=ParentHWnd)
    If Res = 0 Then
        '''''''''''''''''''''''''''
        ' an error occurred
        '''''''''''''''''''''''''''
        ErrNum = Err.LastDllError
        DisplayErrorText "Error With SetParent", ErrNum
    Else
        Debug.Print "The UserForm is a child of the Excel Application  _
        (Class XLMAIN). Note that the form will be visible even" & _
        " as you open and close windows, or minimize windows. If you  _
        restore the Excel window and move it around on the" & _
        " screen, the form will move with the Application window."
    End If
    SetForegroundWindow UserFormHWnd
    Form.Repaint
End Sub
```

```
Public Sub MakeFormChildOfNothing(Form As Object)
    '''''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' MakeFormChildOfNothing
    ' Sets the parent of the form to 0&
    '''''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''''''''
    Dim Res As Long
    Dim ParentHWnd As Long
    Dim ChildHWnd As Long
    Dim ErrNum As Long
    '''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' Set Parent HWnd to ActiveWindowHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''''''''
    'ParentHWnd = 0&
    ParentHWnd = GetDesktopWindow()
    '''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' Set Child HWnd to UserFormHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''''''''
    ChildHWnd = UserFormHWnd
    '''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' Call SetParent to make ChildHWnd a child of
    ' ParentHWnd.
    '''''''''''''''''''''''''''''''''''''''''''''''''''''
    Res = SetParent(hWndChild:=ChildHWnd, hWndNewParent:=ParentHWnd)
    'Res = SetWindowLong(hwnd:=ChildHWn
    'd, nIndex:=GWL_HWNDPARENT, dwNewLong:=0&)
    If Res = 0 Then
        '''''''''''''''''''''''''''''
        ' an error occurred
        '''''''''''''''''''''''''''''
        ErrNum = Err.LastDllError
        DisplayErrorText "Error With SetParent", ErrNum
    Else
        Debug.Print "The UserForm is a child of the Windows Desktop (see  _
        the GA_PARENT item above -- it has the same" & _
        " window handle as the Windows Desktop). The Parent Window shows  _
        as XLMAIN because XLMAIN is the " & _
        " owner of the window (see the GA_ROOTOWNER item above.)  Note  _
        that you can move the Excel Application" & _
        " window and the form will remain at its original location on the  _
        screen. You can also move the" & _
        " form outside of the Application's main window. This is the  _
        default behavior of an Excel Userform."
    End If
    SetForegroundWindow UserFormHWnd
    Form.Repaint
End Sub

Public Function GetSystemErrorMessageText(ErrorNumber As Long) As String
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

```
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' GetSystemErrorMessageText
    '
    ' This function gets the system error message te
    'xt that corresponds to the error code returned by the
    ' GetLastError API function or the Err.LastDllErr
    'or property. It may be used ONLY for these error codes.
    ' These are NOT the error numbers returned by Err
    '.Number (for these errors, use Err.Description to get
    ' the description of the message).
    ' The error number MUST be the value re
    'turned by GetLastError or Err.LastDLLError.
    '
    ' In general, you should use Err.LastDllError rather th
    'an GetLastError because under some circumstances the value of
    ' GetLastError will be reset to 0 before the value is r
    'eturned to VB. Err.LastDllError will always reliably return
    ' the last error number raised in a DLL.
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Dim ErrorText As String
    Dim TextLen As Long
    Dim FormatMessageResult As Long
    Dim LangID As Long
    LangID = 0&
    ErrorText = String$(FORMAT_MESSAGE_TEXT_LEN, " ")
    TextLen = Len(ErrorText)
    On Error Resume Next
    FormatMessageResult = 0&
    FormatMessageResult = FormatMessage( _
    dwFlags:=FORMAT_MESSAGE_FROM_SYSTEM Or FORMAT_MESSAGE_IGNORE_INSERTS,  _
    _
    lpSource:=0&, _
    dwMessageId:=ErrorNumber, _
    dwLanguageId:=0&, _
    lpBuffer:=ErrorText, _
    nSize:=TextLen, _
    Arguments:=0&)
    On Error GoTo 0
    If FormatMessageResult > 0 Then
        ErrorText = TrimToNull(ErrorText)
        GetSystemErrorMessageText = ErrorText
    Else
        ' Format message didn't return any text. th
        'ere is no text description for the specified error.
        GetSystemErrorMessageText = "NO ERROR DESCRIPTION AVAILABLE"
    End If
End Function

Public Sub DisplayErrorText(Context As String, ErrNum As Long)
    ''''''''''''''''''''''''''''''''''''''''''''''''
    ' Displays a standard error message box. For this
```

```
    ' procedure, ErrNum should be the number returned
    ' by the GetLastError API function or the value
    ' of Err.LastDllError. It is NOT the number
    ' returned by Err.Number.
    '''''''''''''''''''''''''''''''''''''''''''''''''
    Dim ErrText As String
    ErrText = GetSystemErrorMessageText(ErrNum)
    MsgBox Context & vbCrLf & _
    "Error Number: " & CStr(ErrNum) & vbCrLf & _
    "Error Text:   " & ErrText, vbOKOnly
End Sub

Public Function TrimToNull(Text As String) As String
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' TrimToNull
    ' Returns all the text in Text to the left of the vbNullChar
    ' character.
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Dim Pos As Integer
    Pos = InStr(1, Text, vbNullChar, vbTextCompare)
    If Pos > 0 Then
        TrimToNull = Left(Text, Pos - 1)
    Else
        TrimToNull = Text
    End If
End Function

Public Function GetWindowClassName(hWnd As Long) As String
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' GetWindowClassName
    ' Returns the window class name of the specified HWnd. Returns
    ' vbNullString if an error occurred.
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Dim ClassName As String
    Dim Length As Long
    Dim Res As Long
    If hWnd = 0 Then
        GetWindowClassName = "<none>"
        Exit Function
    End If
    ClassName = String$(MAX_PATH, vbNullChar)
    Length = Len(ClassName)
    Res = GetClassName(hWnd:=hWnd, lpClassName:=ClassName,  _
    nMaxCount:=Length)
    If Res = 0 Then
        DisplayErrorText Context:="Error Retrieiving Window Class for  _
        HWnd: " & CStr(hWnd), _
        ErrNum:=Err.LastDllError
        GetWindowClassName = vbNullString
    Else
        ClassName = TrimToNull(ClassName)
        GetWindowClassName = ClassName
```

```
    End If
End Function

Public Function GetHWndWindowText(hWnd As Long) As String
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' GetHWndWindowText
    ' This returns the WindowText of the HWnd.
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Dim txt As String
    Dim Res As Long
    Dim l As Long
    txt = String$(1024, vbNullChar)
    l = Len(txt)
    Res = GetWindowText(hWnd, txt, l)
    If Res Then
        txt = TrimToNull(txt)
        If txt = vbNullString Then
            txt = "<none>"
        End If
    Else
        txt = vbNullString
    End If
    GetHWndWindowText = txt
End Function

Public Function GetParentWindowClass(hWnd As Long) As String
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    ' GetParentWindowClass
    ' Returns the window class name of the parent window of HWnd.
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Dim ParentHWnd As Long
    Dim ClassName As String
    ParentHWnd = GetWindowLong(hWnd:=hWnd, nIndex:=GWL_HWNDPARENT)
    If ParentHWnd = 0 Then
        DisplayErrorText Context:="Error Retrieiving Parent Window for  _
        HWnd: " & CStr(hWnd) & _
        " Window Class: " & GetWindowClassName(hWnd), ErrNum:=Err. _
        LastDllError
        GetParentWindowClass = vbNullString
        Exit Function
    End If
    ClassName = GetWindowClassName(ParentHWnd)
    GetParentWindowClass = ClassName
End Function
```

--- Helper ---

```
#If VBA7 And Win64 Then
    Public Declare PtrSafe Sub Sleep Lib "kernel32" (ByVal dwMilliseconds  _
    As Long)
#Else
    public Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)
    #End If

    Public MouseFolder As String

    Rem mouse
    Public MouseArray() As Variant
    Rem declaration for keys event reading
    Public Declare PtrSafe Function GetAsyncKeyState Lib "User32" (ByVal  _
    vKey As Long) As Integer
    Rem declaration for mouse events
    Public Declare PtrSafe Sub mouse_event Lib "User32" (ByVal dwFlags As  _
    Long, ByVal dx As Long, ByVal dy As Long, ByVal cbuttons As Long,  _
    ByVal dwExtraInfo As Long)
    Public Const MOUSEEVENTF_LEFTDOWN = &H2
    Public Const MOUSEEVENTF_LEFTUP = &H4
    Public Const MOUSEEVENTF_MIDDLEDOWN = &H20
    Public Const MOUSEEVENTF_MIDDLEUP = &H40
    Public Const MOUSEEVENTF_RIGHTDOWN As Long = &H8
    Public Const MOUSEEVENTF_RIGHTUP As Long = &H10
    Public Const MOUSEEVENTF_MOVE = &H1
    Public Const MOUSEEVENTF_ABSOLUTE = &H8000
    Rem declaration for setting mouse position
    Public Declare PtrSafe Function SetCursorPos Lib "User32" (ByVal X As  _
    Long, ByVal Y As Long) As Long
    Rem declaration for getting mouse position
    Public Declare PtrSafe Function GetCursorPos Lib "User32" (lpPoint As  _
    POINTAPI) As Long
    Public Type POINTAPI
        X As Long
        Y As Long
    End Type

    Sub FollowLink(FolderPath As String)
        Dim oShell As Object
        Dim Wnd As Object
        Set oShell = CreateObject("Shell.Application")
        For Each Wnd In oShell.Windows
            If Wnd.Name = "File Explorer" Then
                If Wnd.Document.Folder.Self.path = FolderPath Then Exit  _
                Sub
            End If
        Next Wnd
        Application.ThisWorkbook.FollowHyperlink Address:=FolderPath,  _
        NewWindow:=True
    End Sub
```

```
Function InputboxString(Optional sTitle As String = "Select String",  _
Optional sPrompt As String = "Select String", Optional DefaultValue =  _
"") As String
    Dim stringVariable As String
    stringVariable = Application.InputBox( _
    Title:=sTitle, _
    Prompt:=sPrompt, _
    Type:=2, _
    Default:=DefaultValue)
    InputboxString = CStr(stringVariable)
End Function

Public Function CLIP(Optional StoreText As String) As String
    Dim X As Variant
    X = StoreText
  With CreateObject("htmlfile")
    With .parentWindow.clipboardData
      Select Case True
      Case Len(StoreText)
          .SetData "text", X
      Case Else
          CLIP = .GetData("text")
      End Select
    End With
  End With
End Function

Function IsFileFolderURL(path) As String
    Dim retval
    retval = "I"
    If (retval = "I") And FileExists(path) Then retval = "F"
    If (retval = "I") And FolderExists(path) Then retval = "D"
    If (retval = "I") And HttpExists(path) Then retval = "U"
    ' I => Invalid | F => File | D => Directory | U => Valid Url
    CheckPath = retval
End Function

Function FileExists(ByVal strFile As String, Optional bFindFolders As  _
Boolean) As Boolean
    'Purpose:   Return True if the file exists, even if it is hidden.
    'Arguments: strFile: File name to look fo
    'r. Current directory searched if no path included.
    '           bFindFolders. If strFile is a folder,
    'FileExists() returns False unless this argument is True.
    'Note:      Does not look inside subdirectories for the file.
    'Author:    Allen Browne. http://allenbrowne.com June, 2006.
    Dim lngAttributes As Long

    'Include read-only files, hidden files, system files.
    lngAttributes = (vbReadOnly Or vbHidden Or vbSystem)
  If bFindFolders Then
      lngAttributes = (lngAttributes Or vbDirectory)         _
```

```
            'Include folders as well.
        Else
            'Strip any trailing slash,
            'so Dir does not look inside the folder.
            Do While Right$(strFile, 1) = "\"
                strFile = Left$(strFile, Len(strFile) - 1)
            Loop
        End If
        'If Dir() returns something, the file exists.
        On Error Resume Next
        FileExists = (Len(Dir(strFile, lngAttributes)) > 0)
    End Function

    Function FolderExists(ByVal strPath As String) As Boolean
        On Error Resume Next
        FolderExists = ((GetAttr(strPath) And vbDirectory) = vbDirectory)
    End Function

    Function TrailingSlash(varIn As Variant) As String
        If Len(varIn) > 0 Then
            If Right(varIn, 1) = "\" Then
                TrailingSlash = varIn
            Else
                TrailingSlash = varIn & "\"
            End If
        End If
    End Function

    Function HttpExists(ByVal sURL As String) As Boolean
        Dim oXHTTP As Object
        Set oXHTTP = CreateObject("MSXML2.XMLHTTP")
        If Not UCase(sURL) Like "HTTP:*" Then
            sURL = "http://" & sURL
        End If
        On Error GoTo haveError
        oXHTTP.Open "HEAD", sURL, False
        oXHTTP.send
        HttpExists = IIf(oXHTTP.Status = 200, True, False)
        Exit Function
haveError:
        Debug.Print Err.Description
        HttpExists = False
    End Function

    Function ListboxSelectedCount(listboxCollection As Variant) As Long
        Dim i As Long
        Dim listItem As Long
        Dim selectedCollection As Collection
        Set selectedCollection = New Collection
        Dim listboxCount As Long
        'if arguement passed is collection of listboxes
        If TypeName(listboxCollection) = "Collection" Then
```

```
            For listboxCount = 1 To listboxCollection.Count
                If listboxCollection(listboxCount).ListCount > 0 Then
                    For listItem = 0 To listboxCollection(listboxCount). _
                    ListCount - 1
                        If listboxCollection(listboxCount). _
                        Selected(listItem) = True Then
                            SelectedCount = SelectedCount + 1
                        End If
                    Next listItem
                End If
            Next listboxCount
             'if arguement passed is single Listbox
        Else
            If listboxCollection.ListCount > 0 Then
                For i = 0 To listboxCollection.ListCount - 1
                    If listboxCollection.Selected(i) = True Then
                        SelectedCount = SelectedCount + 1
                    End If
                Next i
            End If
        End If
        ListboxSelectedCount = SelectedCount
    End Function

    Function ListboxSelectedIndexes(Lbox As MSForms.ListBox) As Collection
        'listboxes start at 0
        Dim i As Long
        Dim SelectedIndexes As Collection
        Set SelectedIndexes = New Collection
        If Lbox.ListCount > 0 Then
            For i = 0 To Lbox.ListCount - 1
                If Lbox.Selected(i) Then SelectedIndexes.Add i
            Next i
        End If
        Set ListboxSelectedIndexes = SelectedIndexes
    End Function
```

--- uMouseRecorder ---

```
Private Completed As Boolean

Function CursorPosition() As Variant
    Dim lngCurPos As POINTAPI, activeX As Long, activeY As Long
    GetCursorPos lngCurPos
    activeX = lngCurPos.X
    activeY = lngCurPos.Y
    Dim out(1) As Variant
    out(0) = activeX
    out(1) = activeY
    CursorPosition = out
End Function

Sub ShowCoordinates(X As Long, Y As Long)
    uCoordinates.Load
    uCoordinates.Left = X
    uCoordinates.Top = Y
    uCoordinates.Show
End Sub

Private Sub iLogLink_MouseDown(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
LogLink
LoadListbox
End Sub

Sub LogLink()
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim msg As String
    msg = InputboxString()
    If Len(msg) = 0 Then Exit Sub
    If msg = "False" Then Exit Sub
    If IsFileFolderURL(msg) = "I" Then Exit Sub
    Dim cell As Range
    Set cell = ws.Range("A" & Rows.Count).End(xlUp).Offset(1)
    cell = "go"
    cell.Offset(0, 1) = msg
End Sub

Private Sub iLogLink_MouseMove(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
infoLab.Caption = iLogLink.ControlTipText
End Sub

Private Sub iLogRight_MouseDown(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
LogClick "right"
LoadListbox
End Sub

Private Sub iCoordinates_MouseDown(ByVal Button As Integer, ByVal Shift  _
```

```
As Integer, ByVal X As Single, ByVal Y As Single)
IndexMouseLocation
End Sub

Private Sub iCoordinates_MouseMove(ByVal Button As Integer, ByVal Shift  _
As Integer, ByVal X As Single, ByVal Y As Single)
infoLab.Caption = iCoordinates.ControlTipText
End Sub

Private Sub iSize_MouseDown(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
If Me.Height > 125 Then
    Me.Height = 125
    iSize.SpecialEffect = fmSpecialEffectRaised
Else
    Me.Height = 275
    iSize.SpecialEffect = fmSpecialEffectSunken
    CreateListboxHeader lBoxData, lBoxHeader, Array("X", "Y", "L", "R",  _
    "NOTE")
End If
End Sub

Private Sub iSize_MouseMove(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
infoLab.Caption = iSize.ControlTipText
End Sub

Private Sub lBoxData_MouseMove(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
infoLab.Caption = lBoxData.ControlTipText
End Sub

Private Sub UserForm_Initialize()
Me.Height = 125
Me.Width = 230
LoadPosition Me
UserformOnTop Me
Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
MouseFolder = Environ("USERprofile") & "\Documents\vbArc\MouseMacro\"
If FolderExists(MouseFolder) = False Then FoldersCreate MouseFolder
checkFile
ClicksOnly.Value = ws.Range("h2")
LoadMRcaption
LoadListbox

CreateListboxHeader lBoxData, lBoxHeader, Array("X", "Y", "L", "R",  _
"NOTE")
End Sub

Private Sub UserForm_MouseMove(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
infoLab.Caption = "Hold ESC to STOP recording or playback"
```

```
End Sub

Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
SavePosition Me
End Sub

Private Sub ClicksOnly_Click()
Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
ws.Range("H2") = ClicksOnly
End Sub

Sub DeleteRows()
    If ListboxSelectedCount(uMouseRecorder.lBoxData) = 0 Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Dim c As Long
    c = ListboxSelectedCount(lBoxData)
    Set rng = ws.Range(ws.Cells(2 + R, 1), ws.Cells(2 + R, 5)).Resize(c)
    rng.Delete Shift:=xlUp
End Sub

Sub DoubleClick()
    'Double click as a quick series of two clicks
    mouse_event MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0
    mouse_event MOUSEEVENTF_LEFTUP, 0, 0, 0, 0
    mouse_event MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0
    mouse_event MOUSEEVENTF_LEFTUP, 0, 0, 0, 0
End Sub

Sub DuplicateRows()
    If ListboxSelectedCount(uMouseRecorder.lBoxData) = 0 Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Dim c As Long
    c = ListboxSelectedCount(lBoxData)
    Set rng = ws.Range(ws.Cells(2 + R, 1), ws.Cells(2 + R, 5)).Resize(c)
    Dim var
    var = rng.Value
    rng.Offset(rng.Rows.Count).Insert
    rng.Offset(rng.Rows.Count).Resize(rng.Rows.Count) = var
    Application.CutCopyMode = False
    LoadListbox
End Sub

Sub EditMemo()
    Dim s As String
    If ListboxSelectedCount(uMouseRecorder.lBoxData) = 0 Then Exit Sub
    Dim ans As String
```

```
    ans = InputboxString
    If ans = "False" Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Dim c As Long
    c = ListboxSelectedCount(lBoxData)
    Set rng = ws.Cells(2 + R, 5).Resize(c)
    rng.Value = ans
    LoadListbox
End Sub

Sub EditRow()
    Dim s As String
    If ListboxSelectedCount(uMouseRecorder.lBoxData) = 0 Then Exit Sub
    s = lBoxData.List(lBoxData.ListIndex, 0)
    s = s & "|" & lBoxData.List(lBoxData.ListIndex, 1)
    s = s & "|" & lBoxData.List(lBoxData.ListIndex, 2)
    s = s & "|" & lBoxData.List(lBoxData.ListIndex, 3)
    s = s & "|" & lBoxData.List(lBoxData.ListIndex, 4)
    Dim ans As String
    ans = InputboxString(, , s)
    If ans = "False" Then Exit Sub
    If UBound(Split(s, "|")) <> 4 Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Dim c As Long
    c = ListboxSelectedCount(lBoxData)
    Set rng = ws.Range(ws.Cells(2 + R, 1), ws.Cells(2 + R, 5)).Resize(c)
    rng.Value = (Split(ans, "|"))
    LoadListbox
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

Sub LeftClick()
    mouse_event MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0
    mouse_event MOUSEEVENTF_LEFTUP, 0, 0, 0, 0
End Sub

'Private Sub iMove_MouseDown(ByVal Button As Integer, ByV
'al Shift As Integer, ByVal X As Single, ByVal Y As Single)
'If Button = 1 Then
'    m_sngDownX = X
'    m_sngDownY = Y
'End If
```

```
'End Sub
'Private Sub iMove_MouseMove(ByVal Button As Integer, ByV
'al Shift As Integer, ByVal X As Single, ByVal Y As Single)
'If Button And 1 Then
'    Me.left = Me.left + (X - m_sngDownX)
'    Me.top = Me.top + (Y - m_sngDownY)
'End If
'End Sub
Sub LoadListbox()
    Dim rng As Range
    Set rng = RecordRange
    lBoxData.Clear
    If rng Is Nothing Then Exit Sub
    lBoxData.columnCount = rng.Columns.Count
    lBoxData.List = rng.Value
End Sub

Sub LoadMRcaption()
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim FileFullPath As String
    FileFullPath = RecordFileFullName
    If ws.Range("H1") = "" Then
        If ws.Range("A2") = "" Then
            Me.Caption = "New Recording"
        ElseIf ws.Range("A2") <> "" Then
            Me.Caption = "Existing Recording - NOT SAVED"
        End If
    ElseIf ws.Range("H1") <> "" Then
        Me.Caption = IIf(FileExists(FileFullPath), "Loaded - " & ws. _
        Range("H1"), "New Recording")
    End If
End Sub

Sub LoadRecord()
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim FName As String
    FName = PickRecord(MouseFolder)
    If FName = "" Or Right(FName, 7) <> "_mr.txt" Then
        infoLab.Caption = "No valid file selected"
        Exit Sub
    End If
    newRecord
    FName = Mid(FName, InStrRev(FName, "\") + 1)
    FName = Left(FName, InStr(1, FName, "_") - 1)
    uMouseRecorder.LoadedRecording.Caption = FName
    ws.Range("H1") = FName
    Dim recFile As String
    recFile = RecordFileFullName
    Dim arr
    arr = TXTtoArray(recFile)
    If IsEmpty(arr) Then Exit Sub
    Dim rng As Range
```

```
    Set rng = ws.Range("A2").CurrentRegion.Offset(1)
    rng.ClearContents
    rng.Resize(UBound(arr, 1), 4) = arr
End Sub

Private Sub LocMouse_Click()
PreviewMousePosition
End Sub

Sub LogAsk()
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim cell As Range
    Set cell = ws.Range("A" & Rows.Count).End(xlUp).Offset(1)
    Dim msg As String
    msg = InputboxString()
    If Len(msg) = 0 Then Exit Sub
    If msg = "False" Then Exit Sub
    cell = "ask"
    cell.Offset(0, 1) = msg
End Sub

Sub IndexMouseLocation()
    Dim lngCurPos As POINTAPI
    Dim activeX As Long, activeY As Long
    On Error GoTo LoopEnd
    Application.EnableCancelKey = xlErrorHandler
    Do
        GetCursorPos lngCurPos
        activeX = lngCurPos.X
        activeY = lngCurPos.Y
        LabX.Text = activeX
        LabY.Text = activeY
        Sleep 20
        DoEvents
    Loop
LoopEnd:
    Application.EnableCancelKey = xlInterrupt
End Sub

Sub LogClick(ClickType As String)
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Erase MouseArray
    Dim rng As Range
    Dim lngCurPos As POINTAPI
    Dim activeX As Long, activeY As Long
    On Error GoTo LoopEnd
    Application.EnableCancelKey = xlErrorHandler
    Do
        GetCursorPos lngCurPos
        activeX = lngCurPos.X
        activeY = lngCurPos.Y
        LabX.Text = activeX
```

```
        LabY.Text = activeY
        Sleep 20
        DoEvents
    Loop
LoopEnd:
    'If err = 18 Then
    Application.EnableCancelKey = xlInterrupt
    Set rng = ws.Range("A" & Rows.Count).End(xlUp).Offset(1, 0)
    Set rng = rng.Resize(, 5)
    rng.Value = Array(ClickType, activeX, activeY, "", "")

    infoLab.Caption = "Macro recorded."
    'End If
End Sub

Sub LogClickImmediate()
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Erase MouseArray
    Dim rng As Range
    Dim lngCurPos As POINTAPI
    Dim previousX As Long, previousY As Long, activeX As Long, activeY As  _
    Long
    Dim previousL As Long, previousR As Long, activeL As Long, activeR As  _
    Long
    Dim arrayCounter As Long: arrayCounter = 1
    On Error GoTo LoopEnd
    Application.EnableCancelKey = xlErrorHandler
    Dim counter As Long
    Do
        ReDim Preserve MouseArray(1 To arrayCounter)
        GetCursorPos lngCurPos
        activeL = IIf(GetAsyncKeyState(1) = 0, 0, 1)
        activeR = IIf(GetAsyncKeyState(2) = 0, 0, 1)
        activeX = lngCurPos.X
        activeY = lngCurPos.Y
        If previousL <> activeL Or previousR <> activeR Then
            previousX = activeX
            previousY = activeY
            previousL = activeL
            previousR = activeR
            MouseArray(arrayCounter) = Join(Array(previousX, previousY,  _
            activeL, activeR), ",")
            arrayCounter = arrayCounter + 1
            DoEvents
            counter = counter + 1
            If counter = 4 Then GoTo LoopEnd
        End If
    Loop
LoopEnd:
    If Err = 18 Then
        Application.EnableCancelKey = xlInterrupt
        Set rng = ws.Range("A" & Rows.Count).End(xlUp).Offset(1, 0)
```

```
        Set rng = rng.Resize(UBound(MouseArray), 1)
        rng = WorksheetFunction.Transpose(MouseArray)
        rng.TextToColumns rng, comma:=True
        Range(rng.Cells(1, 1), rng.Cells(2, 4)).Delete Shift:=xlUp
        'ws.Range("A3:D3").Delete Shift:=xlUp
        infoLab.Caption = "Macro recorded."
        '        infoLab.Caption = "Macro recorded at
        'rows: " & rng.Row & " to " & rng.Row + rng.Rows.Count
        Exit Sub
    End If
EH:
End Sub

Sub LogDoulbe()
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Erase MouseArray
    Dim rng As Range
    Dim lngCurPos As POINTAPI
    Dim activeX As Long, activeY As Long
    On Error GoTo LoopEnd
    Application.EnableCancelKey = xlErrorHandler
    Do
        GetCursorPos lngCurPos
        activeX = lngCurPos.X
        activeY = lngCurPos.Y
        LabX.Text = activeX
        LabY.Text = activeY
        Sleep 20
        DoEvents
    Loop
LoopEnd:
    If Err = 18 Then
        Application.EnableCancelKey = xlInterrupt
        Set rng = ws.Range("A" & Rows.Count).End(xlUp).Offset(1, 0)
        Set rng = rng.Resize(5)
        rng.Value = Array("double", activeX, activeY, "", "")
        infoLab.Caption = "Macro recorded."
        '        infoLab.Caption = "Macro recorded at
        'rows: " & rng.Row & " to " & rng.Row + rng.Rows.Count
        Exit Sub
    End If
EH:
End Sub

Sub LogText()
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim cell As Range
    Set cell = ws.Range("A" & Rows.Count).End(xlUp).Offset(1)
    Dim msg As String
    msg = InputboxString()
    If Len(msg) = 0 Then Exit Sub
    If msg = "False" Then Exit Sub
```

```
        cell = "sendkeys"
        cell.Offset(0, 1) = msg
    End Sub

    Sub MouseReplay(Optional rng As Range)
        Completed = False
        'ActiveWindow.WindowState = xlMaximized
        Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
        Dim cell As Range
        If rng Is Nothing Then
            Set rng = ws.Range("A2").CurrentRegion
            Set rng = rng.Offset(1).Resize(rng.Rows.Count - 1, 1)
        End If
        If WorksheetFunction.CountA(rng) = 0 Then Exit Sub
        On Error GoTo LoopEnd
        Application.EnableCancelKey = xlErrorHandler
        Dim DefaultSleep As Long
        DefaultSleep = 300
        Dim msg As String

        For Each cell In rng

            Rem if automatic record of clicks and motion
            If IsNumeric(cell) Then

                SetCursorPos cell, cell.Offset(, 1)

                If cell.Offset(0, 2) > 1 Then
                    dragMouse cell.Value, cell.Offset(0, 1), cell.Offset(0, 2) _
                    , cell.Offset(0, 3)
                ElseIf cell.Offset(0, 2) = 1 Then
                    If cell.Offset(1, 2) = 0 Then
                        '                       If cell
                        '.Offset(2, 2) = 0 And cell.Offset(-1, 2) = 0 Then
                        LeftClick
                        Set cell = cell.Offset(2, 0)
                        Rem This way doesn't work if logging clicks only an
                        'd not motion because two left clicks will be inter
                        'preted as double click
                        '                       ElseIf cell.Offset(2, 2) = 1 Then
                        '                           DoubleClick
                        '                           Set cell = cell.Offset(2, 0)
                        '                       End If
                    End If
                ElseIf cell.Offset(0, 3) = 1 Then
                    RightClick
                End If
            Else
                Rem if manual entry
                If cell = "wait" Then
                    Sleep IIf(cell.Offset(0, 1) <> "", cell.Offset(0, 1),  _
                    DefaultSleep)
```

```
            ElseIf cell = "go" Then
                msg = Replace(cell.Offset(0, 1), """", "")
                If IsFileFolderURL(msg) <> "I" Then
                    FollowLink msg
                    Sleep 500
                End If
            ElseIf cell = "move" Then
                SetCursorPos cell.Offset(0, 1), cell.Offset(0, 2)
            ElseIf cell = "left" Then
                SetCursorPos cell.Offset(0, 1), cell.Offset(0, 2)
                LeftClick
            ElseIf cell = "right" Then
                SetCursorPos cell.Offset(0, 1), cell.Offset(0, 2)
                RightClick
            ElseIf cell = "double" Then
                SetCursorPos cell.Offset(0, 1), cell.Offset(0, 2)
                DoubleClick
            ElseIf cell = "drag" Then
                dragMouse cell.Offset(0, 1), cell.Offset(0, 2), cell. _
                Offset(0, 3), cell.Offset(0, 4)
            ElseIf cell = "ask" Then
                msg = InputboxString(0, cell.Offset(0, 1))
                If Len(msg) > 0 Then
                    CLIP msg
                    SendKeys CLIP, True
                End If
            ElseIf cell = "sendkeys" Then
                Dim ClipText As String
                ClipText = IIf(cell.Offset(0, 2) = "", cell.Offset(0, 1),  _
                String(cell.Offset(0, 2), cell.Offset(0, 1)))
                CLIP ClipText
                SendKeys CLIP, True


            End If
        End If

            DoEvents
            Sleep 20        'DefaultSleep
            If Completed Then Exit Sub
        Next
    LoopEnd:
        '    If err = 18 Then
        Application.EnableCancelKey = xlInterrupt
        Do While GetAsyncKeyState(1) = 1
            mouse_event MOUSEEVENTF_LEFTUP, 0, 0, 0, 0
            DoEvents
        Loop
        Completed = True
        '    End If
    End Sub

    Sub MoveRows(offsetRows As Long)
```

```
    If ListboxSelectedCount(uMouseRecorder.lBoxData) = 0 Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Dim c As Long
    c = ListboxSelectedCount(lBoxData)
    Set rng = ws.Range(ws.Cells(2 + R, 1), ws.Cells(2 + R, 5)).Resize(c)
    On Error Resume Next        ' in case user makes unreasonable action  _
    like only 1 row exists and try to move it
    rng.Cut
    If 2 + R + offsetRows < 2 Then
        ws.Range("A2:E2").Insert
    ElseIf 2 + R + offsetRows > ws.Range("A1").CurrentRegion.Rows.Count  _
    Then
        Dim lRow As Long
        lRow = ws.Range("A1").CurrentRegion.Rows.Count
        ws.Range("A" & lRow).Resize(, 5).Insert
    Else
        rng.Offset(offsetRows).Insert
    End If
    Application.CutCopyMode = False
End Sub

Sub MoveToBottom()
    If lBoxData.ListIndex = -1 Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Dim c As Long
    c = ListboxSelectedCount(lBoxData)
    Set rng = ws.Range(ws.Cells(2 + R, 1), ws.Cells(2 + R, 5)).Resize(c)
    On Error Resume Next        ' in case user makes unreasonable action  _
    like only 1 row exists and try to move it
    rng.Cut
    Dim lRow As Long
    lRow = ws.Range("A1").CurrentRegion.Rows.Count + 1
    ws.Range("A" & lRow).Resize(, 5).Insert
    Application.CutCopyMode = False
End Sub

Sub MoveToTop()
    If lBoxData.ListIndex = -1 Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Dim c As Long
    c = ListboxSelectedCount(lBoxData)
    Set rng = ws.Range(ws.Cells(2 + R, 1), ws.Cells(2 + R, 5)).Resize(c)
    On Error Resume Next        ' in case user makes unreasonable action  _
```

```
    like only 1 row exists and try to move it
    rng.Cut
    ws.Range("A2").Resize(, 5).Insert
    Application.CutCopyMode = False
End Sub

Function PickRecord(Optional initFolder As String) As String
    If initFolder = "" Then initFolder = MouseFolder
    Dim strFile As String
    Dim fd As Office.FileDialog
    Set fd = Application.FileDialog(msoFileDialogFilePicker)
    With fd
        .Filters.Clear
        .Filters.Add "MouseRecord", "*.txt"
        .Title = "Choose Mouse Record"
        .AllowMultiSelect = False
        .InitialFileName = initFolder
        If .Show = True Then
            strFile = .SelectedItems(1)
            PickRecord = strFile
        End If
    End With
End Function

Sub PlayBackSelectedRows()
    If ListboxSelectedCount(uMouseRecorder.lBoxData) = 0 Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Dim c As Long
    c = ListboxSelectedCount(lBoxData)
    Set rng = ws.Cells(2 + R, 1).Resize(c)
    MouseReplay rng
End Sub

Sub PlayFromHere()
    If ListboxSelectedCount(uMouseRecorder.lBoxData) = 0 Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Dim c As Long
    c = lBoxData.ListCount - R
    Set rng = ws.Cells(2 + R, 1).Resize(c)
    MouseReplay rng
End Sub

Sub PlayUntilHere()
    If ListboxSelectedCount(uMouseRecorder.lBoxData) = 0 Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
```

```
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Set rng = ws.Cells(2, 1).Resize(R)
    MouseReplay rng
End Sub

Sub PreviewMousePosition()
    If lBoxData.ListIndex = -1 Then Exit Sub
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim R As Long
    R = ListboxSelectedIndexes(lBoxData)(1)
    Dim c As Long
    c = ListboxSelectedCount(lBoxData)
    Set rng = ws.Range(ws.Cells(2 + R, 1), ws.Cells(2 + R, 2))
    SetCursorPos rng.Cells(1, 1), rng.Cells(1, 2)
End Sub

Function RecordFileFullName() As String
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    RecordFileFullName = MouseFolder & ws.Range("H1") & "_mr.txt"
End Function

Function RecordRange() As Range
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    If ws.Range("A2") = "" Then Exit Function
    Dim rng As Range
    Set rng = ws.Range("A1").CurrentRegion
    Set rng = rng.Offset(1).Resize(rng.Rows.Count - 1, 5)
    Set RecordRange = rng
End Function

Sub RecordStart(Optional recordWholeMotion As Boolean)
    'ActiveWindow.WindowState = xlMaximized
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim lngCurPos As POINTAPI
    Dim previousX As Long, previousY As Long, activeX As Long, activeY As  _
    Long
    Dim previousL As Long, previousR As Long, activeL As Long, activeR As  _
    Long
    Erase MouseArray
    Dim arrayCounter As Long: arrayCounter = 1
    On Error GoTo LoopEnd
    Application.EnableCancelKey = xlErrorHandler
    Do
        ReDim Preserve MouseArray(1 To arrayCounter)
        GetCursorPos lngCurPos
        activeL = IIf(GetAsyncKeyState(1) = 0, 0, 1)
        activeR = IIf(GetAsyncKeyState(2) = 0, 0, 1)
        activeX = lngCurPos.X
        activeY = lngCurPos.Y
```

```
        If recordWholeMotion Then
            If previousX <> lngCurPos.X Or previousY <> lngCurPos.Y Or  _
            previousL <> activeL Or previousR <> activeR Then
                previousX = activeX
                previousY = activeY
                previousL = activeL
                previousR = activeR
                MouseArray(arrayCounter) = Join(Array(previousX,  _
                previousY, activeL, activeR), ",")
                arrayCounter = arrayCounter + 1
                DoEvents
            End If
        Else
            If previousL <> activeL Or previousR <> activeR Then
                previousX = activeX
                previousY = activeY
                previousL = activeL
                previousR = activeR
                MouseArray(arrayCounter) = Join(Array(previousX,  _
                previousY, activeL, activeR), ",")
                arrayCounter = arrayCounter + 1
                DoEvents
            End If
            LabX.Text = activeX
            LabY.Text = activeY
        End If
    Loop
LoopEnd:
    If Err = 18 Then
        Application.EnableCancelKey = xlInterrupt
        Set rng = ws.Range("A" & Rows.Count).End(xlUp).Offset(1, 0)
        Set rng = rng.Resize(UBound(MouseArray), 1)
        rng = WorksheetFunction.Transpose(MouseArray)
        rng.TextToColumns rng, comma:=True
        rng.Columns(1).Cells.Font.Bold = False
        rng.Cells.Font.Bold = True
        infoLab.Caption = "Macro recorded at rows: " & rng.Row & " to " &  _
        rng.Row + rng.Rows.Count - 3
        Range(rng.Cells(1, 1), rng.Cells(2, 4)).Delete Shift:=xlUp
        LoadMRcaption
    End If
End Sub

Function RecordedMacro() As String
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Set rng = ws.Range("A2").CurrentRegion.Offset(1)
    Set rng = rng.Resize(rng.Rows.Count - 1)
    Dim arr
    arr = rng.Value
    RecordedMacro = ArrayToString(arr)
End Function
```

```
Sub RightClick()
    'Right click
    mouse_event MOUSEEVENTF_RIGHTDOWN, 0, 0, 0, 0
    mouse_event MOUSEEVENTF_RIGHTUP, 0, 0, 0, 0
End Sub

Sub SaveRecord()
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Set rng = ws.Range("A2").CurrentRegion
    Set rng = rng.Offset(1).Resize(rng.Rows.Count - 1)
    OverwriteTxt RecordFileFullName, RecordedMacro
End Sub

Sub checkFile()
    Dim recFile As String
    recFile = RecordFileFullName
    Dim recFileName As String
    recFileName = IIf(FileExists(recFile) = True, recFile, "NONE")
    LoadedRecording.Caption = recFileName
    Me.LoadedRecording.ControlTipText = Mid(recFileName,  _
    InStrRev(recFileName, "\") + 1)
End Sub

Sub dragMouse(x0 As Long, y0 As Long, x1 As Long, y1 As Long)
    SetCursorPos x0, y0
    mouse_event MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0
    Sleep 20
    SetCursorPos x1, y1
    Sleep 20
    mouse_event MOUSEEVENTF_LEFTUP, 0, 0, 0, 0
End Sub

Private Sub iBottom_MouseDown(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
MoveToBottom
LoadListbox
End Sub

Private Sub iBottom_MouseMove(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
infoLab.Caption = iBottom.ControlTipText
End Sub

Private Sub iDelete_MouseDown(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
DeleteRows
LoadListbox
End Sub

Private Sub iDelete_MouseMove(ByVal Button As Integer, ByVal Shift As  _
```

```
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iDelete.ControlTipText
    End Sub

    Private Sub iDown_MouseDown(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    MoveRows 2
    LoadListbox
    End Sub

    Private Sub iDown_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iDown.ControlTipText
    End Sub

    Private Sub iDuplicate_MouseDown(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    DuplicateRows
    End Sub

    Private Sub iDuplicate_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iDuplicate.ControlTipText
    End Sub

    Private Sub iFolder_MouseDown(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    FollowLink MouseFolder
    End Sub

    Private Sub iFolder_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iFolder.ControlTipText
    End Sub

    Private Sub iLoadRecord_MouseDown(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    LoadRecord
    LoadedRecording.ControlTipText = LoadedRecording.Caption
    Dim s As String
    s = LoadedRecording.Caption
    s = Mid(s, InStrRev(s, "\") + 1)
    Me.Caption = s
    End Sub

    Private Sub iLoadRecord_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iLoadRecord.ControlTipText
    End Sub

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

```
    LogAsk
    LoadListbox
    End Sub

    Private Sub iLogAsk_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iLogAsk.ControlTipText
    End Sub

    Private Sub iLogClick_MouseDown(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    LogClick "left"
    LoadListbox
    End Sub

    Private Sub iLogClick_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iLogClick.ControlTipText
    End Sub

    Private Sub iLogDouble_MouseDown(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    LogClick "double"
    LoadListbox
    End Sub

    Private Sub iLogDouble_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iLogDouble.ControlTipText
    End Sub

    Private Sub iLogInput_MouseDown(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    LogText
    LoadListbox
    End Sub

    Private Sub iLogInput_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iLogInput.ControlTipText
    End Sub

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

    End Sub

    Private Sub iMemo_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iMemo.ControlTipText
    End Sub
```

```
    Private Sub iNewFile_MouseDown(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    newRecord
    LoadListbox
    End Sub

    Private Sub iNewFile_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iNewFile.ControlTipText
    End Sub

    Private Sub iPlayAll_MouseDown(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    Me.Hide
    MouseReplay
    Me.Show
    End Sub

    Private Sub iPlayAll_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iPlayAll.ControlTipText
    End Sub

    Private Sub iPlayFromHere_MouseDown(ByVal Button As Integer, ByVal Shift  _
    As Integer, ByVal X As Single, ByVal Y As Single)
    Me.Hide
    PlayFromHere
    Me.Show
    End Sub

    Private Sub iPlayFromHere_MouseMove(ByVal Button As Integer, ByVal Shift  _
    As Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iPlayFromHere.ControlTipText
    End Sub

    Private Sub iPlaySelection_MouseDown(ByVal Button As Integer, ByVal Shift  _
    As Integer, ByVal X As Single, ByVal Y As Single)
    PlayBackSelectedRows
    End Sub

    Private Sub iPlaySelection_MouseMove(ByVal Button As Integer, ByVal Shift  _
    As Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iPlaySelection.ControlTipText
    End Sub

    Private Sub iPlayUntilHere_MouseDown(ByVal Button As Integer, ByVal Shift  _
    As Integer, ByVal X As Single, ByVal Y As Single)
    Me.Hide
    PlayUntilHere
    Me.Show
    End Sub
```

```
Private Sub iPlayUntilHere_MouseMove(ByVal Button As Integer, ByVal Shift  _
As Integer, ByVal X As Single, ByVal Y As Single)
infoLab.Caption = iPlayUntilHere.ControlTipText
End Sub

Private Sub iRecClick_MouseDown(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
RecordStart ClicksOnly.Value
LoadListbox
End Sub

Private Sub iRecClick_MouseMove(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
infoLab.Caption = iRecClick.ControlTipText
End Sub

Private Sub iRecDrag_MouseDown(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
recordDrag
LoadListbox
End Sub

Private Sub iRecDrag_MouseMove(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
infoLab.Caption = iRecDrag.ControlTipText
End Sub

Private Sub iSave_MouseDown(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
If ws.Range("A2") = "" Then
    infoLab.Caption = "Record something first"
    Exit Sub
End If
Dim FName As String
FName = InputboxString(, , ws.Range("H1"))
If Len(FName) <> 0 And FName <> "False" Then
    ws.Range("H1") = FName
    SaveRecord
End If
LoadMRcaption
End Sub

Private Sub iSave_MouseMove(ByVal Button As Integer, ByVal Shift As  _
Integer, ByVal X As Single, ByVal Y As Single)
infoLab.Caption = iSave.ControlTipText
End Sub

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

```
    LoadListbox
    End Sub

    Private Sub iTop_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iTop.ControlTipText
    End Sub

    Private Sub iUp_MouseDown(ByVal Button As Integer, ByVal Shift As Integer, _
    ByVal X As Single, ByVal Y As Single)
    MoveRows -1
    LoadListbox
    End Sub

    Private Sub iUp_MouseMove(ByVal Button As Integer, ByVal Shift As Integer, _
    ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = iUp.ControlTipText
    End Sub

    Private Sub info_MouseMove(ByVal Button As Integer, ByVal Shift As  _
    Integer, ByVal X As Single, ByVal Y As Single)
    infoLab.Caption = info.ControlTipText
    End Sub

    Private Sub lBoxData_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
    EditRow
    End Sub

    Private Sub labHome_Click()
    labHome.SpecialEffect = fmSpecialEffectSunken
    labRec.SpecialEffect = fmSpecialEffectRaised
    fFile.Left = 30
    fPlay.Left = 30
    fRecord.Left = 255
    fLog.Left = 255
    End Sub

    Private Sub labRec_Click()
    labHome.SpecialEffect = fmSpecialEffectRaised
    labRec.SpecialEffect = fmSpecialEffectSunken
    fFile.Left = 255
    fPlay.Left = 255
    fRecord.Left = 30
    fLog.Left = 83
    End Sub

    Rem @NOT WORKING - forces motion top to bottom?
    Sub moveFromAtoB(x0 As Long, y0 As Long, x1 As Long, y1 As Long)
        Dim steep As Boolean: steep = Abs(y1 - y0) > Abs(x1 - x0)
        Dim t As Integer
        If steep Then
            '// swap(x0, y0);
```

```
        t = x0
        x0 = y0
        y0 = t
        ' // swap(x1, y1);
        t = x1
        x1 = y1
        y1 = t
    End If
    If x0 > x1 Then
        '// swap(x0, x1);
        t = x0
        x0 = x1
        x1 = t
        '// swap(y0, y1);
        t = y0
        y0 = y1
        y1 = t
    End If
    Dim deltax As Integer: deltax = x1 - x0
    Dim deltay As Integer: deltay = Abs(y1 - y0)
    Dim deviation As Integer: deviation = deltax / 2
    Dim ystep As Integer
    Dim Y  As Integer: Y = y0
    If y0 < y1 Then
        ystep = 1
    Else
        ystep = -1
    End If
    Dim X As Integer
    For X = x0 To x1 Step ystep
        If steep Then
            SetCursorPos Y, X
        Else
            SetCursorPos X, Y
        End If
        deviation = deviation - deltay
        If deviation < 0 Then
            Y = Y + ystep
            deviation = deviation + deltax
        End If
        DoEvents
        Sleep 1
    Next
End Sub

Sub newRecord()
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    ws.Range("A2").CurrentRegion.Offset(1).ClearContents
    ws.Range("R7").CurrentRegion.Offset(1).ClearContents
    ws.Range("H1").ClearContents
    LoadMRcaption
End Sub
```

```
Sub recordDrag()
    Dim ws As Worksheet: Set ws = ThisWorkbook.Sheets("uMouseRecorder")
    Dim rng As Range
    Dim lngCurPos As POINTAPI
    Dim previousX As Long, previousY As Long, activeX As Long, activeY As  _
    Long
    Dim previousL As Long, previousR As Long, activeL As Long, activeR As  _
    Long
    Erase MouseArray
    Dim arrayCounter As Long: arrayCounter = 1
    On Error GoTo LoopEnd
    Application.EnableCancelKey = xlErrorHandler
    Do
        ReDim Preserve MouseArray(1 To arrayCounter)
        GetCursorPos lngCurPos
        activeL = IIf(GetAsyncKeyState(1) = 0, 0, 1)
        activeR = IIf(GetAsyncKeyState(2) = 0, 0, 1)
        activeX = lngCurPos.X
        activeY = lngCurPos.Y
        If previousX <> lngCurPos.X Or previousY <> lngCurPos.Y Or  _
        previousL <> activeL Or previousR <> activeR Then
            previousX = activeX
            previousY = activeY
            previousL = activeL
            previousR = activeR
            MouseArray(arrayCounter) = Join(Array(previousX, previousY,  _
            activeL, activeR), ",")
            arrayCounter = arrayCounter + 1
        End If
    Loop
LoopEnd:
    If Err = 18 Then
        Application.EnableCancelKey = xlInterrupt
        Dim arr
        arr = MouseArray
        arr = Filter(arr, ",1,", , vbTextCompare)
        Set rng = ws.Range("A" & Rows.Count).End(xlUp).Offset(1, 0)
        'Set rng = rng.Resize(UBound(arr), 1)
        rng = WorksheetFunction.Transpose(arr)
        rng.TextToColumns rng, comma:=True
        rng.Offset(0, 4) = "DRAG"
        rng.Offset(2).Resize(rng.Rows.Count - 3, 5).Delete xlUp
        rng.Resize(1, 5).Delete xlUp
        rng.Offset(0, 2).Resize(1, 2).Value = rng.Offset(1).Resize(1, 2). _
        Value
        rng.Offset(1).Resize(1, 5).Delete xlUp
        infoLab.Caption = "Drag recorded."
        '        infoLab.Caption = "Drag recorded at
        'rows: " & rng.Row & " to " & rng.Row + rng.Rows.Count
    End If
End Sub
```

```
    ' Enum MouseButtonConstants
    ' vbLeftButton
    ' vbMiddleButton
    ' vbRightButton
    ' End Enum
    '
    ''simulate the MouseDown event
    ' Sub ButtonDown(Optional ByVal Button As MouseButtonConstants = _
    '    vbLeftButton)
    '    Dim lFlag As Long
    '    If Button = vbLeftButton Then
    '        lFlag = MOUSEEVENTF_LEFTDOWN
    '    ElseIf Button = vbMiddleButton Then
    '        lFlag = MOUSEEVENTF_MIDDLEDOWN
    '    ElseIf Button = vbRightButton Then
    '        lFlag = MOUSEEVENTF_RIGHTDOWN
    '    End If
    '    mouse_event lFlag, 0, 0, 0, 0
    'End Sub
    '
    ''simulate the MouseUp event
    '
    ' Sub ButtonUp(Optional ByVal Button As MouseButtonConstants = _
    '    vbLeftButton)
    '    Dim lFlag As Long
    '    If Button = vbLeftButton Then
    '        lFlag = MOUSEEVENTF_LEFTUP
    '    ElseIf Button = vbMiddleButton Then
    '        lFlag = MOUSEEVENTF_MIDDLEUP
    '    ElseIf Button = vbRightButton Then
    '        lFlag = MOUSEEVENTF_RIGHTUP
    '    End If
    '    mouse_event lFlag, 0, 0, 0, 0
    'End Sub
    '
    ''simulate the MouseClick event
    '
    ' Sub ButtonClick(Optional ByVal Button As MouseButtonConstants = _
    '    vbLeftButton)
    '    ButtonDown Button
    '    ButtonUp Button
    'End Sub
    '
    ''simulate the MouseDblClick event
    '
    ' Sub ButtonDblClick(Optional ByVal Button As MouseButtonConstants = _
    '    vbLeftButton)
    '    ButtonClick Button
    '    ButtonClick Button
    'End Sub
```

```
'Sub AlternativeLogPlayback()
'Rem from different logging style
'Dim DefaultSleep As Long
'DefaultSleep = 1000
'Dim cell As Range, rng As Range
'Set rng = ActiveSheet.Range("A1").CurrentRegion
'Set rng = rng.Resize(, 1).offset(1).Resize(rng.rows.count - 1)
'    Dim lngCurPos As POINTAPI, activeX As Long, activeY As Long
'    GetCursorPos lngCurPos
'    activeX = lngCurPos.x
'    activeY = lngCurPos.y
'For Each cell In rng
'    If cell <> "drag" Then
'        If cell.offset(0, 1) <> "" And cell.offset(0, 2) <> "" Then
'            'moveFromAtoB activeX, activeY, CLng
'(cell.offset(0, 1)), CLng(cell.offset(0, 2).Value)
'            SetCursorPos cell.offset(0, 1), cell.offset(0, 2)
'        End If
'   End If
'    If cell = "move" Then
'        'moveFromAtoB activeX, activeY, CLng(c
'ell.offset(0, 1)), CLng(cell.offset(0, 2).Value)
'       SetCursorPos cell.offset(0, 1), cell.offset(0, 2)
'    ElseIf cell = "left" Then LeftClick
'    ElseIf cell = "double" Then DoubleClick
'    ElseIf cell = "right" Then RightClick
'    ElseIf cell = "drag" Then
'        GetCursorPos lngCurPos
'        activeX = lngCurPos.x
'        activeY = lngCurPos.y
'        dragMouse activeX, activeY, cell.offset(0, 1), cell.offset(0, 2)
'    ElseIf cell = "type" Then
'        CLIP cell.offset(0, 1)
'        SendKeys CLIP, True
'    ElseIf cell = "ask" Then
'        Dim msg As String
'        msg = InputboxString()
'        CLIP msg
'        SendKeys CLIP, True
'    End If
'    If cell = "wait" Then
'        Sleep IIf(cell.offset(0, 1)
'<> "", cell.offset(0, 1), DefaultSleep)
'    Else
'        Sleep DefaultSleep
'    End If
'    DoEvents
'Next
'End Sub
```

```
Function FolderExists(ByVal strPath As String) As Boolean
    On Error Resume Next
    FolderExists = ((GetAttr(strPath) And vbDirectory) = vbDirectory)
    On Error GoTo 0
End Function

Sub FoldersCreate(FolderPath As String)
    Dim individualFolders() As String
    Dim tempFolderPath As String
    Dim arrayElement As Variant
    individualFolders = Split(FolderPath, "\")
    For Each arrayElement In individualFolders
        tempFolderPath = tempFolderPath & arrayElement & "\"
        If FolderExists(tempFolderPath) = False Then
            MkDir tempFolderPath
        End If
    Next arrayElement
End Sub

Sub SavePosition(Form As Object)
    SaveSetting "My Settings Folder", Form.Name, "Left Position", Form. _
    Left
    SaveSetting "My Settings Folder", Form.Name, "Top Position", Form.Top
End Sub

Sub LoadPosition(Form As Object)
    If GetSetting("My Settings Folder", Form.Name, "Left Position") = "" _
    And GetSetting("My Settings Folder", Form.Name, "Top Position") = ""  _
    Then
    Form.StartUpPosition = 1
Else
    Form.Left = GetSetting("My Settings Folder", Form.Name, "Left  _
    Position")
    Form.Top = GetSetting("My Settings Folder", Form.Name, "Top Position")
End If
End Sub

Sub CreateListboxHeader(body As MSForms.ListBox, header As MSForms. _
ListBox, arrHeaders)

    header.Width = body.Width
    Dim i As Long

    'must have a listbox to use as headers

    'header.columnCount = body.columnCount
    If header.ColumnWidths <> body.ColumnWidths And body.ColumnWidths <>  _
    "" Then
        header.ColumnWidths = body.ColumnWidths
    End If
    'add headerelements
    header.Clear
```

```
        header.AddItem

    If ArrayDimensions(arrHeaders) = 1 Then
        For i = 0 To UBound(arrHeaders)
            'make it prety
            header.List(0, i) = arrHeaders(i)
        Next i
    Else
        For i = 1 To UBound(arrHeaders, 2)
            header.List(0, i - 1) = arrHeaders(1, i)
        Next i
    End If
    body.ZOrder (1)
    header.ZOrder (0)
    header.SpecialEffect = fmSpecialEffectFlat
    header.BackColor = RGB(200, 200, 200)
    'align header to body
    header.Height = 15
    header.Width = body.Width
    header.Left = body.Left
    header.Top = body.Top - header.Height - 1
    header.Font.Bold = True
    header.Font.Name = "Comic Sans MS"
    header.Font.Size = 9
    '    header.ForeColor =
    '    header.BackColor =
End Sub

Public Function FileExists(ByVal FileName As String) As Boolean
    If InStr(1, FileName, "\") = 0 Then Exit Function
    If Right(FileName, 1) = "\" Then FileName = Left(FileName,  _
    Len(FileName) - 1)
    FileExists = (Dir(FileName, vbArchive + vbHidden + vbReadOnly +  _
    vbSystem) <> "")
End Function

Function ArrayDimensions(ByVal vArray As Variant) As Long
    Dim dimnum As Long
    On Error GoTo FinalDimension
    For dimnum = 1 To 60000
        ErrorCheck = LBound(vArray, dimnum)
    Next
FinalDimension:
    ArrayDimensions = dimnum - 1
End Function
```

```
--- 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")
```

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