

Get Open File Name dialog. When user clicks cancel.

Get Open File Name dialog. When user clicks cancel.

Source:

<http://www.tech-archive.net/Archive/Office/microsoft.public.office.developer.vba/2007-12/msg00038.html>

- *From:* DZ <DZ@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 24 Dec 2007 11:56:00 -0800
-

How can I take an action (Example: Display a message box), if the user clicks Cancel in the Open Dialog.

I got this API to work, but I know very little about how to manipulate it. Some sample code and an indication of where to place the code would be very appreciated.

As it stands now, when the user clicks cancel, the return value is the first file in My Documents. I would like have it return nothing if the user clicks cancel, then display a message box.

Here is API as I am currently using it:

```
'=====
Private Type tagOPENFILENAME
  lStructSize As Long
  hwndOwner As Long
  hInstance As Long
  strFilter As String
  strCustomFilter As String
  nMaxCustFilter As Long
  nFilterIndex As Long
  strFile As String
  nMaxFile As Long
  strFileName As String
  nMaxFileName As Long
  strInitialDir As String
  strTitle As String
  Flags As Long
  nFileOffset As Integer
  nFileExtension As Integer
  strDefExt As String
  lCustData As Long
  lpfnHook As Long
  lpTemplateName As String
End Type
```

Get Open File Name dialog. When user clicks cancel.

```
Private Declare Function aht_apiGetOpenFileName Lib "comdlg32.dll" _  
Alias "GetOpenFileNameA" (OFN As tagOPENFILENAME) As Boolean
```

```
Private Declare Function aht_apiGetSaveFileName Lib "comdlg32.dll" _  
Alias "GetSaveFileNameA" (OFN As tagOPENFILENAME) As Boolean  
Private Declare Function CommDlgExtendedError Lib "comdlg32.dll" () As Long
```

```
Const ahtOFN_READONLY = &H1  
Const ahtOFN_OVERWRITEPROMPT = &H2  
Const ahtOFN_HIDEREADONLY = &H4  
Const ahtOFN_NOCHANGEDIR = &H8  
Const ahtOFN_SHOWHELP = &H10  
Const CSIDL_DESKTOPDIRECTORY = &H10  
' You won't use these.  
' Const ahtOFN_ENABLEHOOK = &H20  
' Const ahtOFN_ENABLETEMPLATE = &H40  
' Const ahtOFN_ENABLETEMPLATEHANDLE = &H80  
Const ahtOFN_NOVALIDATE = &H100  
Const ahtOFN_ALLOWMULTISELECT = &H200  
Const ahtOFN_EXTENSIONDIFFERENT = &H400  
Const ahtOFN_PATHMUSTEXIST = &H800  
Const ahtOFN_FILEMUSTEXIST = &H1000  
Const ahtOFN_CREATEPROMPT = &H2000  
Const ahtOFN_SHAREAWARE = &H4000  
Const ahtOFN_NOREADONLYRETURN = &H8000  
Const ahtOFN_NOTESTFILECREATE = &H10000  
Const ahtOFN_NONETWORKBUTTON = &H20000  
Const ahtOFN_NOLONGNAMES = &H40000  
' New for Windows 95  
Const ahtOFN_EXPLORER = &H80000  
Const ahtOFN_NODEREFERENCELINKS = &H100000  
Const ahtOFN_LONGNAMES = &H200000
```

```
Function GetOpenFile(Optional varDirectory As Variant, _  
Optional varTitleForDialog As Variant) As Variant  
' Here's an example that gets an Access database name.  
Dim strFilter As String  
Dim lngFlags As Long  
Dim varFileName As Variant  
' Specify that the chosen file must already exist,  
' don't change directories when you're done  
' Also, don't bother displaying  
' the read-only box. It'll only confuse people.  
lngFlags = ahtOFN_FILEMUSTEXIST Or _  
ahtOFN_HIDEREADONLY Or ahtOFN_NOCHANGEDIR  
If IsMissing(varDirectory) Then  
varDirectory = ""  
End If  
If IsMissing(varTitleForDialog) Then  
varTitleForDialog = ""
```

Get Open File Name dialog. When user clicks cancel.

Get Open File Name dialog. When user clicks cancel.

End If

```
' Define the filter string and allocate space in the "c"  
' string Duplicate this line with changes as necessary for  
' more file templates.
```

```
strFilter = ahtAddFilterItem(strFilter, _  
"Access (*.mdb)", "*.MDB;*.MDA")  
' Now actually call to get the file name.  
varFileName = ahtCommonFileOpenSave( _  
OpenFile:=True, _  
InitialDir:=varDirectory, _  
Filter:=strFilter, _  
Flags:=lngFlags, _  
DialogTitle:=varTitleForDialog)  
If Not IsNull(varFileName) Then  
varFileName = TrimNull(varFileName)  
Else  
'varFileName = ""
```

End If

GetOpenFile = varFileName

End Function

```
Function ahtCommonFileOpenSave( _  
Optional ByRef Flags As Variant, _  
Optional ByVal InitialDir As Variant, _  
Optional ByVal Filter As Variant, _  
Optional ByVal FilterIndex As Variant, _  
Optional ByVal DefaultExt As Variant, _  
Optional ByVal FileName As Variant, _  
Optional ByVal DialogTitle As Variant, _  
Optional ByVal Hwnd As Variant, _  
Optional ByVal OpenFile As Variant) As Variant
```

```
' This is the entry point you'll use to call the common  
' file open/save dialog. The parameters are listed  
' below, and all are optional.  
,
```

' In:

' Flags: one or more of the ahtOFN_* constants, OR'd together.

' InitialDir: the directory in which to first look

' Filter: a set of file filters, set up by calling

' AddFilterItem. See examples.

' FilterIndex: 1-based integer indicating which filter

' set to use, by default (1 if unspecified)

' DefaultExt: Extension to use if the user doesn't enter one.

' Only useful on file saves.

' FileName: Default value for the file name text box.

' DialogTitle: Title for the dialog.

' hwnd: parent window handle

' OpenFile: Boolean(True=Open File/False=Save As)

' Out:

Get Open File Name dialog. When user clicks cancel.

Get Open File Name dialog. When user clicks cancel.

```
' Return Value: Either Null or the selected filename
Dim OFN As tagOPENFILENAME
Dim strFileName As String
Dim strFileTitle As String
Dim fResult As Boolean
' Give the dialog a caption title.
If IsMissing(InitialDir) Then InitialDir = CurDir
If IsMissing(Filter) Then Filter = ""
If IsMissing(FilterIndex) Then FilterIndex = 1
If IsMissing(Flags) Then Flags = 0&
If IsMissing(DefaultExt) Then DefaultExt = ""
If IsMissing(FileName) Then FileName = ""
If IsMissing(DialogTitle) Then DialogTitle = ""
If IsMissing(Hwnd) Then Hwnd = application.hWndAccessApp
If IsMissing(OpenFile) Then OpenFile = True
' Allocate string space for the returned strings.
strFileName = left(FileName & String(256, 0), 256)
strFileTitle = String(256, 0)
' Set up the data structure before you call the function
With OFN
.lStructSize = Len(OFN)
.hwndOwner = Hwnd
.strFilter = Filter
.nFilterIndex = FilterIndex
.strFile = strFileName
.nMaxFile = Len(strFileName)
.strFileTitle = strFileTitle
.nMaxFileTitle = Len(strFileTitle)
.strTitle = DialogTitle
.Flags = Flags
.strDefExt = DefaultExt
.strInitialDir = InitialDir
' Didn't think most people would want to deal with
' these options.
.hInstance = 0
.strCustomFilter = ""
.nMaxCustFilter = 0
.lpfHook = 0
'New for NT 4.0
.strCustomFilter = String(255, 0)
.nMaxCustFilter = 255
End With
' This will pass the desired data structure to the
' Windows API, which will in turn it uses to display
' the Open/Save As Dialog.
If OpenFile Then
fResult = aht_apiGetOpenFileName(OFN)
Else
fResult = aht_apiGetSaveFileName(OFN)
End If
```

Get Open File Name dialog. When user clicks cancel.

Get Open File Name dialog. When user clicks cancel.

```
' The function call filled in the strFileTitle member  
' of the structure. You'll have to write special code  
' to retrieve that if you're interested.
```

```
If fResult Then
```

```
' You might care to check the Flags member of the  
' structure to get information about the chosen file.
```

```
' In this example, if you bothered to pass in a  
' value for Flags, we'll fill it in with the outgoing  
' Flags value.
```

```
If Not IsMissing(Flags) Then Flags = OFN.Flags  
ahtCommonFileOpenSave = TrimNull(OFN.strFile)
```

```
Else
```

```
ahtCommonFileOpenSave = vbNullString
```

```
End If
```

```
End Function
```

```
Function ahtAddFilterItem(strFilter As String, _  
strDescription As String, Optional varItem As Variant) As String
```

```
' Tack a new chunk onto the file filter.
```

```
' That is, take the old value, stick onto it the description,
```

```
' (like "Databases"), a null character, the skeleton
```

```
' (like "*.mdb;*.mda") and a final null character.
```

```
If IsMissing(varItem) Then varItem = "*.*)"
```

```
ahtAddFilterItem = strFilter & _
```

```
strDescription & vbNullChar & _
```

```
varItem & vbNullChar
```

```
End Function
```

```
Private Function TrimNull(ByVal strItem As String) As String
```

```
Dim intPos As Integer
```

```
intPos = InStr(strItem, vbNullChar)
```

```
If intPos > 0 Then
```

```
TrimNull = left(strItem, intPos - 1)
```

```
Else
```

```
TrimNull = strItem
```

```
End If
```

```
End Function
```

```
Function OpenDLG()
```

```
Dim strFilter As String
```

```
Dim lngFlags As Long
```

```
Dim strFileNameOnly As String
```

```
strFilter = ahtAddFilterItem(strFilter, "Excel Files (*.XLS)", "*.XLS")
```

```
' "C:\
```

```
'CSIDL_DESKTOPDIRECTORY
```

```
' Environ("UserProfile") & "\Desktop"
```

```
'The full path is placed in Module level variable so the full path can be  
passed to the
```

```
'Excel automation procedure and the filename only to the text on the form
```

```
strFullPathOfExcelFile =
```

Get Open File Name dialog. When user clicks cancel.

Get Open File Name dialog. When user clicks cancel.

```
ahCommonFileOpenSave(InitialDir:=Environ("UserProfile") & "\\Desktop", _  
Filter:=strFilter, FilterIndex:=3, Flags:=lngFlags, _  
DialogTitle:="Select an Excel file and click 'Open'")
```

```
' Since you passed in a variable for lngFlags,  
' the function places the output flags value in the variable.  
Debug.Print Hex(lngFlags)
```

```
'These are the text boxes on the form  
strFileNameOnly = RturnFileName(strFullPathOfExcelFile)
```

```
Me.txtFileName = strFileNameOnly
```

```
Me.txtOutPutFileName = left(strFileNameOnly, Len(strFileNameOnly) - 4) &  
"(CONSLD).XLS"
```

```
End Function
```

```
***** Code End *****
```

.