

Re: Sending debugging messages to a second document.

Re: Sending debugging messages to a second document.

Source:

<http://www.tech-archive.net/Archive/Word/microsoft.public.word.vba.general/2007-11/msg00117.html>

- *From:* Russ <drsNOSPAMmikle@xxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Wed, 07 Nov 2007 05:49:34 -0500
-

Barry,

Not sure what you mean by macro hangs?(infinite loop?)

Here is another method to create log files by Peter Hewett.

Here is code amended to work on both MacWord and WinWord.
I used Application.PathSeparator and changed the Kill() function to KillLog() to avoid an endless loop in the class.

Insert a new class and name it LogIt then paste this Class code into it.
Peter Hewett's amended code:

***** Start of LogIt Class code *****

```
Private mstrFile As String
Private mstrPath As String
Private mboolEnabled As Boolean
Private mboolTimeStamp As Boolean

Public Sub Output(ByVal strText As String)
Const cProcedureName As String = "Log::Output"

Dim hFile As Long

On Error GoTo OutputError

' Everything must be right for us to log the passed text
If CanOutput Then

' Prefix log text with timestamp
If TimeStamp Then strText = Now & vbTab & strText

' Open log file for append
hFile = FreeFile
Open mstrPath & mstrFile For Append Access Write As hFile
```

Re: Sending debugging messages to a second document.

Re: Sending debugging messages to a second document.

```
' Write the output and tidy up  
Print #hFile, strText  
Close hFile  
End If
```

```
OutputExit:  
Exit Sub
```

```
OutputError:  
mboolEnabled = False  
Err.Raise Err.Number, cProcedureName, Err.Description  
End Sub ' Output
```

```
Public Sub KillLog()  
On Error GoTo HandleError
```

```
' Delete the current log file if it exists  
If Len(Dir$(mstrPath & mstrFile)) > 0 Then  
Kill mstrPath & mstrFile  
End If
```

```
ExitHere:  
Exit Sub
```

```
HandleError:  
mboolEnabled = False  
Err.Raise vbObjectError + 8001, "Log::Kill", _  
Err.Description  
Resume ExitHere  
End Sub ' Reset
```

```
Public Property Get Path() As String  
Path = mstrPath  
End Property  
Public Property Let Path(ByVal strPath As String)  
strPath = Trim$(strPath)  
If Right$(strPath, 1) <> Application.PathSeparator _  
Then strPath = strPath & Application.PathSeparator  
mstrPath = Trim$(strPath)  
End Property
```

```
Public Property Get File() As String  
File = mstrFile  
End Property  
Public Property Let File(ByVal FileName As String)  
mstrFile = FileName  
End Property
```

```
Public Property Get Enabled() As Boolean  
Enabled = mboolEnabled
```

Re: Sending debugging messages to a second document.

Re: Sending debugging messages to a second document.

```
End Property
Public Property Let Enabled(ByVal EnableLogFile As Boolean)
mboolEnabled = EnableLogFile
End Property

Public Property Get TimeStamp() As Boolean
TimeStamp = mboolTimeStamp
End Property
Public Property Let TimeStamp(ByVal TimeStampOutput As Boolean)
mboolTimeStamp = TimeStampOutput
End Property

Public Property Get CanOutput() As Boolean
CanOutput = LenB(Path) > 0 And LenB(File) > 0 And Enabled
End Property

Private Sub Class_Initialize()
Me.Enabled = True
End Sub
```

***** End of LogIt Class code *****

```
Public Sub LogItTest()
Dim logFile As LogIt

' Create log file
Set logFile = New LogIt
logFile.Path = "Macintosh HD:Users:UserName:Desktop:"
logFile.File = "DEBUG LogItTest.txt"

' Use it...
logFile.KillLog "To in effect, clear log, if desired.
logFile.TimeStamp = True
logFile.Output "This is the first log entry"
logFile.TimeStamp = False
logFile.Output "This is the second log entry "
Set logFile = Nothing
End Sub
```

Multiple logfiles could be going at the same time to log only certain parts of code. You just need to create multiple instances of each Logit class.

Windows users could also use environmental variables:

```
logFile.Path = Environ("HOMEDRIVE") & Environ("HOMEPATH") & "\\My Documents\\"
```

Re: Sending debugging messages to a second document.

Re: Sending debugging messages to a second document.

Jay:

With some minor modification, your example code is working, up to a point. Apparently the OS or Word or something is buffering the writes to the log file. By the time the macro hangs, only a small portion of the trace statements have been written out; the rest are lost when the hang occurs.

Can you tell me how to force immediate writes to the log file, so that I don't lose state of the macro at the time of the error?

Thank you for your help.

--

Barry Carroll

(Cleverly disguised as a trained SW engineer.)

Datalogic Scanning, Inc. assumes no responsibility whatsoever for any statements made by me. I'm entirely on my own here.

"Jay Freedman" wrote:

Barry wrote:

I am debugging a VBA subroutine that hangs under certain conditions. In order to pinpoint the spot in the code where the hang occurs, I want to write trace data to a second document, where they will not interfere with the text and ActiveX controls being added to the target document. I know that this is pretty elementary VBA, but I've never done it before and can't seem to make it work.

Can someone show me or point me to an example of how this is done?

Thanks in advance for any help.

See this thread from yesterday:

http://groups.google.com/group/microsoft.public.word.vba.beginners/browse_thread/thread/c0e61d07da657f08/2d848ae91240c9ea?hl=en&lnk=st&q=#2d848ae91240c9ea

--

Regards,

Jay Freedman

Microsoft Word MVP FAQ: <http://word.mvps.org>

Re: Sending debugging messages to a second document.

Re: Sending debugging messages to a second document.

Email cannot be acknowledged; please post all follow-ups to the newsgroup
so
all may benefit.

--

Russ

drsmN0SPAMikleAThotmailD0Tcom.INVALID

.