

Re: Shell function – accessing an Access secure db (.MDW file)

Source: <http://www.tech–archive.net/Archive/VB/microsoft.public.vb.general.discussion/2004–04/2519.html>

From: Rick Rothstein (*rickNOSPAMnews_at_NOSPAMcomcast.net*)

Date: 04/19/04

Date: Mon, 19 Apr 2004 10:05:07 -0400

> *Hi – Does anyone know the correct syntax for the shell function. This is my problem.. my Access db has a workgroup (.MDW) file associated with it and what I am trying to do is run my Access macro, through VB5, that runs a query and exports the results into an Excel spreadsheet. The following statement does work correctly (but it is accessing an Access db that has NO .MDW file associated with it), I need to incorporate the username and password:*

> *Retval = Shell("c:\Program Files\Microsoft Office\Office\Msaccess.exe " &*

–
> *"c:\salesrpt\grp\roster\marsrs.mdb /x mcrDownloadRosterWeb", 0)*

> *Maybe this will help some: (This is a sample of my code, that usually use to access the secure db, note: the username is MarsSRS and the password is phone1phase)*

> *swrkgrpfile = Mid("C:\Salesrpt\Grp\Roster\marsrs.mdb", 1, Len("C:\Salesrpt\Grp\Roster\marsrs.mdb") – 1) + "w"*

> *DBEngine.SystemDB = swrkgrpfile*

> *Set wrkConnect = CreateWorkspace("MarsSRS" + CStr(Time), "MarsSRS", _ "phone1phase", dbUseJet)*

> *Set dbRoster = OpenDatabase("C:\Salesrpt\Grp\Roster\marsrs.mdb")*

> *HOPEFULLY SOMEONE CAN HELP ME OUT HERE.. THANKS!!*

Try wrapping each path/filename in their own set of parentheses.

```
Retval = Shell("""c:\Program Files\Microsoft Office\Office\Msaccess.exe """" & _  
    """"c:\salesrpt\grp\roster\marsrs.mdb /x mcrDownloadRosterWeb"""" , 0)
```

(Hopefully your newsreader doesn't break those long lines apart too badly.)

The following is a compilation of several posts I've given in the past regarding the Shell command. The reason for my suggestion is addressed in there; the remainder is for your consideration.

Rick –MVP

You can use the Shell command. To execute internal DOS command (Dir, Copy, etc. as well as redirection of screen output), the command processor must be specified (using the Environ\$ function and "comspec" as its argument returns the correct command processor path on NT and non-NT systems) . Specifying the command processor is safe & generic and will work with non-internal commands also. That syntax, using an XCopy command as an example is:

```
Shell Environ$("comspec") & " /c xcopy "" & _  
    Source & "" "" & Destination & "" " & Option, vbHide
```

You set the Source and Destination (string variables) to the appropriate paths and the Option (string variable), if any, which can be found by opening an MSDOS Prompt window and typing xcopy /?. (Note: You can type /? after any DOS command at a DOS prompt to list the available options for that command.) One more example would be to list all the files in a directory including subdirectories and subdirectories of subdirectories and all of their files.

```
CommandLine = "dir "" & FileSpec & _  
    "" /s/b > "" & RedirectTo & """"  
Shell Environ$("comspec") & " /c " & CommandLine, vbHide
```

Here, the output of a Dir command is redirected to a file-path you specify in the RedirectTo (string variable). The /s/b are options to the Dir command that tell it to recurse through its subdirectories and not to include header or summary information.

I used a variable for the file name so that I could more easily explain the benefit of encasing it in quotemarks. If you redirect to a file that has spaces in its name, or if there are spaces in the path specification itself, then the filename *must* be quoted to protect the spaces from DOS's desire to use them as delimiters. (That's what all those quotemarks in the Shell statement are for.) If the filename doesn't have spaces in it, the quotes aren't necessary BUT they don't hurt either. Hence, the above will work with either.

As for your PING question, something like the following should work:

```
strIP = "4.17.23.1"  
Shell Environ$("comspec") & " /c ping " & _  
    strIP & " > "" & RedirectFile & """" , vbHide
```

Although you didn't specify it in your original post, I assume you want to use vbHide for the optional 2nd parameter to Shell. This hides the DOS window so that your user doesn't see it. If you want the DOS window to

remain visible, you would use the vbNormalFocus BUT you must use a /k instead of a /c for the command processor argument. Basically, the /c tells the command processor "here comes a command and, when its finished executing, close the DOS shell it is running in" whereas the /k also tells the command processor that a command follows, but it instructs it to leave the DOS session running.

The above assumes you do NOT have to wait for this file to be completely written before your code continues executing. If you have to work with this file right after it is created, consider one of these (which makes your program wait until the DOS process is finished):

MICROSOFT 'S OFFICIAL WAY

=====

See this link

<http://support.microsoft.com/support/kb/articles/Q129/7/96.asp>

Note: This method doesn't use Shell -- it uses CreateProcessA.

FAST AND DIRTY METHOD (WORKS ALMOST ALL THE TIME)

=====

Paste these lines in the (General)(Declarations) section of the form where the Shell is being called (or remove the Private keywords and put them in a BAS module if more than one form will use them):

```
Private Declare Function OpenProcess _
    Lib "kernel32" _
    (ByVal dwDesiredAccess As Long, _
    ByVal bInheritHandle As Long, _
    ByVal dwProcessId As Long) As Long
Private Declare Function CloseHandle _
    Lib "kernel32" _
    (ByVal hObject As Long) As Long
Private Declare Function WaitForSingleObject _
    Lib "kernel32" _
    (ByVal hHandle As Long, _
    ByVal dwMilliseconds As Long) As Long
```

Call your Shell command in this form with the appropriate Shell arguments placed in the parentheses:

```
PID = Shell( <<Put Shell Arguments Here>> )
```

And finally, paste the following IMMEDIATELY after the PID=Shell statement above (making sure to handle the possible error where indicated; i.e. stop the code from falling through to your other commands if the Shell failed):

```
If PID = 0 Then
    '
    'Handle Error, Shell Didn't Work
```

microsoft.public.vb.general.discussion: Re: Shell function – accessing an Access secure db (.MDW file)

```
'  
Else  
    hProcess = OpenProcess(&H100000, True, PID)  
    WaitForSingleObject hProcess, -1  
    CloseHandle hProcess  
End If
```