Windows XP or later Windows: How can I run a batch file in the background with no window displayed?

Asked 16 years, 1 month ago Modified 3 years, 5 months ago

Viewed 137k times

135

I know I have already answered a similar question (Running Batch File in background when windows boots up), but this time I need to launch a batch:

- from another batch,
- without any console window displayed,
- *with all arguments* passed to the invisible batch.

The first batch is executed in a console window. However, I do not want the second batch (launched by the first in a **asynchronous way**) to also display a console window.

I have come up with a VBScript script which does just that, and I put the script as an answer for others to refer to, but if you have other ideas/solutions, feel free to contribute.

Note: The console window of Windows command processor is named not really correct DOS window by many people.

Thank you all for the answers. From what I understand, if I need to asynchronously call a script to run in an invisible mode:

- From a second script already in a console window, `start /b` is enough.
- From Windows, *without triggering a second window*, my solution is still valid.

windows batch-file cmd wsh

Share

Improve this question

Follow

edited May 23, 2017 at 12:03

Community Bot

1 ● 1

asked Nov 18, 2008 at 12:05

VonC

1.3m ● 558 ● 4.7k ● 5.6k

You are launching the batch file from ANOTHER batch file? Does this already running batch file have a window?
– Oddthinking Nov 18, 2008 at 12:07

Yes, this other (first) batch is executed in a DOS windows. However, I do not want the second batch (launch by the first in a asynchronous way) displays also a windows (which would happen with a 'start /b' command) – VonC Nov 18, 2008 at 12:13

10 Answers Sorted by: Highest score (default)

118

Here is a possible solution:

From your first script, call your second script with the following line:

```
wscript.exe invis.vbs run.bat %*
```

Actually, you are calling a vbs script with:

- the [path]\name of your script
- all the other arguments needed by your script (`%*`)

Then, invis.vbs will call your script with the Windows Script Host Run() method, which takes:

- intWindowStyle : 0 means "invisible windows"
- bWaitOnReturn : false means your first script does not need to wait for your second script to finish

Here is invis.vbs:

```
set args = WScript.Arguments
num = args.Count

if num = 0 then
    WScript.Echo "Usage: [CScript | WScript] invis.vbs
arguments>"
    WScript.Quit 1
end if

sargs = ""
```

```
if num > 1 then
    sargs = " "
    for k = 1 to num - 1
        anArg = args.Item(k)
        sargs = sargs & anArg & " "
    next
end if

Set WshShell = WScript.CreateObject("WScript.Shell")

WshShell.Run """" & WScript.Arguments(0) & """" & sarg
```

Share Improve this answer

Follow

edited Dec 4, 2016 at 15:34

Ilyich

5,756 ● 3 ● 41 ● 31

answered Nov 18, 2008 at 12:05

VonC

1.3m ● 558 ● 4.7k ● 5.6k

> The only problem is that this creates a process named "cmd.exe", which is hard to find in the future. – YetAnotherBot Feb 4, 2019 at 14:14

61

Do you need the second batch file to run asynchronously? Typically one batch file runs another synchronously with the `call` command, and the second one would share the first one's window.

You **can** use `start /b` *second.bat* to launch a second batch file **a**synchronously from your first that shares your first one's window. If both batch files write to the console simultaneously, the output will be overlapped and probably indecipherable. Also, you'll want to put an `exit`

command at the end of your second batch file, or you'll be within a second `cmd` shell once everything is done.

Share Improve this answer

Follow

answered Nov 18, 2008 at 12:48

P Daddy

29.5k ● 9 ● 72 ● 94

1 Correct. I believed start /b would open a new windows, but if executed from a DOS windows, it does share the same windows. – VonC Nov 18, 2008 at 15:44

12

I think this is the easiest and shortest solution to running a batch file without opening the DOS window, it can be very distracting when you want to schedule a set of commands to run periodically, so the DOS window keeps poping up, here is your solution. Use a VBS Script to call the batch file ...

```
Set WshShell = CreateObject("WScript.Shell" )
WshShell.Run chr(34) & "C:\Batch Files\ mycommands.bat
Set WshShell = Nothing
```

Copy the lines above to an editor and save the file with .VBS extension. Edit the .BAT file name and path accordingly.

Share Improve this answer

Follow

edited Dec 4, 2016 at 14:28

Ilyich

5,756 ● 3 ● 41 ● 31

answered Jan 15, 2014 at 6:17

indago
2,101 ● 3 ● 31 ● 48

1 Interesting approach, certainly shorter than my solution. +1 – VonC Jan 15, 2014 at 6:18

1 Great solution. Reading this I've notice there is no really need to use `Set WshShell = Nothing`, worked fine for me and you can have an even smaller way to do it. – Patrick Bard Feb 28, 2014 at 7:51

11

Convert the batch file to an exe. Try Bat To Exe Converter or Online Bat To Exe Converter, and choose the option to run it as a ghost application, i.e. no window.

Share Improve this answer

Follow

edited Sep 20, 2011 at 18:22

E-rich
9,501 ● 11 ● 49 ● 83

answered Nov 18, 2008 at 14:26

Rob Kam
10.2k ● 14 ● 57 ● 65

3 Possible, but I will try to avoid any extra step in this instance. – VonC Nov 18, 2008 at 15:54

31 I'm usually not that paranoid but wouldn't it be very stupid to let someone anonymous generate that exe for me? All kinds of malicious stuff could be injected. – Tobias Apr 29, 2012 at 11:13

7

For self-hiding you can use getCmdPID.bat and windowMode.bat:

```
@echo off

echo --- self hiding bat ----
pause
call getCmdPid.bat
set PID=%errorlevel%
call windowMode.bat -pid %PID% -mode hidden
```

Here's my collection of ways to achieve that - and even more - where it was possible I've tried to return also the PID of the started process (all linked scripts can be downloaded and saved with whatever name you find convenient):

1. The **IEXPRESS** solution can be used even on old win 95/98 machines. Iexpress is a really ancient tool that is still packaged with Windows - as arguments accepts only the command and its arguments.

Example usage:

```
call IEXPhidden.bat "cmd /c myBat.bat"  "argument"
```

2. **SCHTASKS** - Again accepts only two arguments - the command and the arguments.Also checks if it's started with elevated permissions and if possible gets the PID of the process with WEVTUTIL

(available from Vista and above so the newer version of windows will receive the PID) command.

Example usage:

```
call SCHPhidden.bat "cmd /c myBat.bat"  "argument"
```

3. 'WScript.Shell' - the script is full wrapper of 'WScript.Shell' and every possible option can be set through the command line options.It's a jscript/batch hybrid and can be called as a bat.

Example usage (for more info print the help with '-h'):

```
call ShellRunJS.bat "notepad.exe" -style 0 -wait
no
```

4. 'Win32_ProcessStartup' - again full wrapper and all options are accessible through the command line arguments.This time it's WSF/batch hybrid with some Jscript and some VBScript pieces of code - but it returns the PID of the started process.If process is not hidden some options like X/Y coordinates can be used (not applicable for every executable - but for example cmd.exe accepts coordinates).

Example usage (for more info print the help with '-h'):

```
call win32process.bat "notepad" -arguments "/A
openFile.txt"  -showWindows 0 -title "notepad"
```

5. The .[NET solution] . Most of the options of ProcessStartInfo options are used (but at the end I was too tired to include everything):

Example usage (for more info print the help with '-h'):

```
call ProcessStartJS.bat "notepad" -arguments "/A
openFile.txt"  -style Hidden -directory "." -title
"notepad" -priority Normal
```

Share Improve this answer

Follow

edited Nov 7, 2020 at 10:09

answered Oct 15, 2015 at 15:21

npocmaka

57.2k ● 18 ● 161 ● 193

4

Run it under a different user name, using "runas" or by scheduling it under a different user in Windows Scheduled Tasks.

Share Improve this answer

Follow

answered Nov 18, 2008 at 14:40

JosephStyons

58.6k ● 64 ● 167 ● 237

Possible, but not exactly what I am after (direct asynchronous call in background) – VonC Nov 18, 2008 at 15:55

I know this already has answers, but if anyone stumbles across this, this might help:

Here's a good way to run it COMPLETELY in the background, without any way to see it. The only way you can interact with it would be to terminate it using taskmgr or Process Hacker.

This uses a third-party tool called NirCmd.

1. Make a script containing this code:

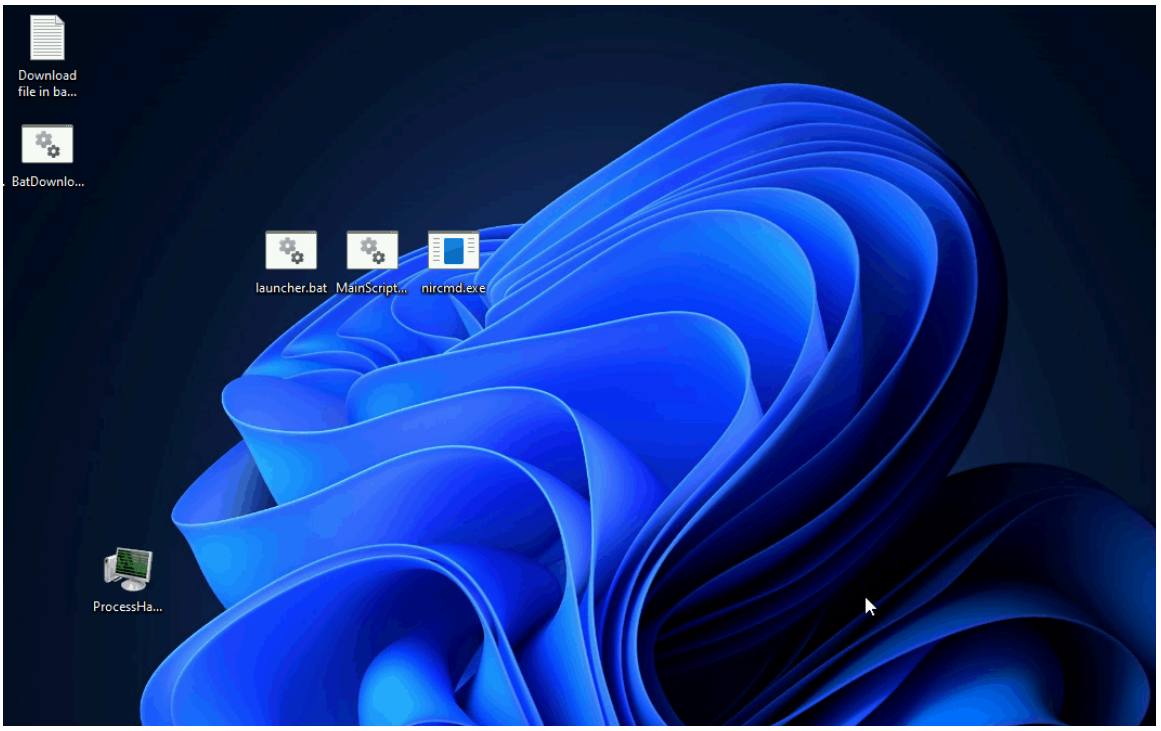
```
@echo off
nircmd exec hide "MainScript.bat"
```

2. Save it as whatever you like. "MainScript.bat" can be anything, but note it for later.
3. Now, make another script with the name you noted earlier. For the example, I'll call it MainScript.bat and give it this code:

```
@echo off
echo Can you see me now?
pause
exit
```

4. This will be the code you want to execute in the background.
5. Make sure they are in the same directory, and download NirCmd into the same directory.

Here's the result of these scripts:



[NirCmd Download Link]

[Demo ZIP File]

Note: NirCmd isn't my tool, but its still helpful.

Share Improve this answer Follow

edited Jul 4, 2021 at 14:17

answered Jul 4, 2021 at 13:49

i Mr Oli i

745 ● 5 ● 13

Haha, unblocked CMD is a tool I use when services let you use bat scripts but not command prompt. `nircmd.exe` isn't my tool, you can find the original author by googling it.
– i Mr Oli i Jul 4, 2021 at 14:16

Right: nirsoft.net/utils/nircmd.html, from Nir Sofer. However, as said here: "I'd feel differently if Nir Sofer released source." – VonC Jul 4, 2021 at 14:25

I haven't changed anything, just uploaded a direct mirror to my GitHub. :) – i Mr Oli i Jul 4, 2021 at 14:29

2

In the other question I suggested autoexnt. That is also possible in this situation. Just set the service to run manually (ie not automatic at startup). When you want to run your batch, modify the autoexnt.bat file to call the batch file you want, and start the autoexnt service.

The batchfile to start this, can look like this (untested):

```
echo call c:\path\to\batch.cmd %* >
c:\windows\system32\autoexnt.bat
net start autoexnt
```

Note that batch files started this way run as the system user, which means you do not have access to network shares automatically. But you can use *net use* to connect to a remote server.

You have to download the Windows 2003 Resource Kit to get it. The Resource Kit can also be installed on other versions of windows, like Windows XP.

Share Improve this answer

Follow

answered Nov 18, 2008 at 13:15

wimh

15.2k ● 6 ● 49 ● 98

1 Interesting, but a tight overkill for this scenario. – VonC Nov 18, 2008 at 15:46

0

You can run your .bat file through a .vbs file
Copy the following code into your .vbs file :

```
Dim WshShell
Dim obj
Set WshShell =
WScript.CreateObject("WScript.Shell")
obj = WshShell.Run("C:\Users\file1.bat", 0)
obj = WshShell.Run("C:\Users\file2.bat", 0)  and
so on
set WshShell = Nothing
```

Share Improve this answer

Follow

edited Jul 9, 2014 at 6:30

avck
3,693 ● 3 ● 29 ● 38

answered Jan 31, 2014 at 12:57

Vinod Chelladurai
539 ● 7 ● 16 ● 27

0

You also can use

```
start /MIN notepad.exe
```

PS: Unfortunatly, minimized window status depends on command to run. V.G. doen't work

```
start /MIN calc.exe
```

Share Improve this answer

Follow

answered Mar 13, 2019 at 13:10

user2928048

4,118 ● 2 ● 14 ● 12