How do you loop through each line in a text file using a windows batch file?

Ask Question

213

I would like to know how to loop through each line in a text file using a Windows batch file and process each line of text in succession.

windows batch-file

★ 58

edited Mar 24 '10 at 14:12
Matthew Murdoch
20.7k 24 84 121

asked Oct 1 '08 at 2:01
Mr. Kraus
4,548 5 22 30

Home

PUBLIC

Stack Overflow

Tags

Users

Jobs

Teams
Q&A for work

Learn More

11 Answers

275

✓

The posts below helped greatly, but did not do what I stated in my question where I needed to process the entire line as a whole. Here is what I found to work.

```
for /F "tokens=*" %%A in (myfile.txt) do [process] %%A
```

The tokens keyword with an asterisk (*) will pull all text for the entire line. If you don't put in the asterisk it will only

For Command on TechNet

I appreciate all of the posts!

If there are spaces in your file path, you need to use `usebackq` . For example.

```
for /F "usebackq tokens=*" %%A in ("my file.txt") do [proc
```

edited Jun 25 '18 at 8:29
Dan
3,276 3 20 49

answered Oct 2 '08 at 18:33
Mr. Kraus
4,548 5 22 30

29 A minor addition: to make this work from the command line interactively, replace `%%A` with `%A` in the above command. Otherwise you'll get `%%A was unexpected at this time.` . – vadipp Nov 12 '12 at 12:02

16 FYI, if you need to do a multi-line command, after "DO" you can put an open parenthesis "(" and a few lines later, end it with a close parenthesis ")" -- and you can just put your code block inside those (indented to your tastes). – BrainSlugs83 Jan 27 '13 at 4:49

If you have issues with files in the same folder with nested batch files, or files that have spaces in their names, look at my solution below. – Marvin Thobejane Mar 10 '14 at 12:04

3 Thanks for that pattern. I have found that I can't put quotes(") around a the file name -- For file names with spaces just gives me the file name. E.g. `for /F "tokens=*" %%A in ("myfile.txt") do echo A = %%A --> A = myfile.txt` . Any ideas how to thwart this? – will Apr 19 '16 at 2:52

Make sure the file you working is encoded in ANSI or UTF8. I wasn't what I was expecting. At this point I noticed the file had been encoded in "UCS-2 BE BOM" for some reason! – Dan Stevens May 3 '16 at 15:49

53

From the Windows command line reference:

To parse a file, ignoring commented lines, type:

```
for /F "eol=; tokens=2,3* delims=," %i in (myfile.txt) do (
```

This command parses each line in Myfile.txt, ignoring lines that begin with a semicolon and passing the second and third token from each line to the FOR body (tokens are delimited by commas or spaces). The body of the FOR statement references %i to get the second token, %j to get the third token, and %k to get all of the remaining tokens.

If the file names that you supply contain spaces, use quotation marks around the text (for example, "File Name"). To use quotation marks, you must use usebackq. Otherwise, the quotation marks are interpreted as defining a literal string to parse.

By the way, you can find the command-line help file on most Windows systems at:

```
"C:\WINDOWS\Help\ntcmds.chm"
```

edited Feb 8 '10 at 10:29

answered Oct 1 '08 at 2:06

2 Nice explanation. – djangofan Sep 21 '11 at 19:35

7 to clarify the *"to use quotation marks, you must use* `usebackq` *"*: `for /f "usebackq" %%a in ("Z:\My Path Contains Spaces\xyz\abc.txt")` – drzaus Feb 21 '14 at 16:15

30

In a Batch File you **MUST** use `%%` instead of `%` : (Type `help for`)

```
for /F "tokens=1,2,3" %%i in (myfile.txt) do call :process
goto thenextstep
:process
set VAR1=%1
set VAR2=%2
set VAR3=%3
COMMANDS TO PROCESS INFORMATION
goto :EOF
```

What this does: The "do call :process %%i %%j %%k" at the end of the for command passes the information acquired in the for command from myfile.txt to the "process" 'subroutine'.

When you're using the for command in a batch program, you need to use double % signs for the variables.

The following lines pass those variables from the for command to the process 'sub routine' and allow you to process this information.

```
set VAR1=%1
 set VAR2=%2
 set VAR3=%3
```

Add in your EOL or Delims as needed of course.

edited Dec 20 '12 at 21:40
Toby Allen
8,797 8 63 117

answered May 4 '10 at 14:39
user332474
301 3 2

23

Improving the first "FOR /F.." answer: What I had to do was to call execute every script listed in MyList.txt, so it worked for me:

```
for /F "tokens=*" %A in  (MyList.txt) do CALL %A ARG1
```

--OR, if you wish to do it over the multiple line:

```
for /F "tokens=*" %A in  (MuList.txt) do (
ECHO Processing %A....
CALL %A ARG1
)
```

Edit: The example given above is for executing FOR loop from command-prompt; from a batch-script, an extra % needs to be added, as shown below:

```
---START of MyScript.bat---
@echo off
for /F "tokens=*" %%A in  ( MyList.TXT) do  (
   ECHO Processing %%A....
   CALL %%A ARG1
)
@echo on
```

edited Jan 24 '12 at 6:07

answered Jan 23 '12 at 18:36

Yogesh Mahajan
409 5 8

21

@MrKraus's answer is instructive. Further, let me add that **if you want to load a file located in the same directory** as the batch file, prefix the file name with %~dp0. Here is an example:

```
cd /d %~dp0
for /F "tokens=*" %%A in (myfile.txt) do [process] %%A
```

NB:: If your file name or directory (e.g. myfile.txt in the above example) has a space (e.g. 'my file.txt' or 'c:\Program Files'), use:

```
for /F "tokens=*" %%A in ('type "my file.txt"') do [proces
```

, with the **type** keyword calling the `type` program, which displays the contents of a text file. If you don't want to suffer the overhead of calling the type command you should change the directory to the text file's directory. Note that type is still required for file names with spaces.

I hope this helps someone!

edited May 23 '17 at 12:10

Community ♦
1 1

Marvin Thobejane
1,426 16 12

There's no need to prefix the filename as the batch file will look in the current folder by default. – foxidrive May 28 '13 at 12:56

@foxidrive: Okay, I hear you. Although care should be taken. For example if a directory was changed it would look in that directory rather than the one the batch file is in. The solution then would be calling `**cd /d %~dp0**` before the for loop. This would make sure you are referencing a file in the directory the batch file is in. Thanks for the observation – Marvin Thobejane Jul 18 '13 at 12:01

2 Thx and +1 for the `type` walkaround – halex Mar 8 '14 at 21:11

I can't get the `type` work around working, I've had to quote my filename because it's in a different directory that contains spaces(Damn you `Program Files`). I'm getting an error `The system cannot find the file `type.` – scragar Jul 3 '14 at 15:48

1 @scragar, have you got the right quote? it needs to be a ' not a `. On my keyboard it's on the same key as @ – FrinkTheBrave Jul 10 '14 at 14:17

14

The accepted answer is good, but has two limitations.
It drops empty lines and lines beginning with `;`

To read lines of any content, you need the delayed expansion toggling technic.

```
@echo off
SETLOCAL DisableDelayedExpansion
FOR /F "usebackq delims=" %%a in (`"findstr /n ^^ text.txt
    set "var=%%a"
    SETLOCAL EnableDelayedExpansion
```

```
        ENDLOCAL
    )
```

Findstr is used to prefix each line with the line number and a colon, so empty lines aren't empty anymore.

DelayedExpansion needs to be disabled, when accessing the `%%a` parameter, else exclamation marks `!` and carets `^` will be lost, as they have special meanings in that mode.

But to remove the line number from the line, the delayed expansion needs to be enabled.
`set "var=!var:*:=!"` removes all up to the first colon (using `delims=:` would remove also all colons at the beginning of a line, not only the one from findstr).
The endlocal disables the delayed expansion again for the next line.

The only limitation is now the line length limit of ~8191, but there seems no way to overcome this.

edited Sep 19 '16 at 17:48

answered Jul 18 '13 at 12:52

jeb
59.2k 13 136 175

13

Or, you may exclude the options in quotes:

```
FOR /F %%i IN (myfile.txt) DO ECHO %%i
```

763 1 9 25

I get error: %%i unexpected at this time – shaifali Gupta Feb 9 '18 at 16:27

1 Two percent signs next to each other %% are treated like a single percent sign in a command (not a batch file). – Paul Feb 24 '18 at 8:09

9

Here's a bat file I wrote to execute all SQL scripts in a folder:

```
REM ******************************************************
REM Runs all *.sql scripts sorted by filename in the curre
REM To use integrated auth change -U <user> -P <password> 
REM ******************************************************

dir /B /O:n *.sql > RunSqlScripts.tmp
for /F %%A in (RunSqlScripts.tmp) do osql -S (local) -d DE
USERNAME_GOES_HERE -P PASSWORD_GOES_HERE -i %%A
del RunSqlScripts.tmp
```

answered Oct 1 '08 at 2:08

mikej

4 Skip the temp file and just use for /f %%A in ('dir /b /o:n *sql') do... – Adam Mitz Oct 1 '08 at 2:22

If you have an NT-family Windows (one with `cmd.exe` as the shell), try the FOR /F command.

answered Oct 1 '08 at 2:03
Michael Ratanapintha
32.2k 4 23 36

2

The accepted anwser using `cmd.exe` and

```
for /F "tokens=*" %F in (file.txt) do whatever "%F" ...
```

works only for "normal" files. It fails miserably with huge files.

For big files, you may need to use Powershell and something like this:

```
[IO.File]::ReadLines("file.txt") | ForEach-Object { whatev
```

or if you have enough memory:

```
foreach($line in [System.IO.File]::ReadLines("file.txt"))
```

This worked for me with a 250 MB file containing over 2 million lines, where the `for /F ...` command got stuck after a few thousand lines.

For the differences between `foreach` and `ForEach-Object`, see Getting to Know ForEach and ForEach-Object.

(credits: Read file line by line in PowerShell)

answered Oct 14 '18 at 19:39
mivk
6,392 1 38 41

1

Modded examples here to list our Rails apps on Heroku - thanks!

```
cmd /C "heroku list > heroku_apps.txt"
find /v "=" heroku_apps.txt | find /v ".TXT" | findstr /r
heroku_apps_list.txt
for /F "tokens=1" %%i in (heroku_apps_list.txt) do heroku
%%i
```

Full code here.

answered Jan 11 '13 at 19:27

sendbits

324 3 4

Per comment to another question above - You can skip the file creation/reading and just use `for /f "tokens=1" %%i in ('find /v "=" heroku_apps.txt ^| find /v ".TXT" ^| findstr /r /v /c:"^$"') do...` (Note the addition of `^` 's used to escape the pipe, so that it is passed to the `for` and not directly to the command processor) – user66001 Feb 3 '13 at 9:13

protected by Community ♦ Nov 5 '13 at 9:47

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?