How can I echo a newline in a batch file?

573

102

How can you you insert a newline from your batch file output?

I want to do something like:

```
echo hello\nworld
```

Which would output:

```
hello
world
```

batch-file newline

edited Jan 30 at 10:22
double-beep
3,142 5 16 32

asked Sep 25 '08 at 11:50
Brian R. Bondy
259k 101 548 595

6 Came in useful for me. I had to do echo \n \n | my_app.exe in a script. I did (echo. && echo.) | my_app.exe – Vignesh Feb 18 '11 at 9:54

2 Easy Approach " Code starts :" > echo hello&echo world , will give u what u need – prudviraj Jan 22 '15 at 13:59

You can insert an invisible ascii chr(255) on a separate line which will force a blank new line. Hold down the [alt] key and press 255 on the keypad. this inserts chr(255) which is a blank square. i.e. "*echo (alt+255)*" You can only use the keypad not the numbers at the top of the querty keyboard! – jwzumwalt Nov 20 '18 at 4:54

18 Answers

427

```
echo hello & echo.world
```

This means you could define `& echo.` as a constant for a newline `\n`.

edited Apr 19 '18 at 13:27
Gerhard Barnard
10.3k 3 14 33

answered Sep 25 '08 at 11:52
Grimtron
5,346 3 19 28

13 Also works without period: echo hello && echo world – Alexander Prokofyev Oct 10 '08 at 9:14

39 the example doesn't need a period, but you do need one to echo a blank empty line: `echo. && echo hello && echo. && echo world` – matt wilkie Jun 16 '11 at 22:37

4 Can you do this with a single echo so it can be redirected to a file? – Shahbaz Oct 25 '11 at 20:36

44 @Shahbaz - `$ (echo Hello && echo World) > ./File.txt` – Nathan J. Brauer Nov 30 '11 at 5:28

24 The period thing in "echo." never stops amazing me. It's so dated, and still I always forget that the dot must be strictly concatenated with the command name, with no spaces between. There's no error in the post of yours, I'm writing this just as a reminder: "echo ." != "echo." ! – quetzalcoatl Feb 10 '12 at 11:19

Use:

Peter Mortensen
14.1k 19 88 114

Matt Lacey
60.2k 10 83 140

12 Is it possible while providing a string within a single echo statement? – Brian R. Bondy Sep 25 '08 at 11:52

5 Why do you need to do it with a single echo statement; where's the harm in having another? I assume you're not simply adding unnecessary constraints for the fun of it... – Rob Sep 25 '08 at 11:54

3 @Rob, I just came across this problem and none of these work. You need to echo in a single statement in my example. I am generating some tex files from HTML and generating a Makefile by using `echo "Makefile contents (which has \n)" > Makefile` With multiple echos, it wouldn't work – Shahbaz Oct 25 '11 at 20:35

3 Use & (or &&) to do them in a single statement: For example, to put "hello\nworld" in a txt file: `(echo hello & echo world) >> ./test.txt` – Ned Martin Mar 23 '16 at 6:50

4 For multi-line output to a file why not just do the following? `echo line 1 > Makefile` and then `echo line 2 >> Makefile` . Use of `>>` causes the output to be appended to the file, which is precisely the behavior you need for this case. – dgnuff Jan 4 '18 at 16:49

126

Here you go, create a .bat file with the following in it :

```
@echo off
REM Creating a Newline variable (the two blank lines are required!)
set NLM=^


set NL=^^^%NLM%%NLM%^%NLM%%NLM%
REM Example Usage:
echo There should be a newline%NL%inserted here.

echo.
pause
```

You should see output like the following:

```
There should be a newline
inserted here.
```

edited Feb 26 '18 at 15:51
jeb
59.4k 13 137 178

answered Nov 6 '08 at 18:43
Ken

46 Very impressive, could you take time to explain how the line set NL=^^^%NLM%%NLM%^%NLM%%NLM% works? I can't quite get my head round it – Andy Morris Nov 12 '09 at 13:14

8 @andy methinks a +8 comment warrants a question: stackoverflow.com/questions/6379619/… – matt wilkie Jun 16 '11 at 23:06

77 This is a wonderful example to show that cmd.exe and Windows batch files are totally insane! – mivk Oct 15 '11 at 10:54

12 NOTE TO SELF: The line "set NLM=^" **must** have nothing after the CARET and **must** have 2 blank lines after it. – mnemotronic Oct 24 '15 at 16:26

2 See my answer for a simpler method. `set nl=^&echo.` , then just `echo hello %nl% world` yields the same effect. – coltonon Jun 29 '16 at 5:37

93

There is a standard feature `echo:` in cmd/bat-files to write blank line, which emulates a new line in your cmd-output:

```
@echo off
@echo line1
@echo:
@echo line2
```

Output of cited above cmd-file:

```
line1

line2
```

answered Jun 26 '10 at 8:11
macropas
2,202 2 18 23

14 `echo,` and `echo.` also work – BlueRaja - Danny Pflughoeft Jun 10 '11 at 15:50

3 @naxa - Nothing official, just collective knowledge derived by a few batch hackers. The goal is to have syntax that can be used to ECHO any value, including nothing, and never have to worry about getting anything other than what is expected. The best summary I know of is at dostips.com/forum/viewtopic.php?p=4554#p4554. It kind of leaves you hanging, but no one has ever come up with a scenario where `ECHO(` fails. – dbenham Apr 18 '14 at 17:20

2 @Pacerier - there are examples of failure for each of those suggestions. Carefully read dostips.com/forum/viewtopic.php?p=4554#p4554. – dbenham Jul 29 '15 at 12:23

59

Like the answer of Ken, but with the use of the delayed expansion.

```
setlocal EnableDelayedExpansion
(set \n=^
%=Do not remove this line=%
)

echo Line1!\n!Line2
echo Works also with quotes "!\n!line2"
```

First a single linefeed character is created and assigned to the \n-variable.
This works as the caret at the line end tries to escape the next character, but if this is a Linefeed it is ignored and the next character is read and escaped (even if this is also a linefeed).
Then you need a third linefeed to end the current instruction, else the third line would be appended to the LF-variable.
Even batch files have line endings with CR/LF only the LF are important, as the CR's are removed in this phase of the parser.

The advantage of using the delayed expansion is, that there is no special character handling at all.
`echo Line1%LF%Line2` would fail, as the parser stops parsing at single linefeeds.

More explanations are at
SO:Long commands split over multiple lines in Vista/DOS batch (.bat) file
SO:How does the Windows Command Interpreter (CMD.EXE) parse scripts?

Edit: Avoid `echo.`

This doesn't answer the question, as the question was about single `echo` that can output multiple lines.

```
echo,
echo;
echo(
echo/
echo+
echo=
```

But the use of `echo.`, `echo\` or `echo:` should be avoided, as they can be really slow, depending of the location where the script will be executed, like a network drive.

edited May 23 '17 at 12:18
Community ♦
1 1

answered Jun 16 '11 at 23:52
jeb
59.4k 13 137 178

If you will be passing the string with newlines as a parameter to a batch subroutine, e.g. `call :pause line1!\n!line2`, start the subroutine with `setlocal EnableDelayedExpansion` and end it with `setlocal DisableDelayedExpansion` to keep the newlines from being interpreted prematurely. – stevek_mcc Nov 8 '16 at 12:36

+1 for the cast of characters `echo,` `echo;` `echo(` `echo/` `echo+` `echo=` and the rogues gallery `echo.` `echo\` `echo:`. Now if you were to explain why in holy heck these work, I'd owe you a free beer. – Bob Stein Apr 7 at 12:33

37

`echo.` Enough said.

If you need it in a single line, use the `&`. For example,

```
echo Line 1 & echo. & echo line 3
```

would output as:

```
Line 1

line 3
```

Outputs

```
hello
world
```

Then just throw in a `%n%` whenever you want a new line in an echo statement. This is more close to your `\n` used in various languages.

Breakdown

`set n=` sets the variable `n` equal to:

`^` Nulls out the next symbol to follow:

`&` Means to do another command on the same line. We don't care about errorlevel(its an echo statement for crying out loud), so no `&&` is needed.

`echo.` Continues the echo statement.

All of this works because you can actually create variables that are code, and use them inside of other commands. It is sort of like a ghetto function, since batch is not exactly the most advanced of shell scripting languages. This only works because batch's poor usage of variables, not designating between ints, chars, floats, strings, etc naturally.

If you are crafty, you could get this to work with other things. For example, using it to echo a tab

```
set t=^&echo.     ::there are spaces up to the double colon
```

edited Jun 29 '16 at 5:33

answered Jul 16 '14 at 23:50
coltonon
588 9 28

4 **Do not** use `echo.` as it first attempts to find a file by that name. Use other punctuation, such as `echo,` – Kuba Ober Dec 22 '16 at 17:56

answered Aug 9 '09 at 1:00

Crend King
2,210 2 24 37

8 The first example, using ">", will create a new file, the second one, using ">>", will append to an existing file (or create it if it doesn't already exist). – Yann Duran Aug 9 '12 at 15:27

`(echo hello && echo world) > temp` mentioned by Nathan J. Brauer in a comment to the (current) accepted answer works. – Gerold Broser Jun 9 '16 at 9:35

15

Just like Grimtron suggests - here is a quick example to define it:

```
@echo off
set newline=^& echo.
echo hello %newline%world
```

Output

```
C:\>test.bat
hello
world
```

edited Oct 21 '15 at 9:30
Peter Mortensen
14.1k 19 88 114

answered Mar 5 '10 at 16:47
user287293

It will actually echo an additional newline after "world" – BlueRaja - Danny Pflughoeft Jun 10 '11 at 15:52

1 @blue the trailing newline seems to be function of the batch file itself. If you repeat the `echo hello %newline%world` line there are no spaces between. – matt wilkie Jun 16 '11 at 22:59

2 `echo "Hello %newline% world"` fails as it is inside of quotes. Because it isn't a *real* newline. – jeb Oct 20 '13 at 20:23

13

```
(echo a & echo. & echo b) > file_containing_multiple_lines.txt
```

answered Sep 2 '14 at 0:09

test30

2,291 23 23

9

You can also do like this,

```
(for %i in (a b "c d") do @echo %~i)
```

The output will be,

```
a
b
c d
```

Note that when this is put in a batch file, '%' shall be doubled.

```
(for %%i in (a b "c d") do @echo %%~i)
```

edited Sep 29 '11 at 19:05

answered Sep 20 '11 at 17:06

albert

302 3 7

This solution also works if you want to echo an ampersand & instead of a newline. – parvus Jun 26 '14 at 13:48

i think i found a bug: the `*` char – stevefestl Mar 26 '17 at 11:58

I hope this helps at least one other person out there :)

answered Apr 22 '13 at 4:56
Wayne Uroda
3,781 2 22 32

3

To start a new line in batch, all you have to do is add "echo[", like so:

```
echo Hi!
echo[
echo Hello!
```

answered Dec 16 '16 at 20:31
PryroTech
273 1 10

3 It's slow and it fails when a file exists with the name `echo[.bat` – jeb Dec 23 '16 at 10:17

What? It should work instantly, and also just delete the echo[.bat file. – PryroTech Dec 23 '16 at 15:56

1 It is pretty much the exact same thing as echo. syntax. – PryroTech Dec 23 '16 at 15:57

4 Yes and `echo.` is a bad solution, too. – jeb Dec 24 '16 at 18:11

As can be seen in previous answers, this will try to locate a file on the path first, which causes it to be so slow. Same is true for `echo.` . Expecting users of your script to delete files on their system just because your script needs a newline doesn't sound like good design to me. – Abel Oct 29 '17 at 20:31

2

This worked for me, no delayed expansion necessary:

```
@echo off
(
echo ^<html^>
```

It writes output like this:

```
<html>
<body>
Hello
</body>
</html>
Press any key to continue . . .
```

edited Dec 3 '13 at 15:48
djangofan
16.2k 47 150 238

answered Jun 2 '10 at 15:35
NahuelGQ
147 9

4 -1 `echo asdf >myfile.txt` will produce the exact same results. `echo` appends a newline to the end of the string. – BlueRaja - Danny Pflughoeft Jun 10 '11 at 15:51

2 Great answer, being the only person who recognized the value of using the parenthesis for new lines. I believe this is how the makers of DOS intended you to do it. – djangofan Dec 3 '13 at 15:48

2

Ken and Jeb solutions works well.

But the new lines are generated with only an LF character and I need CRLF characters (Windows version).

To this, at the end of the script, I have converted LF to CRLF.

Example:

```
TYPE file.txt | FIND "" /V > file_win.txt
del file.txt
rename file_win.txt file.txt
```

edited Apr 1 '17 at 6:40

answered Sep 30 '16 at 15:24

2

If one needs to use famous \n in string literals that can be passed to a variable, may write a code like in the *Hello.bat* script below:

```
@echo off
set input=%1
if defined input (
    set answer=Hi!\nWhy did you call me a %input%?
) else (
    set answer=Hi!\nHow are you?\nWe are friends, you know?\nYou can call me by name.
)

setlocal enableDelayedExpansion
set newline=^


rem Two empty lines above are essential
echo %answer:\n=!newline!%
```

This way multiline output may by prepared in one place, even in other scritpt or external file, and printed in another.

The line break is held in *newline* variable. Its value must be substituted **after** the *echo* line is expanded so I use *setlocal enableDelayedExpansion* to enable exclamation signs which expand variables on execution. And the execution substitutes *\n* with *newline* contents (look for syntax at *help set*). We could of course use *!newline!* while setting the *answer* but *\n* is more convenient. It may be passed from outside (try *Hello R2\nD2*), where nobody knows the name of variable holding the line break (Yes, *Hello C3!newline!P0* works the same way).

Above example may be refined to a subroutine or standalone batch, used like `call:mlecho Hi\nI'm your comuter` :

```
:mlecho
setlocal enableDelayedExpansion
set text=%*
set nl=^


echo %text:\n=!nl!%
goto:eof
```

Please note, that additional backslash won't prevent the script from parsing \n substring.

You are essentially doing the same thing as Jeb's answer above. – Squashman Jun 8 '18 at 15:18

Answers I read above were very helpful but I needed to call a script od subroutine which will parse \n so I'd avoid implementing the trick everywhere. This needed to refine a solution and I shared this refinement. Hope it didn't harm any animal. – Tomator Jun 29 '18 at 8:36

0

You can use `@echo` (@echo + [space] + [insecable space])

Note: The insecable space can be obtained with Alt+0160

Hope it helps :)

[edit] Hmm you're right, I needed it in a Makefile, it works perfectly in there. I guess my answer is not adapted for batch files... My bad.

edited Jul 19 '13 at 7:58

answered Jul 18 '13 at 13:22
johan d.
2,454 13 25

I got `ECHO ist eingeschaltet (ON).` not an empty line, tried at the cmd-prompt – jeb Jul 18 '13 at 13:29

That's a non-breaking space, not a newline. – Ry- ♦ Jul 18 '13 at 15:36

-1

I can't make it any simpler than:

```
echo echo hello^&echo world^&pause>silly.bat
call silly.bat
```

edited Oct 21 '15 at 9:31
Peter Mortensen
14.1k 19 88 114

answered Dec 30 '14 at 5:05
strance
53 3

Here are other examples

answered Jul 30 '18 at 14:41
hoodwinked
29 7

4 But the question isn't about linux nor bash. It's about windows batch – jeb Jul 30 '18 at 15:32

Touche @jeb. Maybe I should become more familiar with windows. Surprised of why it brought me here. – hoodwinked Jul 30 '18 at 17:37

protected by Sam Saffron Dec 20 '10 at 2:44

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?