

Why can I not access a file named `-` even when quoting the filename? [duplicate]

Asked 3 years, 7 months ago Modified 9 months ago Viewed 749 times

This question already has answers here:

[How do you enter a directory that's name is only a minus?](#) (4 answers)

Closed 3 years ago.

I have a file called `-`. I want to display its contents.

- One way is to do

```
cat ./-
```

since `cat -` reads from standard input.

- However, why are `cat "-"` and `cat '-'` also interpreted by the shell as `cat -`?

shell quoting cat

Share Improve this question Follow

edited Nov 23, 2021 at 10:08



AdminBee

19.2k ● 16 ● 43 ● 67


asked Feb 12, 2019 at 10:01



Jhdoe

123 ● 2

- 4 `cat` is specifically implemented to interpret the file name `-` as "use stdin". This means you cannot quote `-` to remove this meaning. If you pass additional quotes to `cat`, e.g. with `cat \"-\"`, it will look for a file name `"-"` that includes quotes. So `cat` would need an option to disable the special meaning of `-`. If you happen to have a file `-` and pass it to `cat` by shell globbing, e.g. `cat *`, you would have to use something like `cat ./*` to work around the

problem. For better suggestions, please, describe your use case where you have to avoid problems with a file `-`. – Bodo Feb 12, 2019 at 11:41 

1 Related: unix.stackexchange.com/q/1519/117549 – Jeff Schaller ♦ Feb 12, 2019 at 13:52

4 Note that the purpose of `./-` is not to "disguise" the argument from the shell, but to make it distinct from `-` for `cat`. The convention that `-` represents standard input is handled *internally* by programs. – chepner Feb 12, 2019 at 14:23

The answer is in the manual: gnu.org/software/bash/manual/bash.html#Quote-Removal – glenn jackman Feb 12, 2019 at 14:59

1 @GiacomoAlzetta, Of course there are, and many to choose from. But seriously, there *are* guidelines ([POSIX Utility Syntax Guidelines](https://pubs.opengroup.org/onlinepubs/9696911199/basedefs/POSIX_Utility_Syntax_Guidelines.html)), and conventions ([getopt](https://www.gnu.org/software/getopt/)), if not all-encompassing mandatory standards. – ilkkachu Feb 12, 2019 at 17:44 

3 Answers

Sorted by: Highest score (default) 



16



The shell removes any quotes before `cat` sees them. So `cat -` and `cat "--` and `cat '-'` all get passed through as the array `['cat', '-']` after whitespace tokenization, wildcard expansion, and quote removal by the shell.

Share Improve this answer Follow

answered Feb 12, 2019 at 10:07



tripleee

7,112 ● 2 ● 30 ● 41



6



Quotes are use by the shell, and not passed to the command:

e.g.

```
cat "hello world" #this passes the single token 'hello world', to 'cat', as
argument 1. However the quotes are not passed.
cat "--" # this passes the single token '-', to 'cat', as argument 1. The same as
if no quotes had been used.
```

Share Improve this answer Follow

answered Feb 12, 2019 at 11:02

GNU `cat` is [coded](#) to compare the given filename against the string `"-"` before trying to open it as a file:

```
if (STREQ (infile, "-"))
{
    have_read_stdin = true;
    input_desc = STDIN_FILENO;
    if (file_open_mode & O_BINARY)
        xset_binary_mode (STDIN_FILENO, O_BINARY);
}
else
{
    input_desc = open (infile, file_open_mode);
    if (input_desc < 0)
    {
        error (0, errno, "%s", quote (infile));
        ok = false;
        continue;
    }
}
```


So, if you have a file named `-`, you need to defeat this logic by giving `cat` the *path and name*.

Quotation marks protect a value from white space splitting and, if single quotes, variable expansion. Quotation marks do not signal that the thing quoted is a file. To explicitly signal a value is a file, prefix it with a relative or absolute path.

All that said, one might suggest that GNU `cat` should also check if `-` is a file in the current working directory, but it'd be unusual for filenames to start with a hyphen or to be solely a hyphen, so that may be a performance concern. David Wheeler's essay, *Fixing UNIX and Linux Filenames*, has a [nice discussion](#) of this problem in historical context.

Share Improve this answer Follow

answered Feb 12, 2019 at 19:52

 bishop
2,906 ● 2 ● 14 ● 28

4 This behaviour (`-` designating standard input) is mandated by [the POSIX standard for the `cat` utility](#) and therefore not something that GNU `cat` should or

could change. – Kusananda ♦ Feb 12, 2019 at 19:55 ✎

@Kusananda GNU *could* [change](#) it: "The GNU Project regards standards published by other organizations as suggestions, not orders. We consider those standards, but we do not 'obey' them." With `POSIXLY_CORRECT` unset, GNU `cat` could also check for a file in the current working directory named `-` and use that file. That is likely the desired behavior, even though it countermands the letter of the standard. – bishop Feb 12, 2019 at 20:42 ✎

- 2 It would also possibly break existing code and/or introduce serious security issues (add a file called `-` and have it be injected in place of standard input in a pipeline). – Kusananda ♦ Feb 12, 2019 at 20:46