

Register to virtually attend our inaugural conference focused on our products, with relevant content for all developers everywhere.



Filename too long in Git for Windows

Asked 8 years, 5 months ago Modified 1 month ago Viewed 870k times



1152



326



I'm using `Git-1.9.0-preview20140217` for Windows. As I know, this release should fix the issue with too long filenames. But not for me.

Surely I'm doing something wrong: I did `git config core.longpaths true` and `git add .` and then `git commit`. Everything went well. But when I now do a `git status`, I get a list of files with `Filename too long`, for example:

```
node_modules/grunt-contrib-imagemin/node_modules/pngquant-bin/node_modules/bin-
wrapper/node_modules/download/node_modules/request/node_modules/form-
data/node_modules/combined-stream/node_modules/delayed-stream/test/integration/test-
handle-source-errors.js: Filename too long
```

It is quite simple to reproduce for me: just create a [Yeoman](#) web application with the Angular generator ("yo angular") and remove `node_modules` from the `.gitignore` file. Then repeat the aforementioned Git commands.

What am I missing here?

windows git

Share Edit Follow

edited Apr 8 at 7:22



StackOfZtuff

2,036 ● 26 ● 22

asked Mar 22, 2014 at 9:14



Papa Mufflon

15.2k ● 5 ● 25 ● 34

Where do you read that that version should fix the long filenames? – [iveqy](#) Mar 22, 2014 at 9:19

Here is the pull request for the patch: github.com/msysgit/git/pull/122 – [Papa Mufflon](#) Mar 22, 2014 at 9:31

@PapaMufflon can you change the accepted answer to the one with more score? It just helped me a lot. – [v.karbovnichy](#) Nov 2, 2017 at 15:28

@v.karbovnichy please read my question carefully. I already ran the command in the top voted answer. But at the time I asked the question, the accepted answer was correct: msys still had this character-limitation. Now that limitation is gone and `git config core.longpaths true` works like it should. – [Papa Mufflon](#) Nov 3, 2017 at 16:33

Ok, I agree then – [v.karbovnichy](#) Nov 5, 2017 at 18:20

Sorted by:

Trending sort available ⓘ

Highest score (default) ▾

18 Answers

▲
1517

Git has a limit of 4096 characters for a filename, except on Windows when Git is compiled with msys. It uses an older version of the Windows API and there's a limit of 260 characters for a filename.

▼
✓

So as far as I understand this, it's a limitation of msys and not of Git. You can read the details here: <https://github.com/msysgit/git/pull/110>



You can circumvent this by using another Git client on Windows or set `core.longpaths` to `true` as explained in other answers.

```
git config --system core.longpaths true
```

Git is build as a combination of scripts and compiled code. With the above change some of the scripts might fail. That's the reason for `core.longpaths` not to be enabled by default.

The windows documentation at <https://docs.microsoft.com/en-us/windows/win32/fileio/maximum-file-path-limitation?tabs=cmd#enable-long-paths-in-windows-10-version-1607-and-later> has some more information:

Starting in Windows 10, version 1607, MAX_PATH limitations have been removed from common Win32 file and directory functions. However, you must opt-in to the new behavior.

A registry key allows you to enable or disable the new long path behavior. To enable long path behavior set the registry key at `HKLM\SYSTEM\CurrentControlSet\Control\FileSystem\LongPathsEnabled` (Type: `REG_DWORD`)

Share Edit Follow

edited Aug 3 at 6:05



Grim

3,200 ● 9 ● 53 ● 113


answered Mar 22, 2014 at 9:24



iveqy

16.1k ● 1 ● 13 ● 20


49 The limitation to 260 chars in a path is not specific to MSYS, it's a general Windows API imitation. This can be worked around by using Unicode paths, but that has other drawbacks, which is why `core.longpaths` is not enabled by default. Also note that Git for Windows it not compiled against MSYS. Instead, it's a native Windows application that comes with a stripped-down MSYS environment. – [sschuberth](#) Feb 14, 2015 at 11:03

- / @sschubertn: Are there any drawbacks other than lack of compatibility with programs that do not support long paths? – JAB Sep 6, 2015 at 20:14 
-
- 7 @JAB Another drawback is that long paths always have to be absolute; relative paths are not supported. For further details please see [here](#). – sschubertn Sep 6, 2015 at 20:38
-
- 5 Or as a quick fix, just try checkout your repo to C:/ on windows thus reducing number of folder path characters. – Akshay Lokur Nov 16, 2016 at 3:49
-
- 7 As of Windows 10, you can edit your registry (or apply a group policy) to remove the Windows API filename length restriction. [howtogeek.com/266621/...](http://howtogeek.com/266621/) – jameslafferty Aug 30, 2018 at 18:27
-

▲ You should be able to run the command

1176

```
git config --system core.longpaths true
```

▼
 or add it to one of your Git configuration files manually to turn this functionality on, once you are on a supported version of Git. It looks like maybe 1.9.0 and after.

Share Edit Follow

edited Nov 5, 2018 at 21:25

answered Sep 30, 2014 at 0:51




Peter Mortensen

30.4k ● 21 ● 102 ● 124



sparkym3


11.9k ● 2 ● 11 ● 3

-
- 14 This config option fixed the issue for me, even with msys as mentioned in the accepted answer. (Specifically, version 1.9.4.msysgit.2). – Alex Osborn Dec 15, 2014 at 21:15
-
- 6 Sourcetree acts a bit weird unless you "also make sure that SourceTree is using the System's Git and not the embedded one." - Thanks to [Matej Drolic](#) for that advice – bstoney Jan 9, 2015 at 1:16
-
- 47 [Here](#) is some background information why this is not enabled by default, and some technical details. – sschubertn Feb 14, 2015 at 11:06
-
- 13 get "could not lock config file C:\Program Files\Git\mingw64/etc/gitconfig" after running command above. But @Yash answer worked for me – divideByZero Oct 7, 2016 at 9:15 
-
- 13 @divideByZero running git bash as administrator prevents that error. – Niek Oct 18, 2016 at 17:34
-

▲ This might help:

319

```
git config core.longpaths true
```

▼
 Basic explanation: This answer suggests not to have such setting applied to the global system (to all projects so avoiding `--system` or `--global` tag) configurations. This command only solves the problem by being specific to the current project.

EDIT:

This is an important answer related to the "permission denied" issue for those whom does not granted to change git settings globally.

Share Edit Follow

edited Jan 20, 2021 at 12:14

answered Mar 5, 2016 at 10:38



Akif

6,401 ● 7 ● 21 ● 49



Sagiruddin Mondal

4,929 ● 2 ● 28 ● 44

- 18 Folks here have noted that this setting can introduce some unpredictable behavior so it seems that it's preferable to use the above command as a local setting on projects where that require it rather than appending `--system` which will apply it to all projects – [Grant Humphries](#) Jun 17, 2016 at 19:34
- 6 hey, that's just a copy-paste of the other highly upvoted answer. might at the very least explain why you prefer removing the `--system` option.. – [Félix Adriel Gagnon-Grenier](#) Oct 29, 2016 at 16:23 ✎
- 2 I didn't have elevated rights and thus this was much easier to do inside the git repository than to ask the IT team to run the global command with elevated rights. Thanks Sagiruddin! – [Shayan Ahmad](#) Aug 18, 2021 at 10:48

Steps to follow (Windows):

187

1. Run **Git Bash** as **administrator** (right-clicking the app shortcut will show the option to Run as Administrator)

2. Run the following command:

```
git config --system core.longpaths true
```

Note: if step 2 does not work or gives any error, you can also try running this command:

```
git config --global core.longpaths true
```

Read more about `git config` [here](#).

Share Edit Follow

edited Oct 21, 2021 at 14:33

answered Mar 3, 2018 at 4:50



Saikat


12k ● 16 ● 96 ● 114

- 8 `git config --global core.longpaths true` saved my day. Thank you – [moshiuramit](#) Nov 17, 2021 at 5:49 ✎

Create .gitconfig and add

117

```
[core]
longpaths = true
```

 You can create the file in a project location (not sure) and also in the global location. In my case the location is `C:\Users\{name}\`.

Share Edit Follow

edited Nov 5, 2018 at 21:27

answered Apr 16, 2016 at 11:55



Peter Mortensen

30.4k ● 21 ● 102 ● 124




Yash

6,024 ● 3 ● 35 ● 25


14 You can also do this with the following command: `git config --global core.longpaths true`

– SoftWyer Nov 9, 2016 at 13:05 

`git config --global core.longpaths true` worked for me thanks – Rama Krshna Ila Sep 29, 2017 at 22:13

2 Using Visual Studio the git bash solutions above did not work for me, but finding the `.git/config` file for the project and editing as shown above did. Thanks yash. – andrew pate May 2, 2018 at 13:23 

this worked for me, i located that file and modified it manually – Patlatus Nov 1, 2018 at 12:44

1 The above mentioned and verified answers are correct but with the permissions which is granted to the file, it might not be possible to update the file with those commands. This approach is really easy because this is the manual approach and it worked for me really well. You can easily find the `.gitconfig` file in the following path `C:\Users\{username}` and simply edit it. – Kavindu N Jul 30, 2019 at 4:19 

▲ To be entirely sure that it takes effect immediately after the repository is initialized, but before the remote history is fetched or any files checked out, it is safer to use it this way:

49

```
git clone -c core.longpaths=true <repo-url>
```



-c key=value

Set a configuration variable in the newly-created repository; this takes effect immediately after the repository is initialized, but before the remote history is fetched or any files checked out. The key is in the same format as expected by `git-config`¹ (e.g., `core.eol=true`). If multiple values are given for the same key, each value will be written to the config file. This makes it safe, for example, to add additional fetch refsspecs to the origin remote.

[More info](#)

Share Edit Follow

answered Dec 1, 2016 at 11:26



Watchmaker

4,458 ● 1 ● 33 ● 37

▲ The better solution is enable the longpath parameter from Git.

37

```
git config --system core.longpaths true
```

But a workaround that works is remove the node_modules folder from Git:

```
$ git rm -r --cached node_modules
$ vi .gitignore
```

Add node_modules in a new row inside the .gitignore file. After doing this, push your modifications:

```
$ git add .gitignore
$ git commit -m "node_modules removed"
$ git push
```

Share Edit Follow

edited Nov 5, 2018 at 21:31



Peter Mortensen

30.4k ● 21 ● 102 ● 124

answered Aug 22, 2016 at 18:44



Janderson Silva

1,707 ● 16 ● 14

- 3 There's a good reason to keep the node_modules folder checked into git: If you want your software to behave the same after a year of modules potentially vanishing from npm. – [cfstras](#) Aug 24, 2016 at 15:00

@cfstras if some library has a vulnerability and you don't update periodically, certainly you'll have security problems. – [Janderson Silva](#) Aug 26, 2016 at 12:31

- 1 Of course you have to upgrade your dependencies. But only when *you* want to, and if something were to break, you would want your backup in git... – [cfstras](#) Aug 26, 2016 at 14:47

Is true. I'll edit my answer. Thank you for your comment. – [Janderson Silva](#) Aug 26, 2016 at 16:40

- 5 No need to commit node_modules : the packages.lock file is here to ensure the version installed by npm install will always be the same, until you make a npm update – [Pierre-Olivier Vares](#) Oct 17, 2019 at 14:28

Executing `git config --system core.longpaths true` thrown an error to me:

31

"error: could not lock config file C:\Program Files (x86)\Git\mingw32/etc/gitconfig: Permission denied"

Fixed with executing the command at the global level:

```
git config --global core.longpaths true
```

Share Edit Follow

answered Dec 20, 2018 at 9:04



Arpit Aggarwal

25.6k ● 14 ● 84 ● 102

The global settings affect only the current user, whereas system settings affect all the users on the machine.

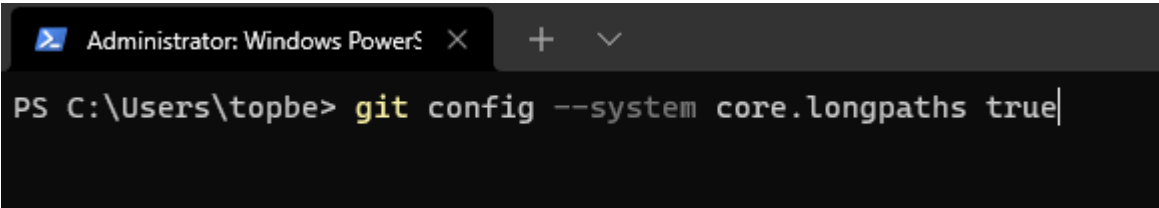
If this is your workstation, they're effectively the same as you may use only one user. – [towel](#) Jan 15, 2019 at

if this is your workstation they're effectively the same as you may use only one user. – [lower](#) Jan 13, 2019 at 17:14

- 5 If you're command line application Ran as Administrator, first command would work! – [Sachith Dickwella](#) Mar 7, 2019 at 5:13

▲ This worked for me

31



```
Administrator: Windows PowerShell
PS C:\Users\topbe> git config --system core.longpaths true
```

Run as terminal as **administrator**. And run the command below.

```
git config --system core.longpaths true
```

Share Edit Follow

answered Nov 22, 2021 at 6:09



[Avocado](#)
396 ● 3 ● 3

```
git config --global core.longpaths true
```

17

▲ The above command worked for me. Using '--system' gave me config file not locked error

▼ Share Edit Follow

answered Nov 26, 2019 at 14:18



[amalik2205](#)
3,586 ● 1 ● 13 ● 20

- 2 for Github Desktop users, this is the only one that works because Github Desktop uses its own Git config. – [Csa77](#) Nov 30, 2019 at 10:38

▲ You could also try to enable long file paths.

16

▼ If you run Windows 10 Home Edition you could change your Registry to enable long paths.

Go to `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\FileSystem` in `regedit` and then set `LongPathsEnabled` to `1`.

↻ If you have Windows 10 Pro or Enterprise you could also use Local Group Policies.

Go to *Computer Configuration* → *Administrative Templates* → *System* → *Filesystem* in `gpedit.msc`,

open *Enable Win32 long paths* and set it to *Enabled*.

Share Edit Follow

edited Nov 5, 2018 at 21:35



Peter Mortensen

30.4k ● 21 ● 102 ● 124

answered Sep 3, 2018 at 10:36



Julian Veerkamp

1,504 ● 1 ● 16 ● 20

8 I believe this must be done in combination with the git config, and it's worth noting it doesn't work with Windows Explorer for the reasons mentioned [here](#). – Neo Sep 4, 2018 at 10:31

Win32 - I wonder if this can help if the application is 64bit? – Henning Larsen Feb 18 at 20:32



14



- Download & Install Git bash from here: <https://git-scm.com/download/win>
- Run the git bash gui as administrator and run this command: `git config --system core.longpaths true`
- Now clone any repository.
- If the problem is not fixed try this command: `git config --global core.longpaths true`
- If it does not help try restarting the windows.

Share Edit Follow

answered Apr 7 at 3:08



Md. Shahariar Hossen

861 ● 6 ● 10



12



TortoiseGit (Windows)

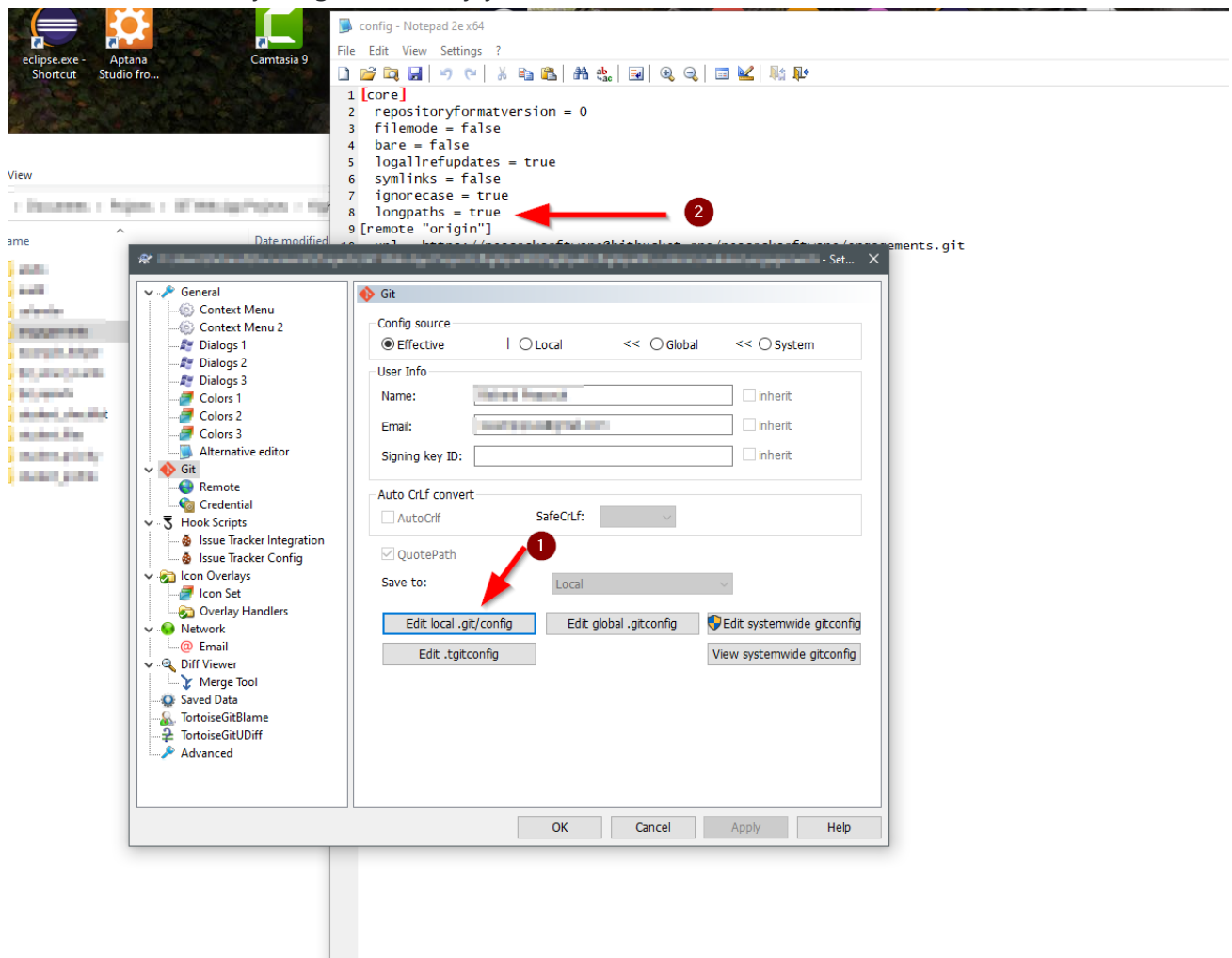
For anyone using TortoiseGit for Windows, I did this:

(1) Right-click on the folder containing your project. Select TortoiseGit -> Settings.

(2) On the "Git" tab, click the button to "Edit local .git/config".

(3) In the text file that pops up, under the [core] section, add: longpaths = true

Save and close everything, then re-try your commit. For me, this worked.



I hope this minimizes any possible system-wide issues, since we are not editing the global .gitconfig file, but rather just the one for this particular repository.

Share Edit Follow

answered Dec 4, 2020 at 20:03



Richard

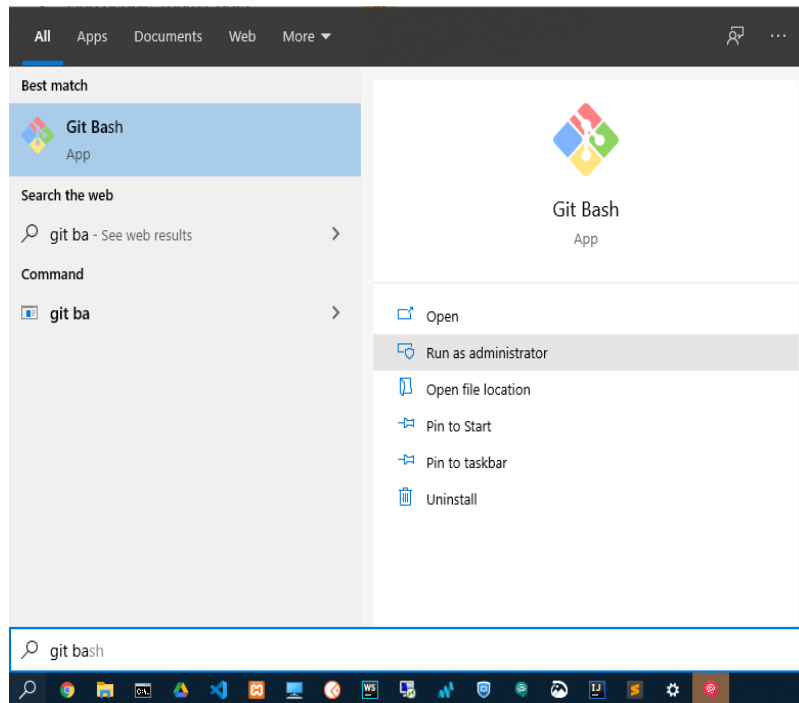
1,772 ● 18 ● 26

In Windows, you can follow these steps which worked for me.

7

1. Open your cmd or git bash as an administrator





2. Give the following command either from cmd or git bash which you ran above as an administrator

```
git config --system core.longpaths true
```

3. This will allow accessing long paths globally
4. And now you can clone the repository with no issues with long paths

Share Edit Follow

edited Nov 28, 2020 at 5:27

answered Nov 28, 2020 at 5:18



Niroshan Ratnayake

2,857 ● 2 ● 14 ● 17

▲ Move repository to root of your drive (temporary fix)

- 7 You can try to temporarily move the local repository (the entire folder) to the root of your drive or as close to the root as possible.



Since the path is smaller at the root of the drive, it sometimes fixes the issues.

On Windows, I'd move this to `c:\` or another drive's root.

Share Edit Follow

edited Nov 5, 2018 at 21:32

answered Jul 27, 2017 at 12:35



Peter Mortensen

30.4k ● 21 ● 102 ● 124



Dheeraj Bhaskar

18.1k ● 9 ● 63 ● 66

- 2 This is the only thing that solved my issue. It was that I had too many folders in the path. – J Brune Apr 26, 2018 at 20:26

In a windows Machine

- 2 Run Command Prompt as administrator then run below command

```
git config --system core.longpaths true
```

Share Edit Follow

answered May 6, 2020 at 4:56



[kartick shaw](#)

755 ● 11 ● 5

I had this error too, but in my case the cause was using an outdated version of npm, v1.4.28.

- 2 Updating to npm v3 followed by

```
rm -rf node_modules  
npm -i
```

worked for me. npm issue 2697 has details of the "maximally flat" folder structure included in npm v3 (released 2015-06-25).

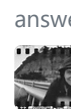
Share Edit Follow

edited Nov 5, 2018 at 21:26



[Peter Mortensen](#)

30.4k ● 21 ● 102 ● 124



answered Nov 2, 2015 at 12:25

[James Green](#)

1,807 ● 1 ● 14 ● 13

If you are working with your encrypted partition, consider moving the folder to an unencrypted partition, for example a **/tmp**, running `git pull`, and then moving back.

1

Share Edit Follow

edited Nov 5, 2018 at 21:32



[Peter Mortensen](#)

30.4k ● 21 ● 102 ● 124



answered Feb 20, 2018 at 22:51

[augustowebd](#)

382 ● 2 ● 8



Highly active question. Earn 10 reputation (not counting the [association bonus](#)) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.