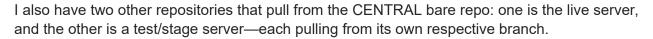
How do I force git pull to overwrite everything on every pull?

Asked 8 years, 2 months ago Active 2 years, 11 months ago Viewed 301k times



I have a CENTRAL bare repository that has three developer repositories pulling and pushing to it normally.

201





The scenario is this: I have a post-update hook script on the CENTRAL repo that automatically accesses the test and live repos and runs a pull command on each. This updates both test and live servers, all depending on what branch has new commits. This all works great.



The problem is this: there may be times in an emergency that files may be directly updated on the server (via ftp or whatever) and the CENTRAL post-update script will then fail since merge/overwrite conflicts will occur. There is no way to avoid this scenario, and it is inevitable.

What I would like to have happen is this: I want the pull from the live and test sites to *always* overwrite/merge on pull. *Always*. These repos will be pull-only as they are not for development.

In all my research, I cannot find a good solution to have a pull *always* force an overwrite of the local files. Is this at all possible? It would make for a great development scenario if so.

git

git-pull

edited May 31 '17 at 9:28

Mel

4,347 • 9 • 30 • 33

asked Mar 6 '12 at 18:36

bmilesp
2.209 • 3 • 12 • 13

- While I've voted for the 'reset to what you just fetched' answer below, I think the solution to your real problem is to not make out-of-band changes. Modifications, no matter how urgent, should *always* go through version control. No one except operators should have direct access to the running sites (e.g. not developers). Using version control consistently means that you have a record of when changes were made, and who made them, and better tools to work with them. Why subvert it, for no real benefit? Phil Miller Mar 6 '12 at 18:54
- @Novelocrat right, i understand what you are saying. Unfortunately, there are a number of scenarios where someone could upload a file directly to the server. In that case i would need to run a number of commands to re-sync the repos. Previously we used an FTP script to move files from the repo to the server. The proposed method above would simply eliminate the FTP step, which has worked very well in the past. bmilesp Mar 6 '12 at 22:45
- So, don't let people access the server directly. Lock out FTP and SSH access, or tell them they'll be fired for

By using our site, you acknowledge that you have read and understand our <u>Cookie Policy</u>, <u>Privacy Policy</u>, and our <u>Terms of Service</u>.



7 Answers





Really the ideal way to do this is to not use pull at all, but instead fetch and reset:

503

```
git fetch origin master
git reset --hard FETCH_HEAD
git clean -df
```



(Altering master to whatever branch you want to be following.)



pull is designed around merging changes together in some way, whereas reset is designed around simply making your local copy match a specific commit.

You may want to consider slightly different options to clean depending on your system's needs.







- @user730569 reset --hard is a command which is used to force the state of the working directory (and the current branch) to a state matching that of a particular commit. Amber Aug 30 '12 at 7:10
- 24 FETCH_HEAD is a reference that is automatically created by fetch to represent the fetched ref. It's not merged, just straight-up overwritten whenever you do a fetch. clean is a command that removes files which are not tracked by git, the -df flags tell it to remove directories (-d) and actually do the removal (-f). Amber Aug 30 '12 at 17:14
- 4 why isn't there a keyword for this? I need this much more often than pull. Wolfgang Fahl May 28 '13 at 9:03
- 14 You might want to use git clean -dn before using git clean -df so you will see what files/folders will get deleted. git clean -df can only be reversed if you had a backup Ibrahim Lawal Sep 13 '13 at 8:12
- @NickMiddleweek I was worried git clean -df will remove gitignored files too, but turns out it won't. git clean --help says "Normally, only files unknown to Git are removed, but if the -x option is specified, ignored files are also removed. This can, for example, be useful to remove all build products." nickang Aug 24 '17 at 2:21



```
git reset --hard HEAD
git fetch --all
git reset --hard origin/your_branch
```

5

By using our site, you acknowledge that you have read and understand our <u>Cookie Policy</u>, <u>Privacy Policy</u>, and our <u>Terms of Service</u>.



If you haven't commit the local changes yet since the last pull/clone, you can use:



```
git checkout *
git pull
```

checkout will clear your local changes with the last local commit, and pull will sincronize it to the remote repository

answered Aug 22 '16 at 17:47





To pull a copy of the branch and **force overwrite of local files** from the origin use:

5

git reset --hard origin/current_branch



All current work will be lost and it will then be the same as the origin branch

1

Andrew Atkinson 3.411 • 4 • 35 • 45



You could try this:

22

git reset --hard HEAD
git pull



(from How do I force "git pull" to overwrite local files?)

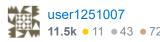
1

Another idea would be to delete the entire git and make a new clone.

edited May 23 '17 at 12:18



answered Mar 6 '12 at 18:42





I'm not sure how to do it in one command but you could do something like:



git reset --hard
git pull



or even



By using our site, you acknowledge that you have read and understand our <u>Cookie Policy</u>, <u>Privacy Policy</u>, and our <u>Terms of Service</u>.



answered Mar 6 '12 at 18:40



To run in one command: git reset --hard && git pull . Alternatively, but not better, git reset --hard; git pull . Using && will only run the second command if the first command was successful.; will run it regardless of exit code of the first command. — mazunki Nov 25 '19 at 19:34



You can change the hook to wipe everything clean.



Danger! Wipes local data!



Remove all local changes to tracked files git reset --hard HEAD



Remove all untracked files and directories
git clean -dfx
git pull ...

answered Mar 6 '12 at 18:40



- what does x do? please explain the switches stevek Sep 4 '15 at 9:40
- The x is supposed to remove all untracked files, I think. Hard to tell in manpage-speak, which is why we have SO. JosephK Jul 26 '16 at 4:42
- @JosephK: That's incorrect. The basic purpose of git clean is already "Remove untracked files from the working tree" (top of the page). Normally, this does not include ignored files, but -x tells git clean to include ignored files as well (except this does not affect files ignored by the -e option). Dietrich Epp Jul 26 '16 at 14:29