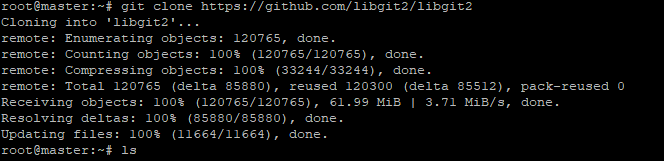
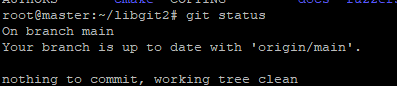
Git commands

1. git clone <https://github.com/libgit2/libgit2>



*root@master:~# cd libgit2*

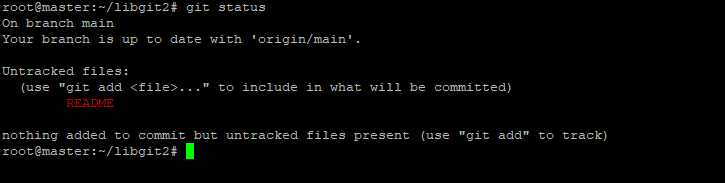
1. *root@master:~/libgit2#* ***git status***



1. Let’s say you add a new file to your project, a simple README file. If the file didn’t exist before, and you run git status, you see your untracked file like so:

echo 'My Project' > README

git status

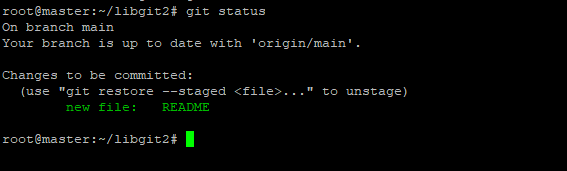


You can see that your new README file is untracked, because it’s under the “Untracked files” heading in your status output. Untracked basically means that Git sees a file you didn’t have in the previous snapshot (commit), and which hasn’t yet been staged; Git won’t start including it in your commit snapshots until you explicitly tell it to do so. It does this so you don’t accidentally begin including generated binary files or other files that you did not mean to include. You do want to start including README, so let’s start tracking the file.

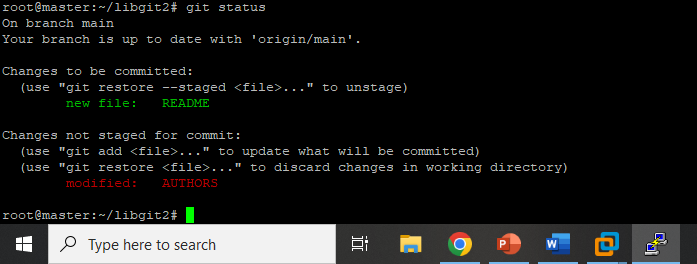
**Tracking New Files**

In order to begin tracking a new file, you use the command git add. To begin tracking the README file, you can run this:

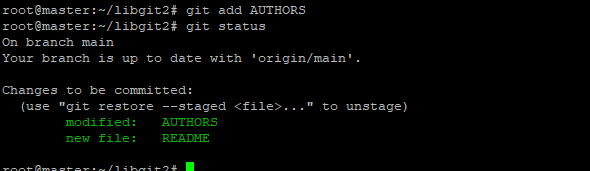
$ git add README



If you modify existing file, check the “git status” output

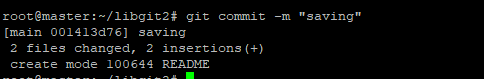


*git add AUTHORS*

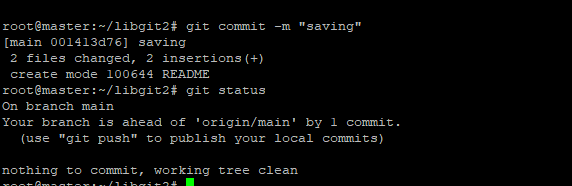


Git commit lab

*git commit -m "saving"*



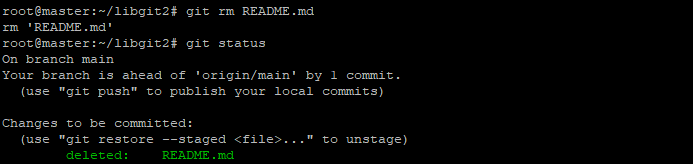
*git status*



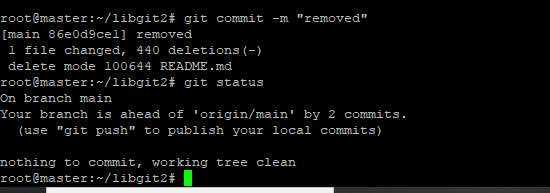
Removing files

*git rm README.md*

*git status*



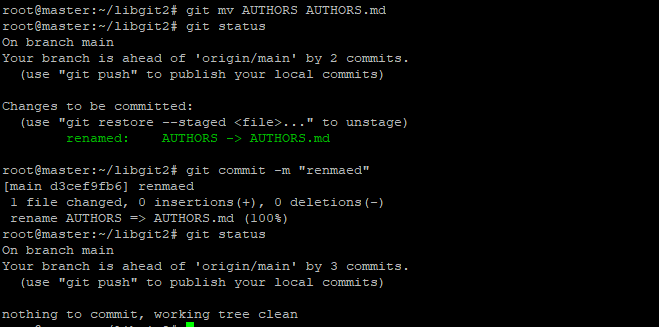
*git commit -m "removed"*



*Move /rename files*

*git mv AUTHORS AUTHORS.md*

*git status*



Check the history

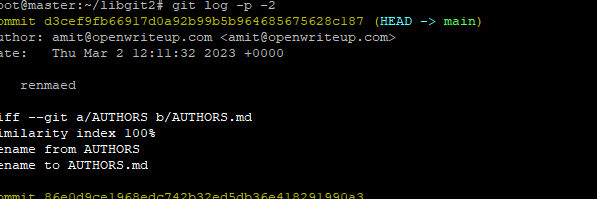
git log command

*git log*



Check two patch history

*git log -p -2*

git log –stat

