

# Make a Pull Request (delete)

Learn how to perform a “pull request.”

We'll cover the following



- Make a pull request
- Process to create a pull request
- Pull requests in practice: rebasing

## Make a pull request #

Now that you have a branch on the forked repository on GitHub, you want to get that branch's changes into the maintainer's repository. This is where you raise the pull request.

Go to GitHub in a browser and view your repository:

<https://github.com/YOURUSERNAME/learngitthehardway>

(<https://github.com/YOURUSERNAME/learngitthehardway>)

The instructions for creating a pull request are here:

<https://help.github.com/articles/creating-a-pull-request/>

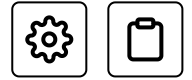
(<https://help.github.com/articles/creating-a-pull-request/>)

## Process to create a pull request #

I won't repeat it here because the workflow can change. But in essence, the general process is to:

- Go to your branch.
- Generate a new pull request.
- Fill out the form.

- Wait.



- Celebrate your PR's acceptance into the code, or chase the maintainer (nicely!) for an update.

You can create a pull request *across forks* (a request to the upstream (original) repository) or against another branch in a GitHub repository. “*Across forks*” is what’s most commonly meant by a public GitHub PR, which is a request to accept a change made to a repository under your control to a repository under someone else’s (usually more *senior* to the project).

## Pull requests in practice: rebasing #

Maintainers will often ask that you rebase your branch to the principal branch (usually `master`) before making a pull request. You will remember the `git rebase`. If you don’t remember, then you might want to go back and read over it again.

Maintainers will want you to rebase so that the work of merging any changes made, since you forked from the origin, is done by you, who is the submitter rather than them. This also makes the history of the main line easier.

If you didn’t understand the above paragraph, then definitely work through the rebase chapter

(<https://www.educative.io/collection/page/10370001/6075350136651776/6228325110906880>) again!

The goal is that all the messy work is done on your secondary branch (which in Git is a more disposable thing) and the good stuff makes its way into the main line. Many projects will delete branches once they have served their purpose, and Git supports this.

```
17 git branch -d mybranch
```

It will even warn you if the branch has not been merged into the branch

you are currently on!



```
18 git branch -d abranch
```

### Output:

```
error: The branch 'abbranch' is not fully merged.  
If you are sure you want to delete it, run 'git branch -D abbranch'  
.
```

[< Back](#)[Next >](#)

Forking And Branching in GitHub

Assessment #3 - Remote Repository ...



Mark as Completed



Report an  
Issue



Ask a Question

([https://discuss.educative.io/tag/make-a-pull-request-delete\\_\\_pull-requests\\_\\_learn-git-the-hard-way](https://discuss.educative.io/tag/make-a-pull-request-delete__pull-requests__learn-git-the-hard-way))