

Forking A Repository

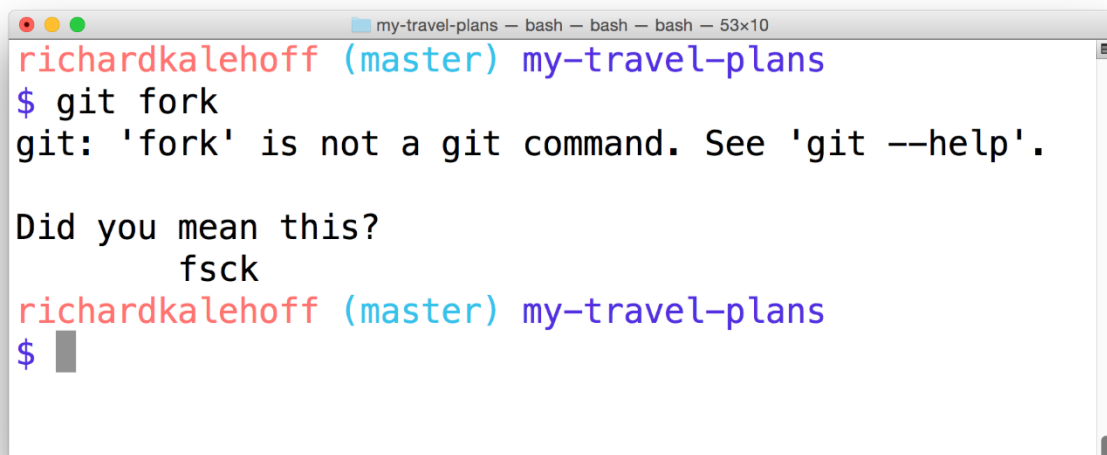
In version control terminology if you "fork" a repository that means you *duplicate* it. Typically you fork a repository that belongs to someone else. So you make an identical copy of *their* repository and that duplicate copy now belongs to *you*.

This concept of "forking" is also different from "cloning". When you clone a repository, you get an identical copy of the repository. But cloning happens on your *local* machine and you clone a *remote* repository. When you fork a repository, a new duplicate copy of the *remote* repository is created. This new copy is *also a remote* repository, but it now belongs to you.

A `fork` Subcommand?

Forking is not done on the command line; there is no `git fork` command. Go ahead, try running the following command:

```
$ git fork
```

A terminal window titled 'my-travel-plans — bash — bash — bash — 53x10' shows a user named 'richardkalehoff' in the '(master)' branch of the 'my-travel-plans' repository. The user enters '\$ git fork'. The terminal responds with 'git: 'fork' is not a git command. See 'git --help'.' It then suggests 'Did you mean this?' followed by 'fsck'. The user then enters '\$' and the prompt returns.

```
richardkalehoff (master) my-travel-plans
$ git fork
git: 'fork' is not a git command. See 'git --help'.

Did you mean this?
    fsck
richardkalehoff (master) my-travel-plans
$
```

Running the (nonexistent!) `git fork` command on the terminal. Git doesn't have a `fork` subcommand, so it responds with a suggestion to use an alternative command.

As you can see, trying to run the `git fork` command produces an error. (Also, `fsck` is not a rude word, it means "filesystem check" and refers to auditing the files for consistency.)

Alter a Cloned Repo

Do the following steps and answer the question below.

Question 1 of 2

Were you able to successfully push your changes to the remote repository of others cloned Repo?

- No

Question 2 of 2

One of the lines that was displayed after you tried to push includes the word "fatal". What comes right after that word?

- unable to access

We can see from this little experiment that if a repository doesn't belong to your account then it means you do not have permission to modify it.

This is where forking comes in! Instead of modifying the original repository directly, if you fork the repository to your own account then you will have full control over that repository.

Forking Lam's Project

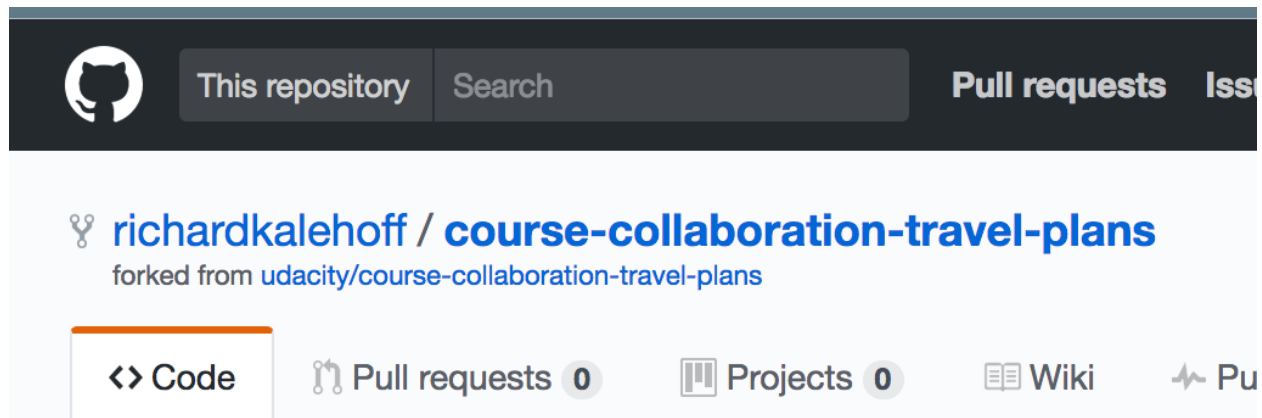
I'm going to clone Lam's project myself. My account does not have permission to edit her repository directly, so I'll fork the repository to my own account.

I want you to sign into your GitHub account and follow along with the rest of these steps:

Task List

- go to <https://github.com/udacity/course-collaboration-travel-plans> and fork it.

Let's take a look at the name of the repository:



Forking a project displays the new project name next to your GitHub profile name. Also, below that it says where the original project exists.

See how this shows my account name (richardkalehoff) and the name of the repository? But then just beneath that it says "forked from udacity/course-collaboration-travel-plans". This shows that this project is in _my_ account but that it has a connection to the original project that it was copied from.

That's pretty neat, right?!? You can fork any public repository that's up on GitHub right now - which means you can get a copy of that repository in your own account that you will have total control over.

Why don't you go fork a few repositories just to get some practice! Here a few you can try it out on:

- <https://github.com/udacity/course-git-blog-project>

- <https://github.com/udacity/frontend-nanodegree-styleguide>
- <https://github.com/GoogleChrome/lighthouse>
- <https://github.com/jquery/jquery>

Push/Pull To The Fork

Because forking a repository gives you a copy of it in your account, you can clone the repo down to your computer, make changes to it, and then push those changes back to the forked repository. But you need to keep in mind that it'll be pushing the changes back to *your* remote repository not to the *original* remote repository that you forked from.

Recap

Forking is an action that's done on a hosting service, like GitHub. Forking a repository creates an identical copy of the original repository and moves this copy to your account. You have total control over this forked repository. Modifying your forked repository does not alter the original repository in any way.