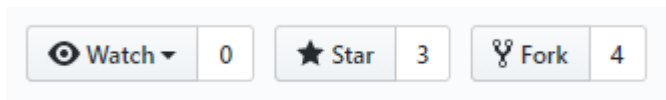


Lab 4a - Forking a repository

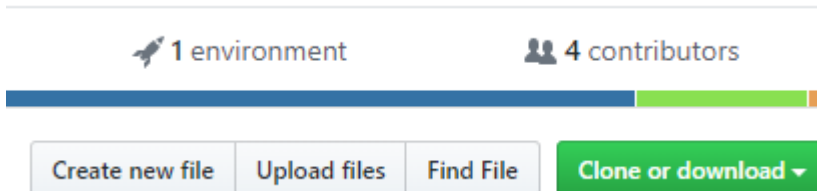
Note: For the sake of these tutorials, we are assuming you are using a terminal capable of running Bash commands. For Windows users, the recommended terminal program for this is Git Bash, which you can download [here](#).

This practice gets you to fork a remote repository.

1. Make sure you're logged into your GitHub account
2. Go to any public repository available. This one for example: <https://github.com/qa-apprenticeships/FlaskAppBasic>
3. Click on **Fork** in the top right of the page for your chosen repository. This will create a copy of the repository under your account



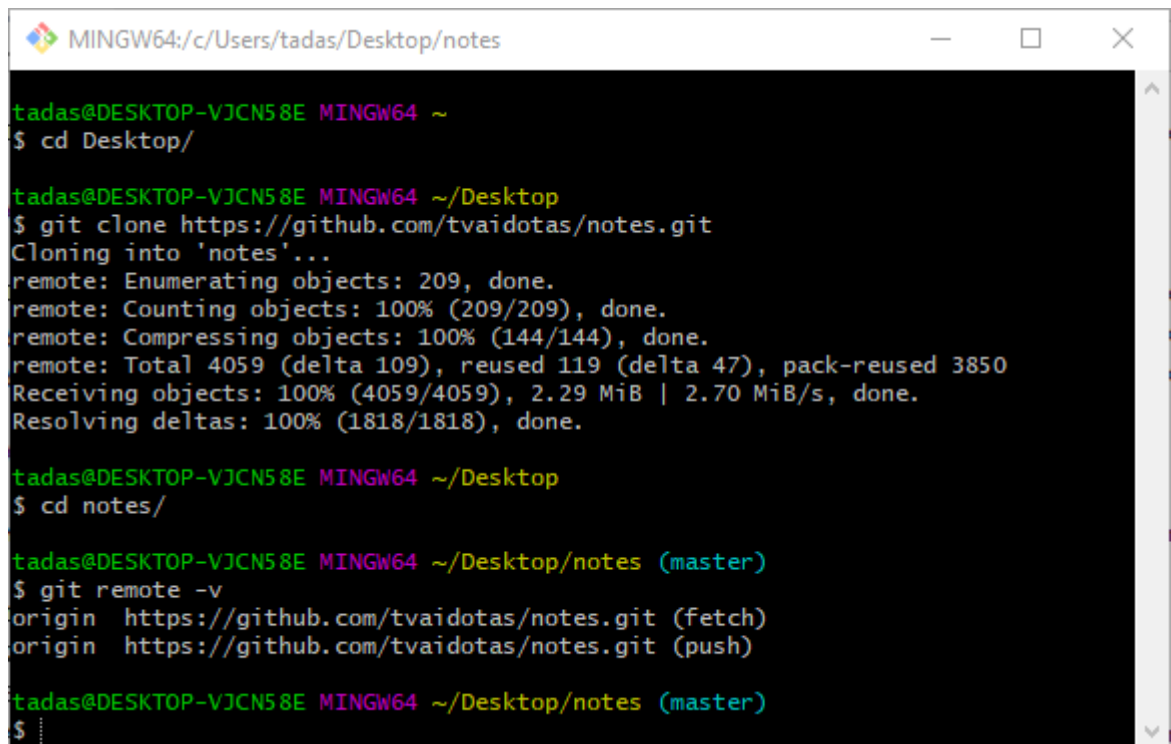
4. You will be redirected to your account's version of the repository
5. Click on the green **Clone or download** button, and copy the URL



6. Open a Bash terminal in the location you want to clone your project
7. Run the command `git clone [URL]`, but replace `[URL]` with the repository URL you copied in the previous step
8. Change directory to the project you just cloned
9. Run the following command, to confirm that you have cloned the correct repository:

```
$ git remote -v
```

You should get a similar output for fetch and push, but the URL will be pointing to your repository.



```
MINGW64:/c/Users/tadas/Desktop/notes

tadas@DESKTOP-VJCN58E MINGW64 ~
$ cd Desktop/

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop
$ git clone https://github.com/tvaidotas/notes.git
Cloning into 'notes'...
remote: Enumerating objects: 209, done.
remote: Counting objects: 100% (209/209), done.
remote: Compressing objects: 100% (144/144), done.
remote: Total 4059 (delta 109), reused 119 (delta 47), pack-reused 3850
Receiving objects: 100% (4059/4059), 2.29 MiB | 2.70 MiB/s, done.
Resolving deltas: 100% (1818/1818), done.

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop
$ cd notes/

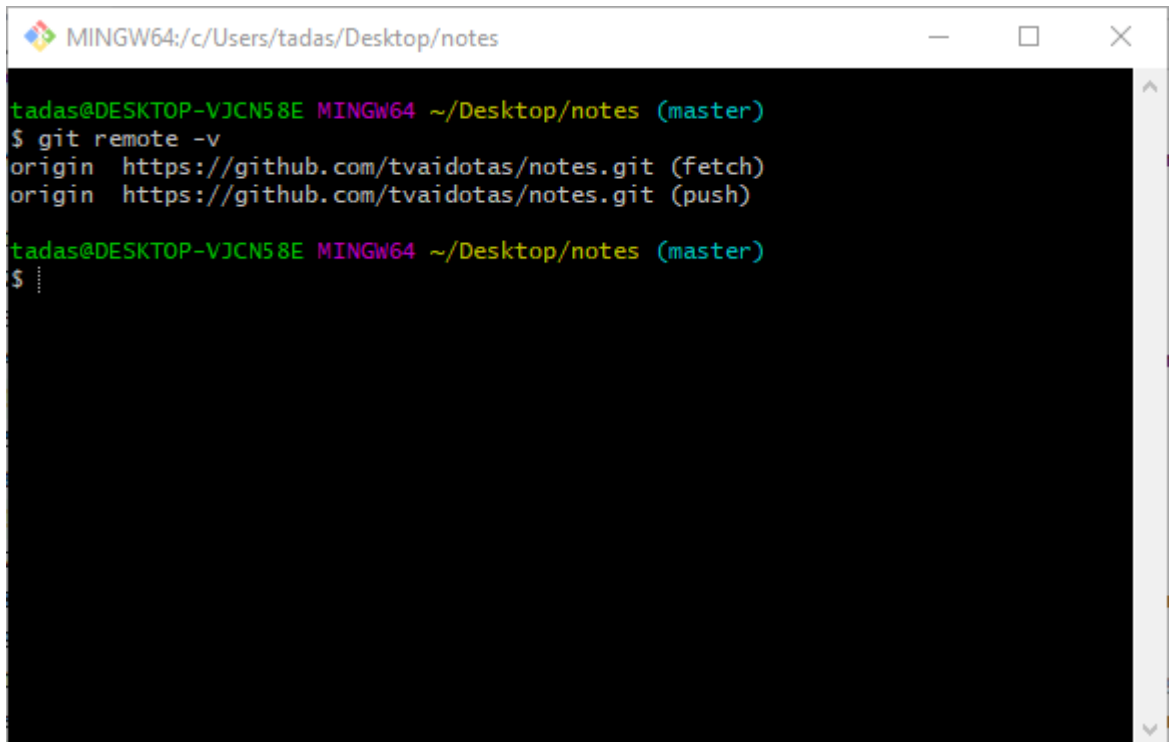
tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$ git remote -v
origin https://github.com/tvaidotas/notes.git (fetch)
origin https://github.com/tvaidotas/notes.git (push)

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$
```

Lab 4b - Updating forked repository from original

Now you will set up your local project to receive updates from the original repository. This is required, so that when the owner of the original repository adds new functionality, bug fixes etc., you will be able to get these changes as well.

1. Open your Bash terminal, in the root directory of your project. You should be able to see that you're on the master branch. Next, you want to execute `git remote -v` to make sure that the upstream to the original repository is not set up yet. You should see a similar output to this.

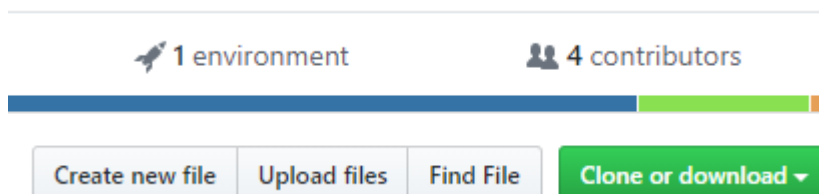
A screenshot of a MINGW64 terminal window. The title bar shows the path 'MINGW64:/c/Users/tadas/Desktop/notes'. The terminal content shows a user named 'tadas@DESKTOP-VJCN58E' in the 'MINGW64' environment at the directory '~/Desktop/notes' on the 'master' branch. The user has executed the command 'git remote -v', which outputs two lines: 'origin https://github.com/tvaidotas/notes.git (fetch)' and 'origin https://github.com/tvaidotas/notes.git (push)'. The prompt '\$' is visible at the bottom.

```
MINGW64:/c/Users/tadas/Desktop/notes

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$ git remote -v
origin https://github.com/tvaidotas/notes.git (fetch)
origin https://github.com/tvaidotas/notes.git (push)

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$
```

2. In your browser, go to the original git repository and copy the repository URL. You can do this by clicking on the green **Clone or download** button and copying the URL



3. Execute the following command in your Bash terminal, but replace the [URL] with the one you just copied:

```
git remote add upstream [URL]
```
4. Next, you want to check that this has worked, which you can do by executing the `git remote -v` command. The output should look similar to this:

```
MINGW64:/c/Users/tadas/Desktop/notes

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$ git remote -v
origin https://github.com/tvaidotas/notes.git (fetch)
origin https://github.com/tvaidotas/notes.git (push)

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$ git remote add upstream https://github.com/bob-crutchley/notes.git

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$ git remote -v
origin https://github.com/tvaidotas/notes.git (fetch)
origin https://github.com/tvaidotas/notes.git (push)
upstream https://github.com/bob-crutchley/notes.git (fetch)
upstream https://github.com/bob-crutchley/notes.git (push)

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$ .....
```

5. Now you will pull the changes from the original repository into yours, which you can do by executing the `git fetch upstream` command in your Bash terminal. The output will depend on whether there are new changes or not:

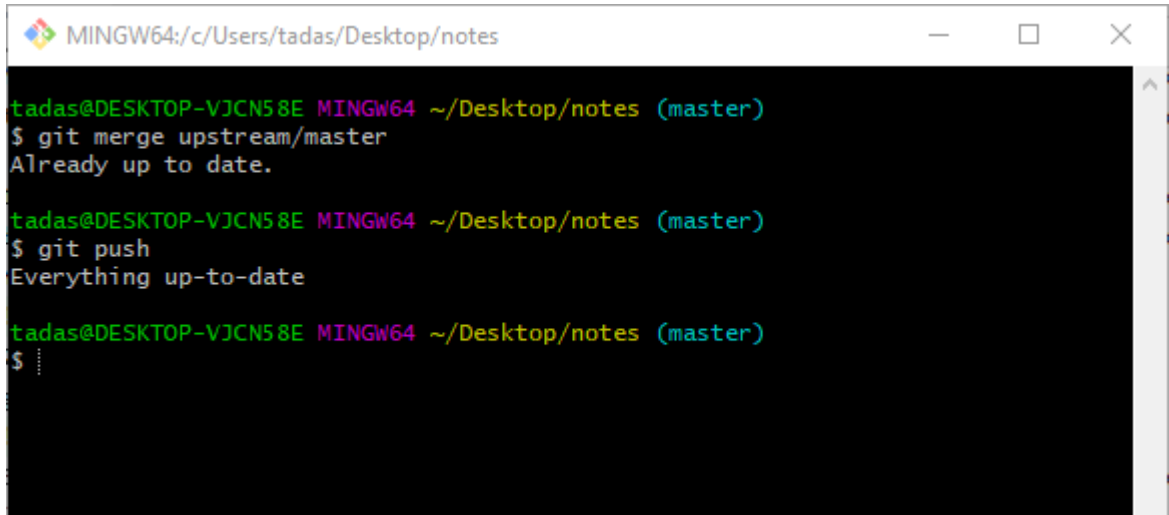
```
MINGW64:/c/Users/tadas/Desktop/notes

origin https://github.com/tvaidotas/notes.git (push)
upstream https://github.com/bob-crutchley/notes.git (fetch)
upstream https://github.com/bob-crutchley/notes.git (push)

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$ git fetch upstream
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Total 69 (delta 27), reused 27 (delta 27), pack-reused 42
Unpacking objects: 100% (69/69), done.
From https://github.com/bob-crutchley/notes
* [new branch]      dev      -> upstream/dev
* [new branch]      issue-43  -> upstream/issue-43
* [new branch]      issue-46  -> upstream/issue-46
* [new branch]      issue-48  -> upstream/issue-48
* [new branch]      issue-56  -> upstream/issue-56
* [new branch]      issue-7   -> upstream/issue-7
* [new branch]      issue-74  -> upstream/issue-74
* [new branch]      issue-82  -> upstream/issue-82
* [new branch]      issue-86  -> upstream/issue-86
* [new branch]      master   -> upstream/master

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$ .....
```

6. You will want to update the master branch, so we will merge the *upstream/master* into the *local origin/master*. You can do this by executing the `git merge upstream/master` command. Keep in mind that there may be merge conflicts that you will need to resolve. If you have resolved the merge conflicts, or if there were none, you should *push* to update your *origin/master* branch.

A screenshot of a Windows terminal window titled "MINGW64:/c/Users/tadas/Desktop/notes". The terminal shows a user named "tadas@DESKTOP-VJCN58E" in the "MINGW64" environment, currently on the "master" branch in the directory "~/Desktop/notes". The user enters the command `git merge upstream/master`, and the output is "Already up to date.". Then, the user enters `git push`, and the output is "Everything up-to-date". The prompt `$` is visible at the end of the line.

```
tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$ git merge upstream/master
Already up to date.

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$ git push
Everything up-to-date

tadas@DESKTOP-VJCN58E MINGW64 ~/Desktop/notes (master)
$
```