

How to use github for collaborative coding

<https://github.com>

This is the beginning of what will hopefully be a useful reference/how-to for using github. It's a giant mess at the moment.

1 Adding and committing

`git add` adds your modified files to the “queue” to be committed later.

```
cl> git add filename
cl> git add -A
```

`git commit` commits the files that have been added and creates a new revision with a log. `git add` and `git commit` update your local repository, but don't do anything to the remote.

The two actions can be combined with `git commit -a`. This will open up a file; write the commit message at the top, then close this file. Then do `git push`.

2 Clone

Cloning is essentially the same as making a copy. Any changes made in the original repository will not be reflected in your copy unless you do a `git pull` command.

To clone a repository:

1. Go to the web page for the repository you want
2. Copy the link (see figure 1) to your terminal using the following command (using the exposure time calculator repo as an example):

```
cl> git clone git@github.com:laurelfarris/ExposureTimeCalculator.git
cl> cd ExposureTimeCalculator
cl> touch test.txt
cl> git add test.txt # Add test.txt to the 'staging area'
cl> git commit -m "creating new file"
cl> git push -u origin master
```

Note: 'cl' stands for 'command line'.

```
cl> git remote add nameofremote https://github.com/holtzmanjon/a575.git
cl> git pull nameofremote
```

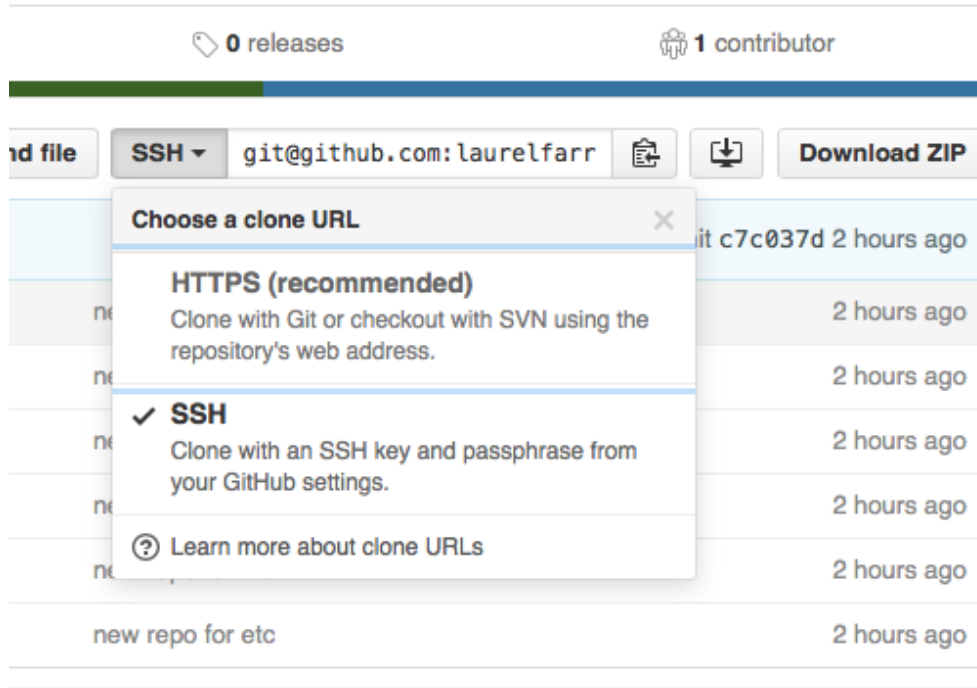


Figure 1: Screenshot of repository webpage to be cloned. You can choose either SSH or HTTPS. With SSH you can put in your ssh key and then you won't be asked for your username and password every time you push something.

3 Fork

Forking a repository will register the project under your own username.

4 Merge

5 Branches

```
cl> git branch # List the current branches. One ('master') by default.
          There will be an asterisk ('*') next to the one you are currently using.
cl> git branch junk # Create a new branch called 'junk'.
          Will get an error if you enter the name of a branch that already exists.
cl> git checkout junk # Change to 'junk' branch.
cl> git checkout master # Back to 'master' branch
cl> git branch -a # See what's going on with your branches...
```

6 Remotes

```
> git remote -v #Useful command that lists the remotes you currently have
                # along with their names (origin, upstream, etc.)
```

```

#Nothing here yet since I didn't clone or add a remote
> git remote add origin https://github.com/laurelfarris/a575hw.git
> git remote -v
    origin https://github.com/holtzmanjon/a575.git (fetch)
    origin https://github.com/holtzmanjon/a575.git (push)
> git remote add origin https://github.com/holtzmanjon/a575.git
    fatal: remote origin already exists.
    #This did not work because there is already a remote called origin.
    #However, I can still add this remote under a different name:
> git remote add upstream https://gitblubhub.com/holtzmannjon/aa575.git
> git remote -v
origin https://github.com/holtzmanjon/a575.git (fetch)
origin https://github.com/holtzmanjon/a575.git (push)
upstream https://gitblubhub.com/holtzmannjon/aa575.git (fetch)
upstream https://gitblubhub.com/holtzmannjon/aa575.git (push)
    # but... SO MANY TYPOS! OH NO!
> git remote set-url upstream https://github.com/holtzmanjon/a575.git
    # 'add' ADDS a remote
    # 'set-url' CHANGES an EXISTING remote;
> git remote -v
origin https://github.com/holtzmanjon/a575.git (fetch)
origin https://github.com/holtzmanjon/a575.git (push)
upstream https://github.com/holtzmanjon/a575.git (fetch)
upstream https://github.com/holtzmanjon/a575.git (push)

```

7 Useful links

This is a lovely website: <https://git-scm.com/docs/>, particularly the ‘Github Cheat Sheet’.

8 Misc

Creating, editing, and deleting files locally: how this affects your repository remotely.

```

> vi test.txt
    *edit edit edit*
> git status
    ... Untracked files: ... test.txt
> git add test.txt # Add test.txt to the staging area
# NOTE: If you delete a file locally, it still needs to be removed from
# your remote repository. '> git add' literally adds things; it doesn't take them
# away. " use 'git add/rm <file>...' to update what will be committed. "
> rm unwanted.txt
> git rm unwanted.txt
> git status
    ... Changes to be committed:  new file: test.txt
> git commit -m "testing" # Commit your changes!
> git status
    ... nothing to commit (working directory clean)
# From terminal messages:

```

```

    (use "git add <file>..." to update what will be committed)
    (use "git checkout -- <file>..." to discard changes in working directory) #??
    no changes added to commit (use "git add" and/or "git commit -a")
# What does "git commit -a" do???

```

9 Getting stuff from remote repositories

```

> git clone https://github.com/username/repositoryname.git
                                # clone (~copy) that repo in a newly created directory.
> git branch                    # No branches here
> cd repositoryname
> git branch
    * master

                                # clone creates a local branch called 'master'
                                #   (based on the remote's active branch)
                                #   Also created a remote called 'origin' of the url
                                #       that was cloned
> git pull origin master # Pulls everything from the origin remote under
                                #   the master branch, and puts it in your current
                                #   directory
# 'git pull' does a 'git fetch' and 'git merge' all at once.
    # e.g. > git fetch remoteName

# for funsies
> git checkout junk # Switched to junk branch... all the stuff I just
                    #   pulled is gone!

> git branch
    * junk
    master
> git checkout master # And it's back!
> git branch
    junk
    * master

```

10 Adding stuff to remote repositories

```

cl> git push -u origin master
    # the remote name ('origin' here) is optional. Origin is the default
    # -u is an option that saves your parameters... should be able to just do
    #   >git push ;;from this point on.

```

Change the remote that you're trying to push to (2 ways):

1. cl> vi .git/config
Change the url, e.g. https://username@github.com/username/repo.git
2. cl> git remote set-url origin https://username@github.com/username/repo.git

You'll be asked for a password when pushing; it will be the same as the one you use to log into github.