

Version Control with Git

Collaborating

☀ Learning Objectives

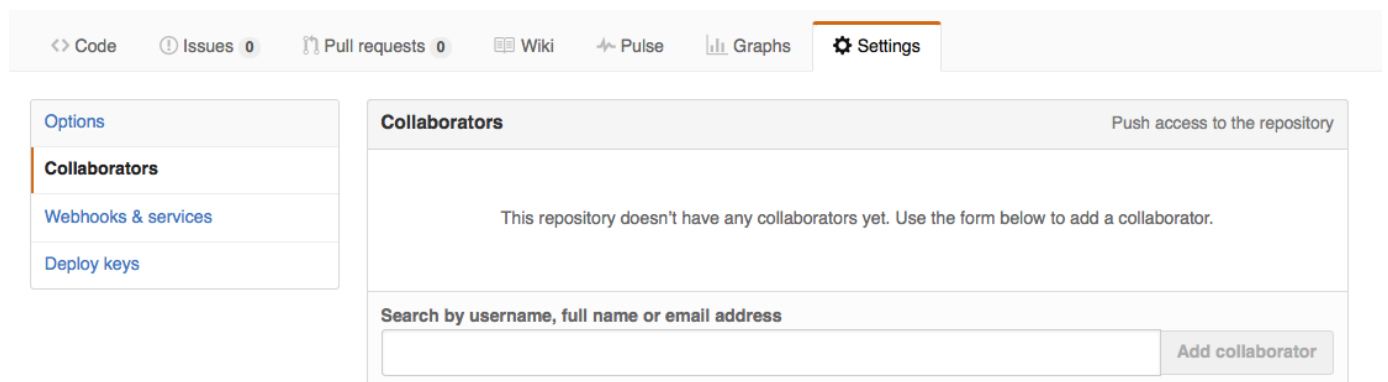
- Collaborate pushing to a common repository.

For the next step, get into pairs. Pick one of your repositories on GitHub to use for collaboration.

🚀 Practicing by yourself

If you're working through this lesson on your own, you can carry on by opening a second terminal window, and switching to another directory (e.g. `/tmp`). This window will represent your partner, working on another computer. You won't need to give anyone access on GitHub, because both 'partners' are you.

The partner whose repository is being used needs to give the other person access. On GitHub, click the settings button on the right, then select Collaborators, and enter your partner's username.



The screenshot shows the GitHub interface for a repository named 'planets'. The top navigation bar includes links for Code, Issues (0), Pull requests (0), Wiki, Pulse, Graphs, and Settings. The Settings page is open, and the 'Collaborators' tab is selected in the left sidebar. The main content area shows a message: 'This repository doesn't have any collaborators yet. Use the form below to add a collaborator.' Below this message is a search bar with the placeholder text 'Search by username, full name or email address' and an 'Add collaborator' button.

Figure: Adding collaborators on GitHub

The other partner should `cd` to another directory (so `ls` doesn't show a `planets` folder), and then make a copy of this repository on your own computer:

```
$ git clone https://github.com/vlad/planets.git
```

Replace 'vlad' with your partner's username (the one who owns the repository).

`git clone` creates a fresh local copy of a remote repository.

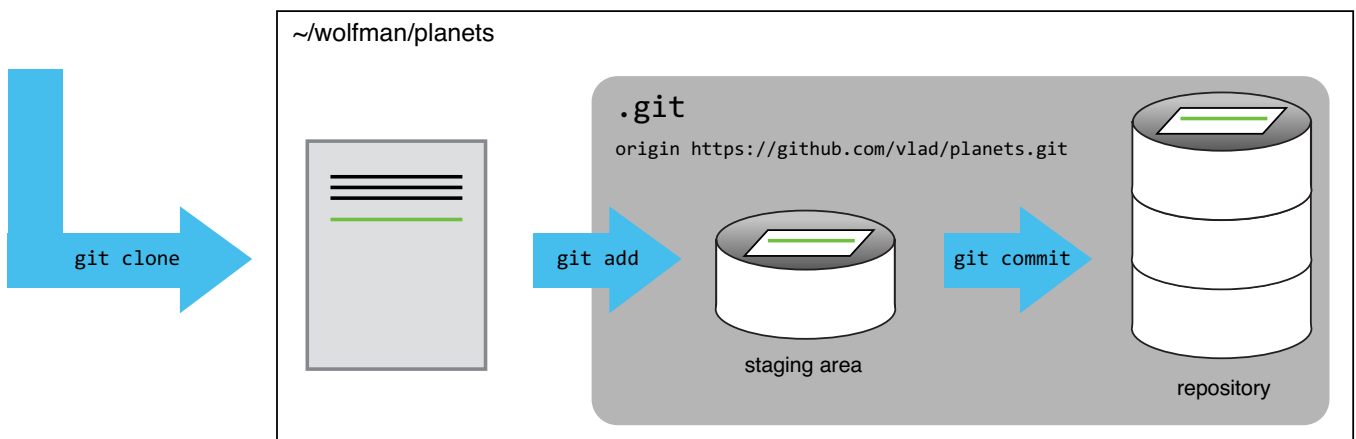
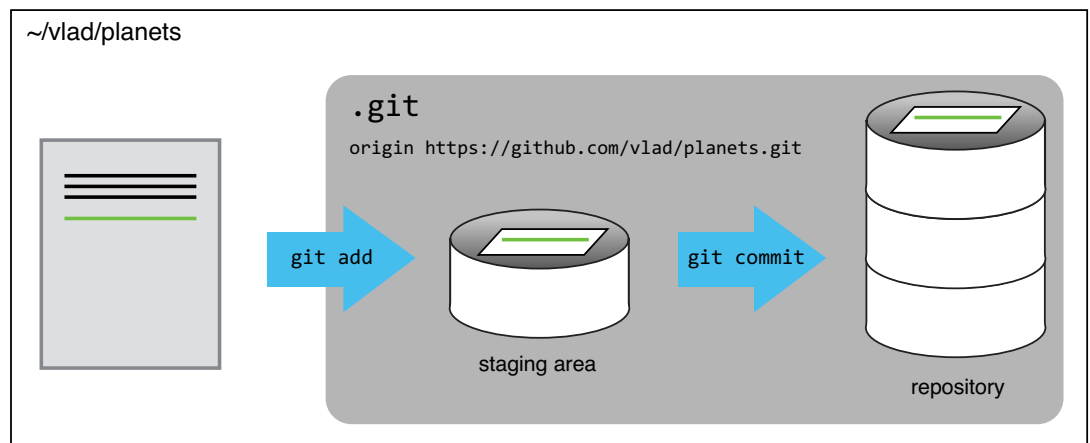


Figure: After Creating Clone of Repository

The new collaborator can now make a change in their copy of the repository:

```
$ cd planets  
$ nano pluto.txt  
$ cat pluto.txt
```

It is so a planet!

```
$ git add pluto.txt  
$ git commit -m "Some notes about Pluto"
```

```
1 file changed, 1 insertion(+)  
create mode 100644 pluto.txt
```

then push the change to GitHub:

```
$ git push origin master
```

```
Counting objects: 4, done.  
Delta compression using up to 4 threads.  
Compressing objects: 100% (2/2), done.  
Writing objects: 100% (3/3), 306 bytes, done.  
Total 3 (delta 0), reused 0 (delta 0)  
To https://github.com/vlad/planets.git  
9272da5..29aba7c  master -> master
```

Note that we didn't have to create a remote called `origin`: Git does this automatically, using that name, when we clone a repository. (This is why `origin` was a sensible choice earlier when we were setting up remotes by hand.)

We can now download changes into the original repository on our machine:

```
$ git pull origin master
```

```
remote: Counting objects: 4, done.  
remote: Compressing objects: 100% (2/2), done.  
remote: Total 3 (delta 0), reused 3 (delta 0)  
Unpacking objects: 100% (3/3), done.  
From https://github.com/vlad/planets  
  * branch            master      -> FETCH_HEAD  
Updating 9272da5..29aba7c  
Fast-forward  
 pluto.txt | 1 +  
 1 file changed, 1 insertion(+)  
 create mode 100644 pluto.txt
```

[Software Carpentry](#)[Source](#)[Contact](#)[License](#)