Git & GitHub

An intro to development tools you will use every day

Collaborate

Write code (or any kind of document) with other people.

Track and revert changes

Mistakes happen. Wouldn't it be nice if you could see the changes that have been made and go back in time to fix something that went wrong?

You already manage versions of your work!

Do you have files somewhere that look like this?

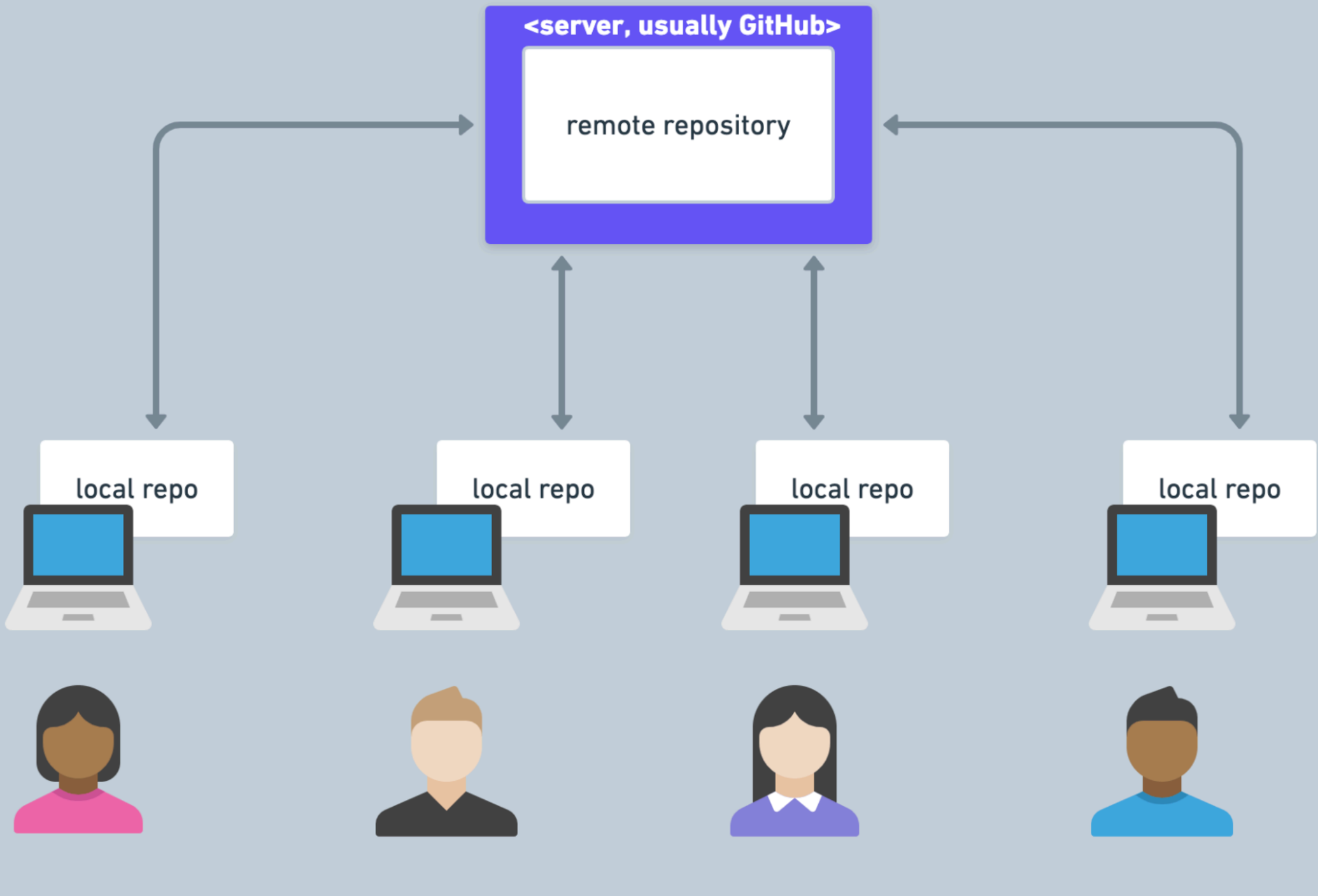
ResumeSeptember2019.docx
ResumeforDukeJob.docx
ResumeOLD.docx
ResumeNEW.docx
ResumeFINAL.docx
ResumeREALLYFINAL.docx

You invented your own version control!

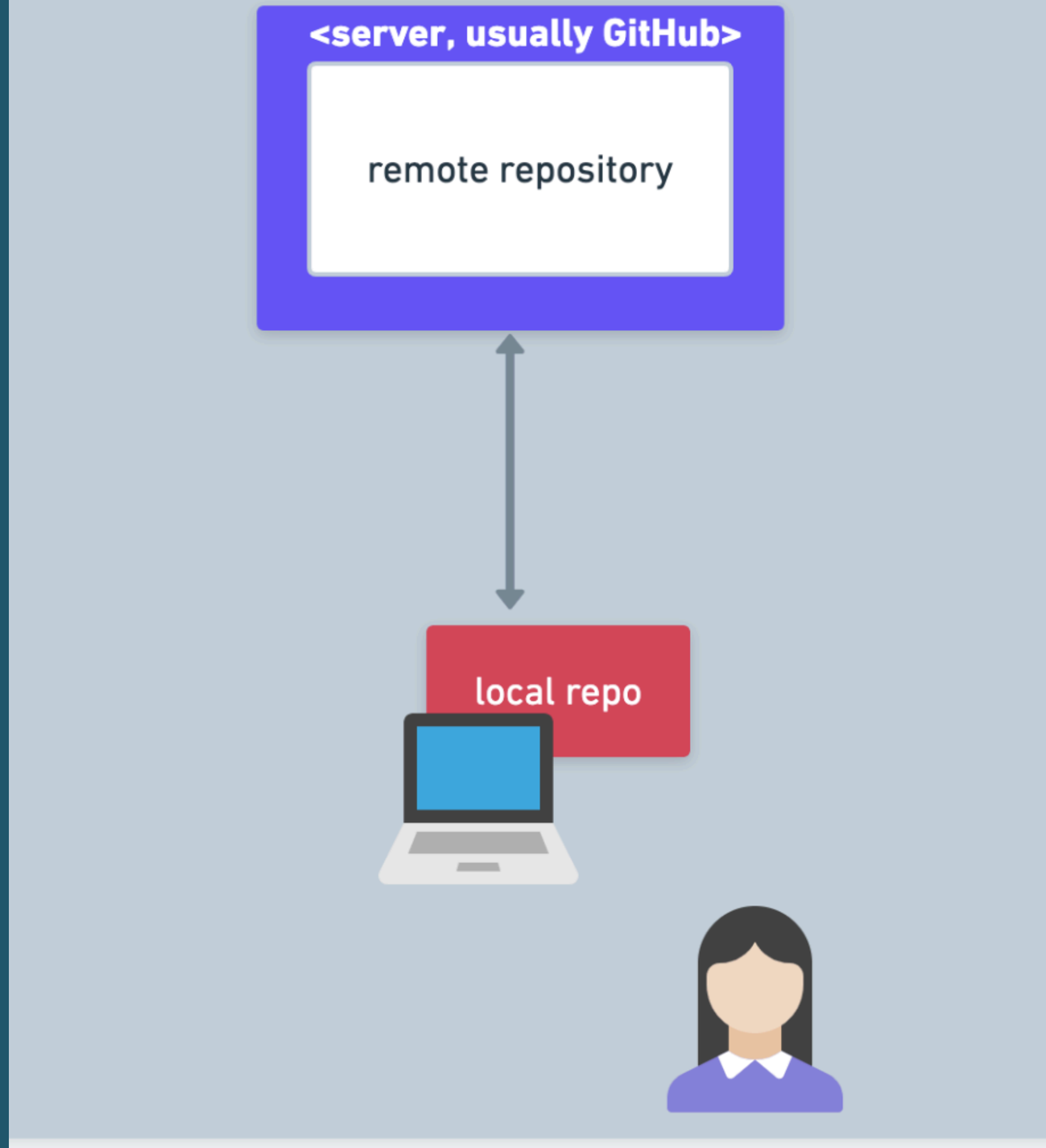
What do we use Git for?

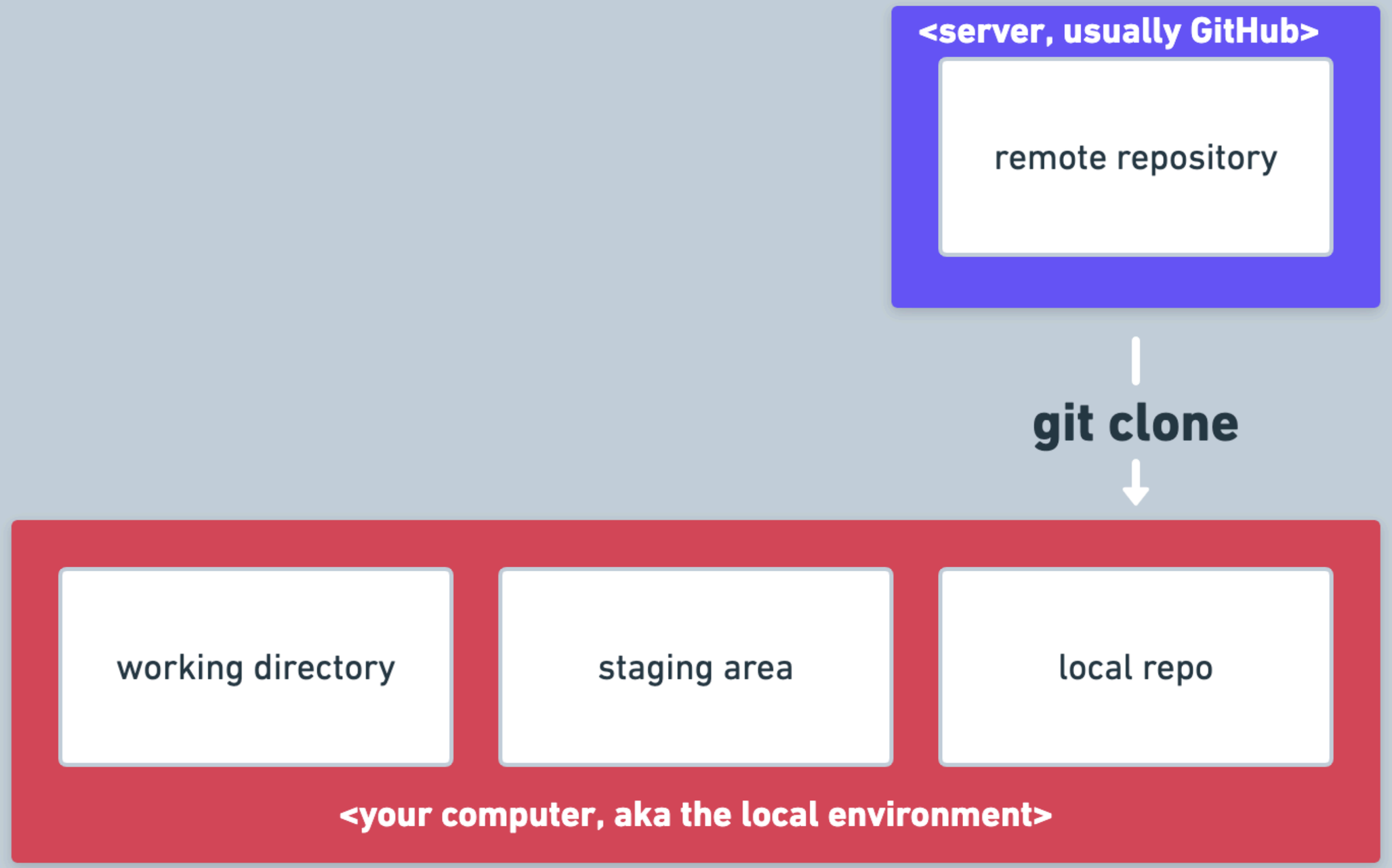
We want to track changes in our files as we work.

We want to put our files and its history of changes on GitHub so they can be shared.


<server, usually GitHub>
remote repository
local repo
local repo
local repo
local repo

working solo for now




<server, usually GitHub>
remote repository
git clone
working directory
staging area
local repo
<your computer, aka the local environment>

Setup

— `git --version`

— `brew install git`

— Set name and email in gitconfig

```
$ git config --global user.name "Captain Barnacles"
$ git config --global user.email "cb@octonauts.org"
```

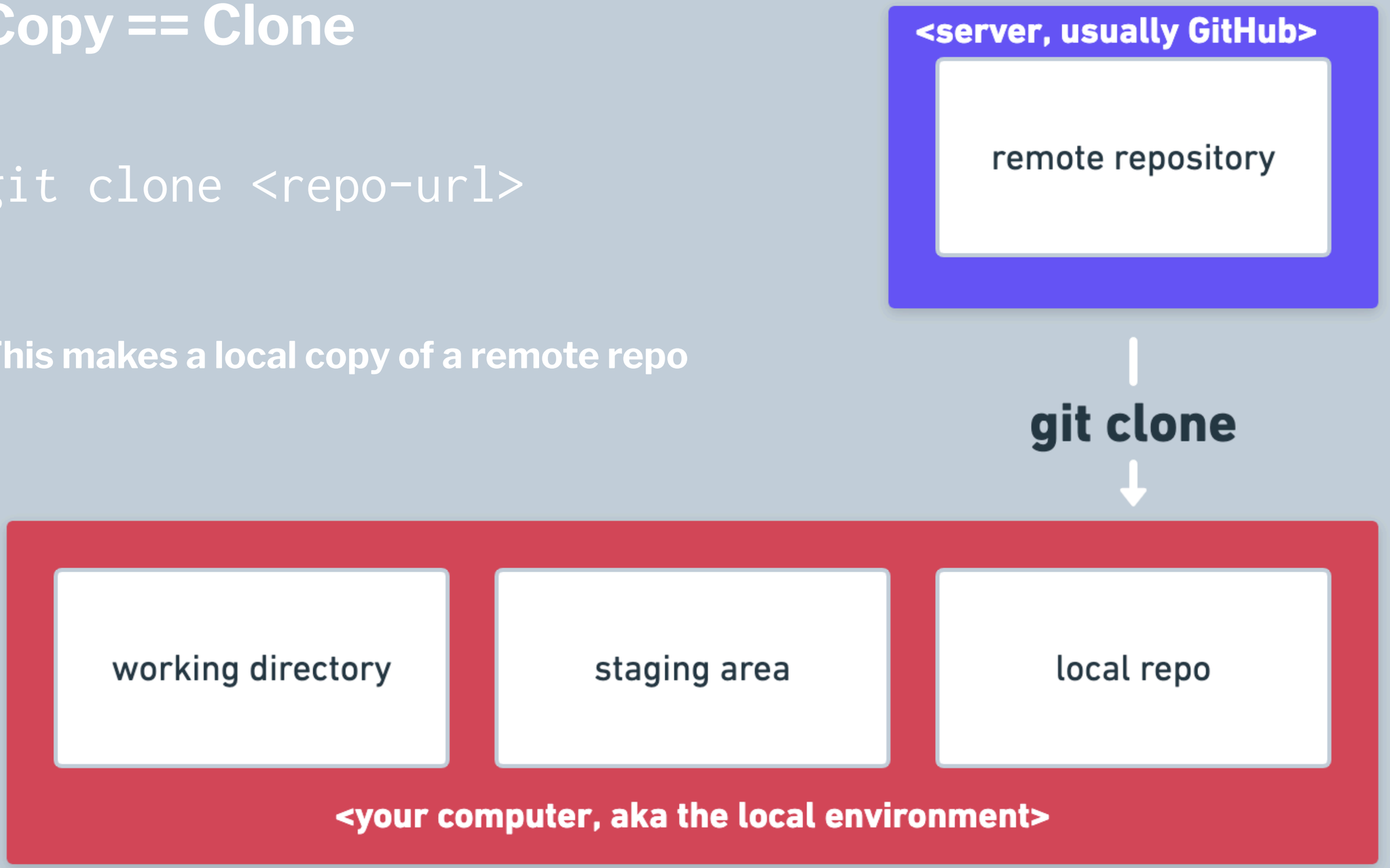
Vocab

— **Repository**: A collection of files and their changes
— **History**: An ordered list of shapshots of changes in a repository over time
— **Working Directory**: The working copy of your files in your editor. These are also your *unstaged* changes.
— **Staging area**: A Git-specific concept: the way git designates a set of changes to be committed.
— **Commit**: One set or snapshot of changes

Copy == Clone

```
git clone <repo-url>
```

This makes a local copy of a remote repo



Branches

A branch is just another copy of your repo that allows you to isolate changes and leave the original copy untouched.

You can later choose to combine these changes in whole or part with the "master" copy, or not.

Branching commands

```
git branch
git checkout
git switch
```


<server, usually GitHub>
remote repository
git pull
git push
working directory
git add
staging area
git commit
local repo
<your computer, aka the local environment>

Git commands

```
git add
git commit
git push
```

Overview of that process

— In a git repository, changes made in our editor (aka our working directory or working tree) need to be manually added to enter into the history

— The first time we add a new file, we tell Git to add the file to the repository to be tracked

— This is also called staging a file. A snapshot of our changes is now in the staging area (aka the index, aka the cache), ready to be saved.

— A commit saves the changes made to a file, not the file as a whole. The commit will have a unique ID so we can track which changes were committed when and by whom.

See what's happening with these commands

```
git log
git diff
git status
```

Homework workflow

1. Accept the assignment invitation from GitHub Classroom. You'll find this in your team's Google Classroom.
2. Clone the homework repo to your local environment
3. Create a new branch and name it something descriptive of the work you're doing, like `pet-finder-assignment`
4. Work on your homework!
5. Periodically add and commit your work to your local repo.
6. When you are done, push your work to the remote repo
7. Open a pull request from your branch to `master`.

Your local workflow

1. add
2. commit
3. push

Git commands

```
git add .
git status
git commit -m "Create content section"
git status
git push origin master
```

Pull Requests

A GitHub feature that lets you compare changes between branches and comment on them.

The request is to merge the changes from one branch into the other (base branch).

GitHub gives you a friendly green button to do this -- but **DON'T DO IT!**

A pull request that you open should be merged by someone else after it has been reviewed and discussed.

a handy cheat sheet