Git + GitHub

Setup

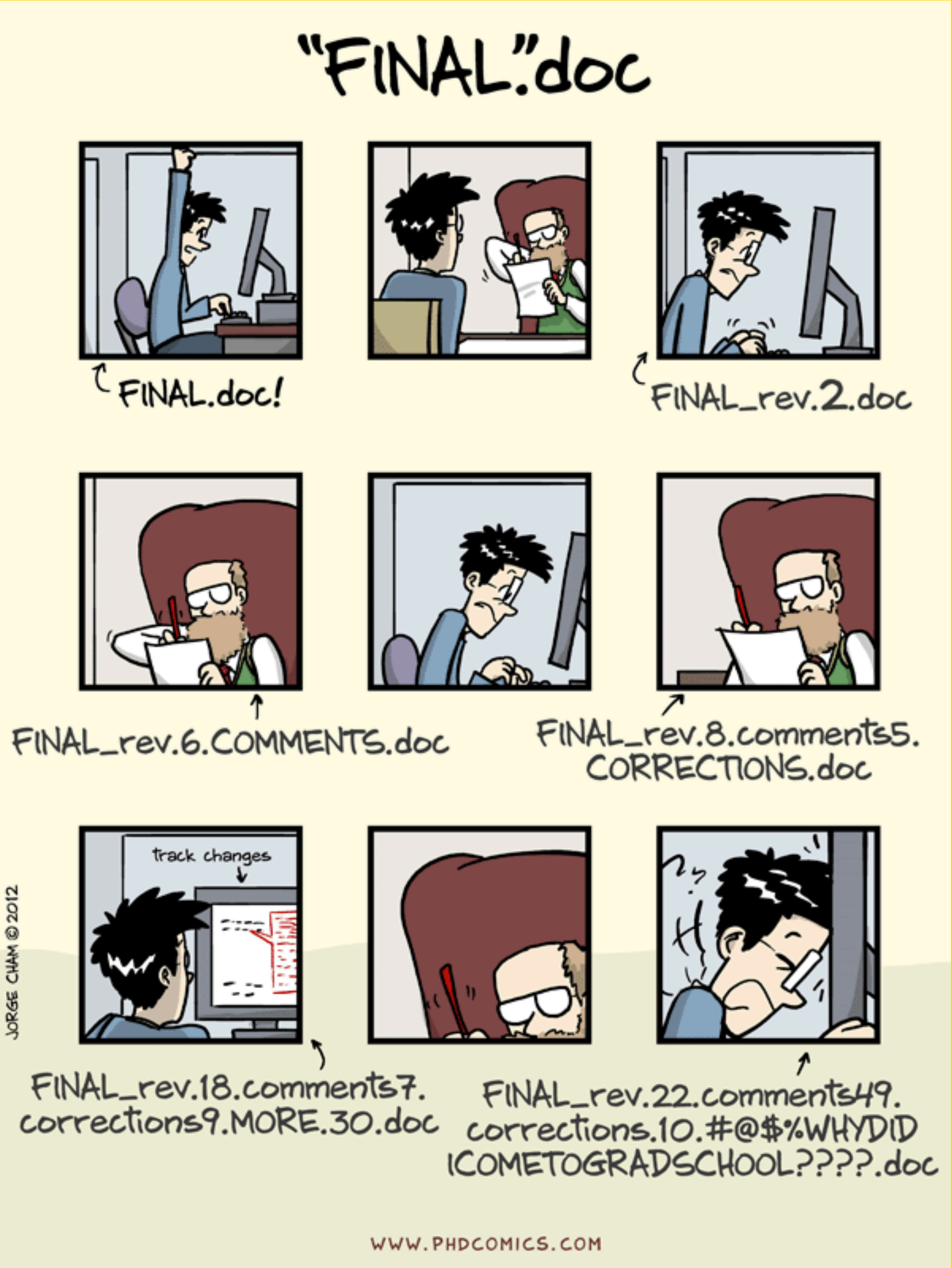
- Make account using GitHub
  - https://github.com/

- **Windows**
  - Git should be installed on your computer as part of your Bash install (described above).
- **macOS**
- **For OS X 10.9 and higher**, install Git for Mac by downloading and running the most recent "mavericks" installer from this list. After installing Git, there will not be anything in your /Applications folder, as Git is a command line program. **For older versions of OS X (10.5-10.8)** use the most recent available installer labelled "snow-leopard" available here.

https://tinyurl.com/gitinstallmac2

Version Control

- Version control systems start with a base version of the document and then record changes you make each step of the way.
- You can think of it as a recording of your progress: you can rewind to start at the base document and play back each change you made, eventually arriving at your more recent version.



Version Control

- Make backups
- Keep history
- View changes
- Experiment
- Collaborate

Setting up Git

```
$ git config --global user.name "Vlad Dracula”
$ git config --global user.email "vlad@tran.sylvan.ia"
```

Creating a Repo

```
$ cd ~/Desktop
$ mkdir planets
$ cd planets
$ git init

$ ls -a

$ git checkout -b main
$ git status
```

- **Repository (repo) -**
  A storage area where a version control system stores the full history of commits (changes) of a project and information about who changed what, when.

- **Main (branch) -**
  Branches allow you to develop features, fix bugs, or safely experiment with new ideas in a contained area of your repository.

Tracking changes

- vim mars.txt
  - “Cold and dry, but everything is my favorite color “

```
$ git status
$ git add mars.txt
```

tell Git to track a file using git **add**

```
$ git status
$ git commit -m "Start notes on Mars as a base"

$ git status

$ git log
```

Git now knows that it’s supposed to keep track of mars.txt, but it hasn’t recorded these changes as a **commit** yet.

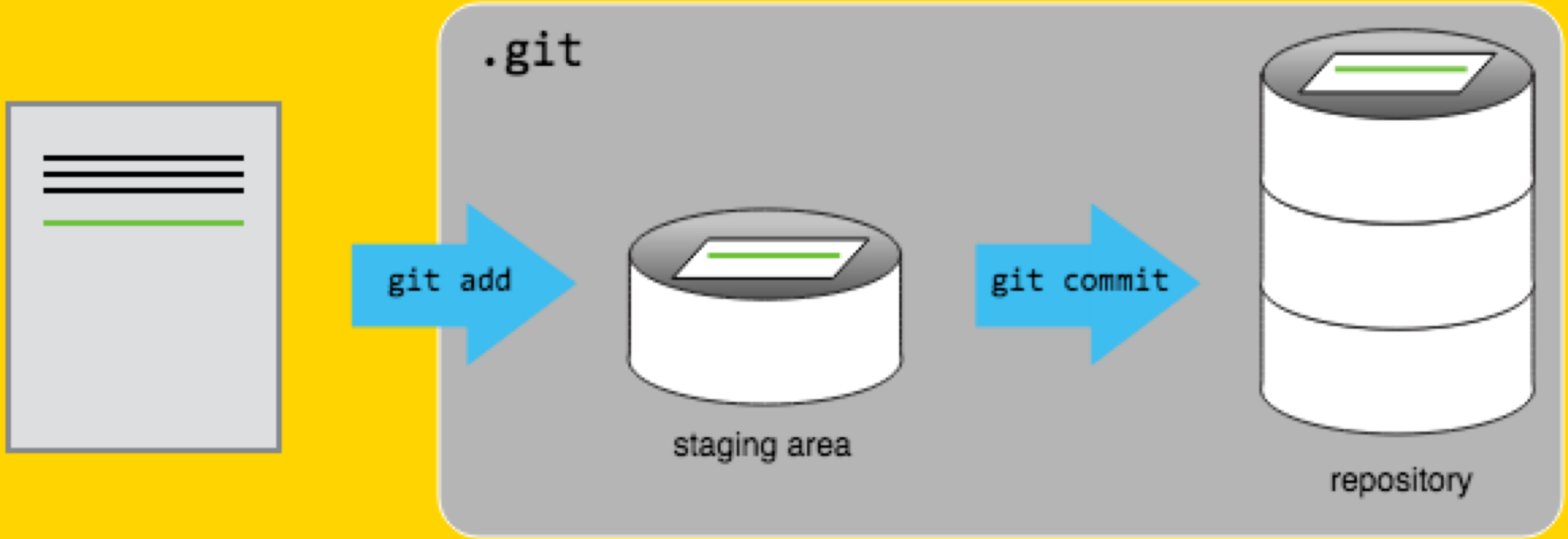
Tracking changes

- vim mars.txt
  - “The two moons are nice”
-
  ```
  $ git status
  $ git diff
  $ git add mars.txt
  $ git commit -m ”Adding notes on  Mars’ moons"

  $ git status

  $ git log
  ```

Tracking changes



Practice

- Add some text to mars.txt noting your decision to consider Venus as a base
- Create a new file venus.txt with your initial thoughts about Venus as a base for you and your friends
- Add changes from both files to the staging area, and commit those changes.

Practice

- Add some text to mars.txt noting your decision to consider Venus as a base
- $ vim mars.txt

- Create a new file venus.txt with your initial thoughts about Venus as a base for you and your friends
- $ vim venus.txt

- Add changes from both files to the staging area, and commit those changes.
- $ git **add** mars.txt venus.txt
- $ git **commit -m** "Write plans to start a base on Venus"

Directories

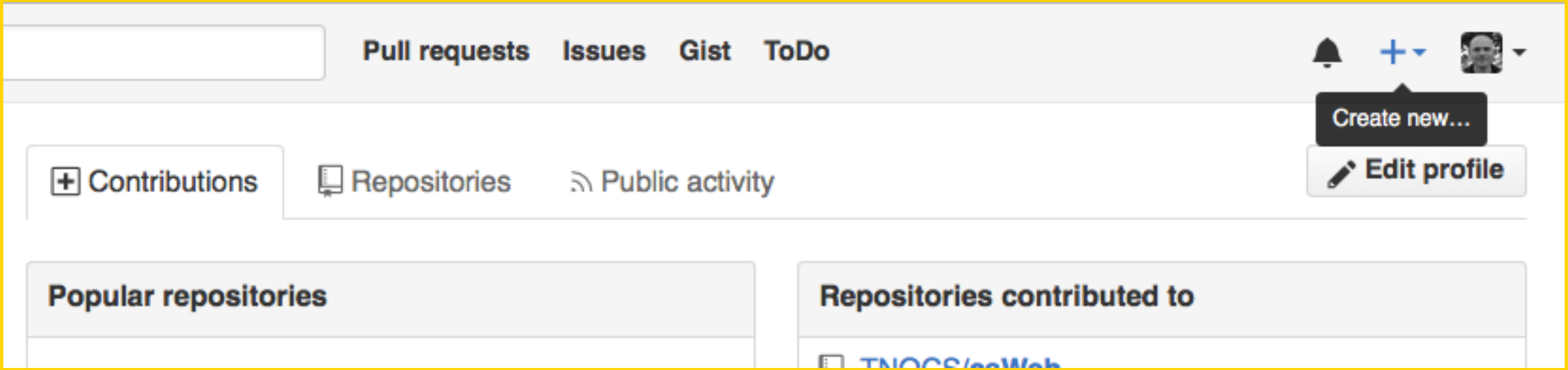
```
$ mkdir spaceships
$ git status
$ git add spaceships
$ git status
```

Git does not track directories on their own, only files within them.

```
$ touch spaceships/apollo-11 spaceships/sputnik-1
$ git status
$ git add spaceships
$ git status
$ git commit -m "Add some initial thoughts on spaceships"
```

GitHub

- Make a new Repo called planets

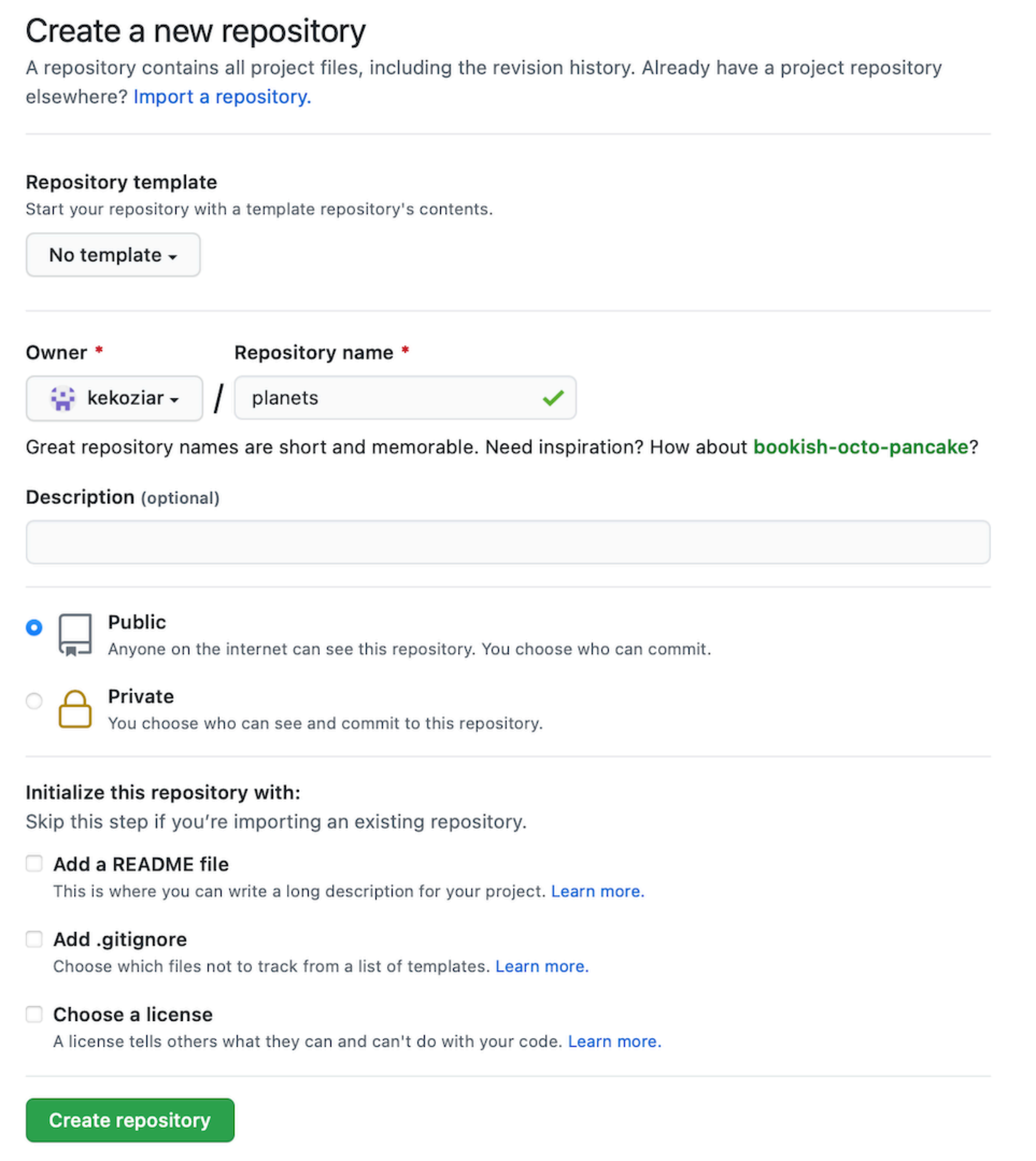


**Note: Since this repository will be connected to a local repository, it needs to be empty. Leave "Initialize this repository with a README" unchecked, and keep "None" as options for both "Add .gitignore" and "Add a license."

GitHub

- This effectively does the following on GitHub's servers:
- $ mkdir planets
- $ cd planets
- $ git init

**Note that our local repository still contains our earlier work on mars.txt, but the remote repository on GitHub appears empty as it doesn't contain any files yet.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Repository template

Start your repository with a template repository's contents.

No template ▾

Owner * **Repository name** *

kekoziar ▾ / planets ✓

Great repository names are short and memorable. Need inspiration? How about **bookish-octo-pancake**?

Description (optional)

● **Public**
Anyone on the internet can see this repository. You choose who can commit.

○ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

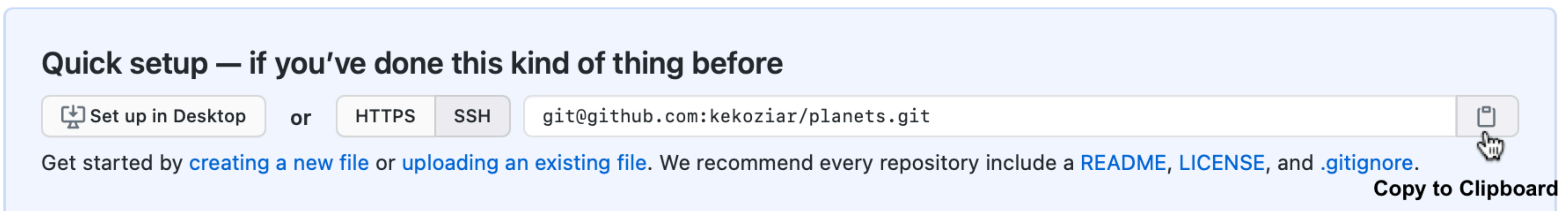
□ **Add a README file**
This is where you can write a long description for your project. Learn more.

□ **Add .gitignore**
Choose which files not to track from a list of templates. Learn more.

□ **Choose a license**
A license tells others what they can and can't do with your code. Learn more.

Connecting two Repos

- Making the GitHub repository a remote for the local repository.



Quick setup — if you've done this kind of thing before

```
git@github.com:kekoziar/planets.git
```

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

We use SSH here because, while it requires some additional configuration, it is a security protocol widely used by many applications. The steps below describe SSH at a minimum level for GitHub. A supplemental episode to this lesson discusses advanced setup and concepts of SSH and key pairs, and other material supplemental to git related SSH.

Connecting Repos

- Copy that URL from the browser, go into the local planets repository, and run this command:
- $ **git remote add origin** git@github.com:***vlad***/planets.git
- **origin** is a local name used to refer to the remote repository
- $ git remote **-v**

Push local changes | Pull remote changes

- This command will push the changes from our local repository to the repository on GitHub:
- Copies changes from a local repository to a remote repository.
  - $ git **push** origin main

- We can pull changes from the remote repository to the local:
- Copies changes from a remote repository to a local repository
  - $ git **pull** origin main
-