Git + GitHub

Setup

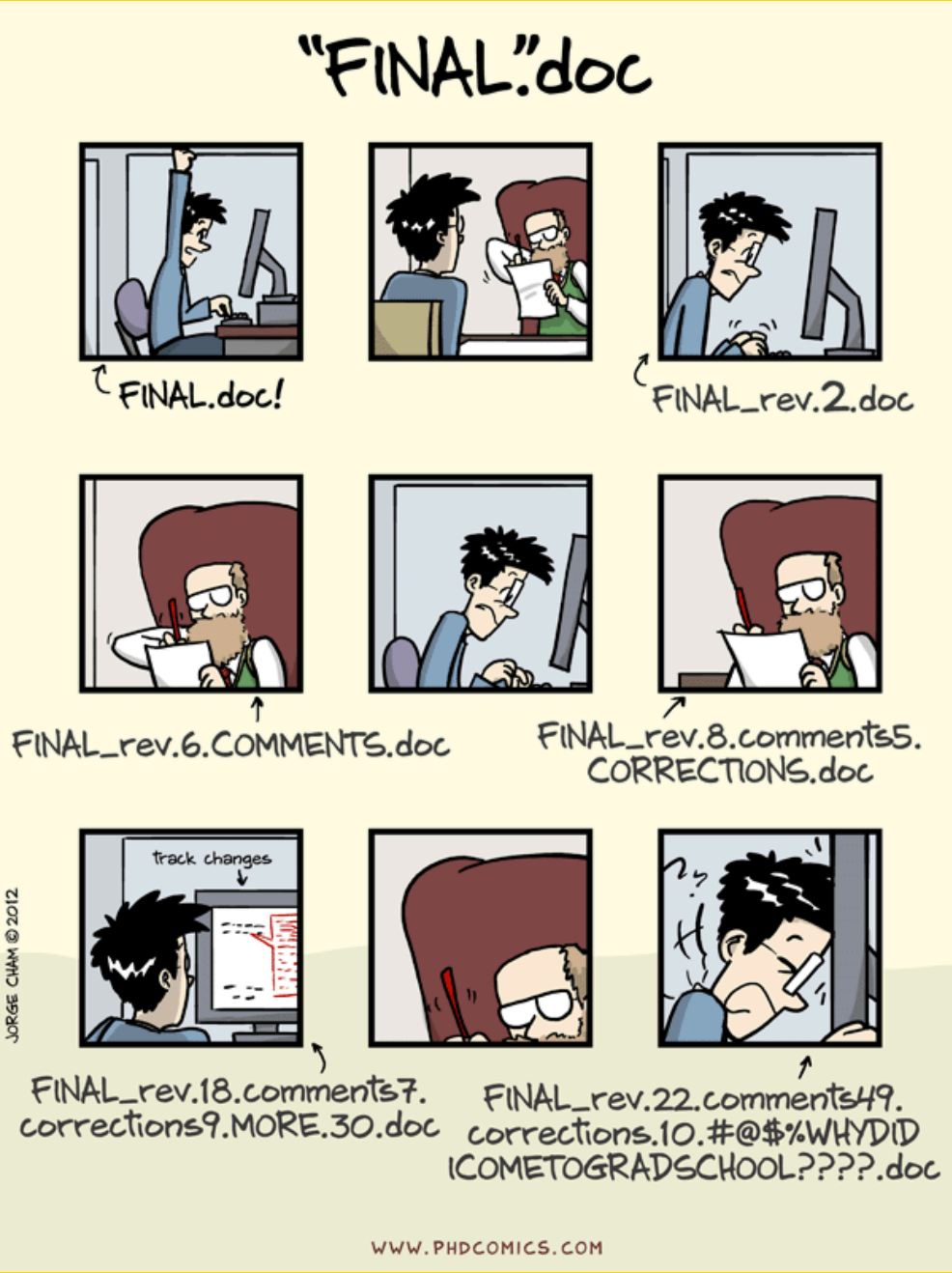
- Make account using GitHub
  - https://github.com/

- **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.

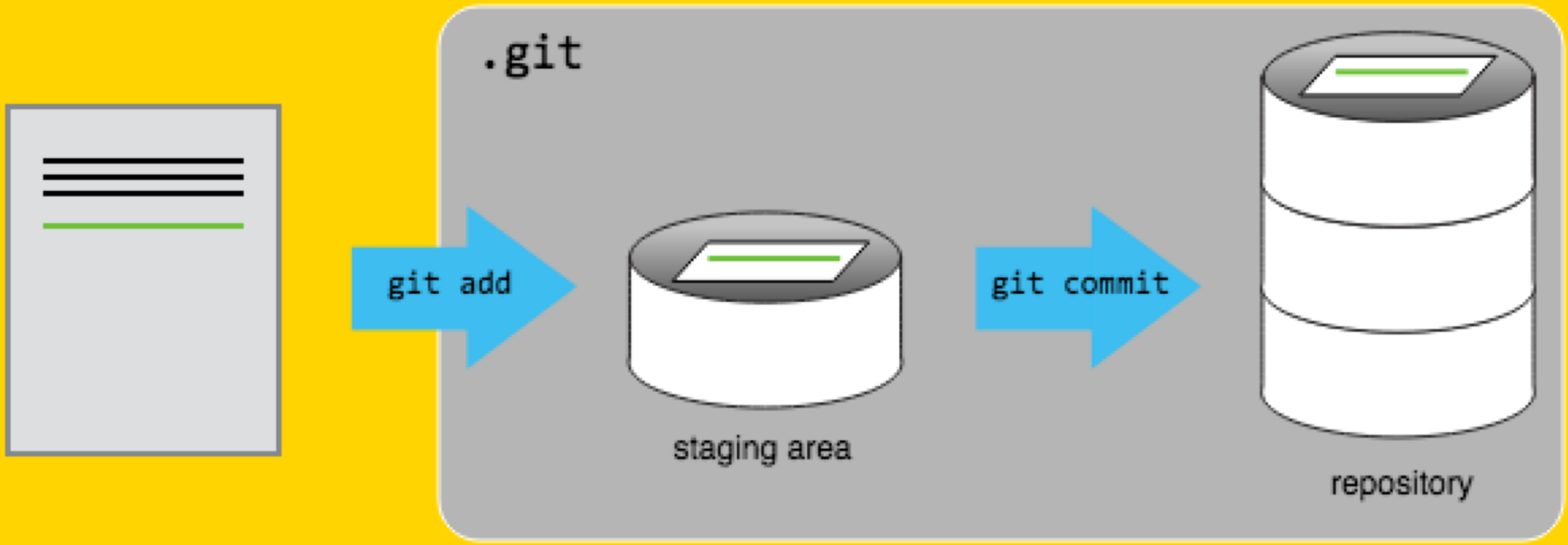
Tracking changes

- 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

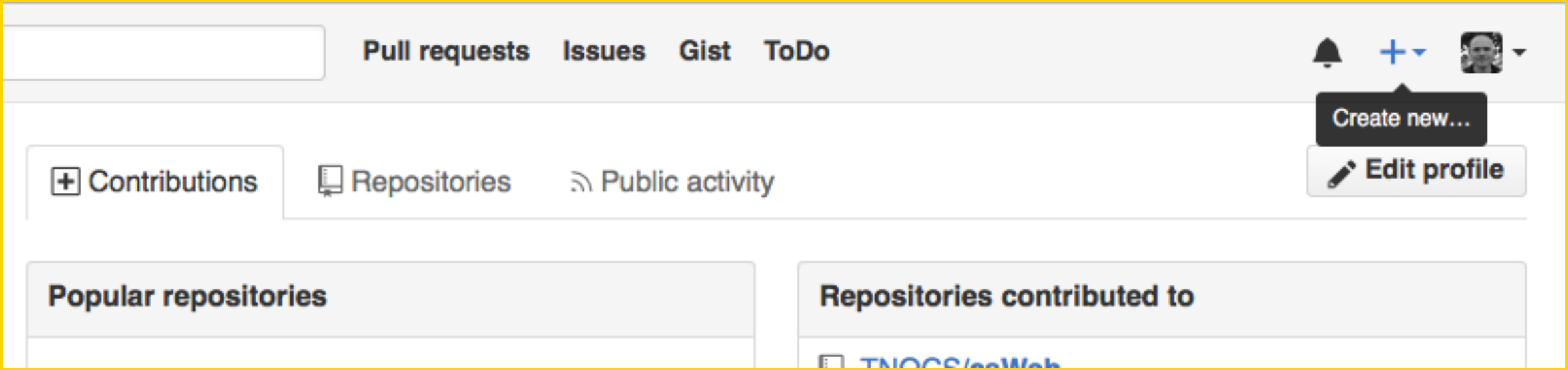
```
$ mkdir spaceships
$ git status
$ git add spaceships
$ git status
```

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

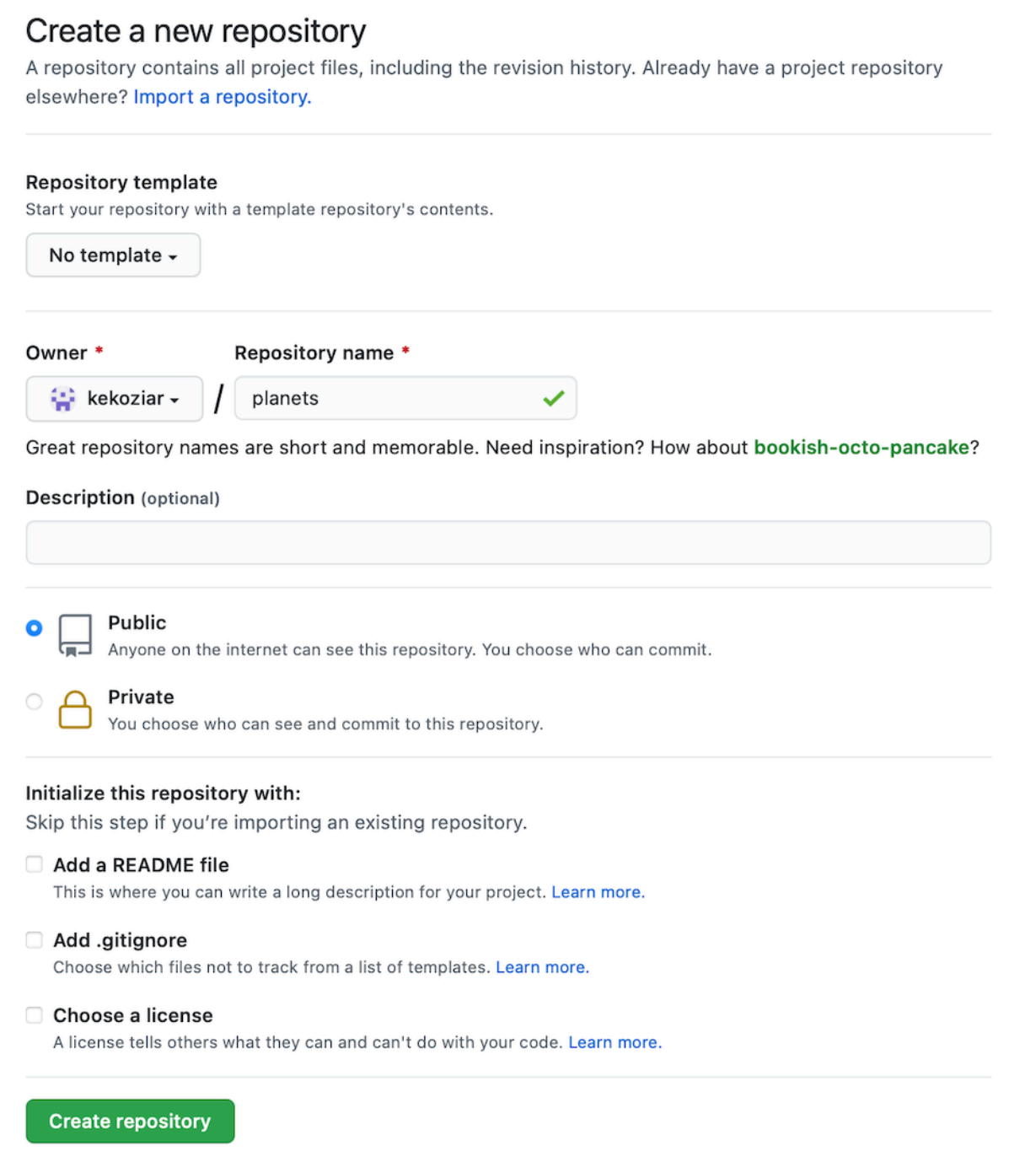


**Note: Since this repository will be connected to a local repository, it needs to be empty. Leave “Initialize this repository with a README” unchecked, and keep “None” as options for both “Add .gitignore” and “Add a license.”

GitHub

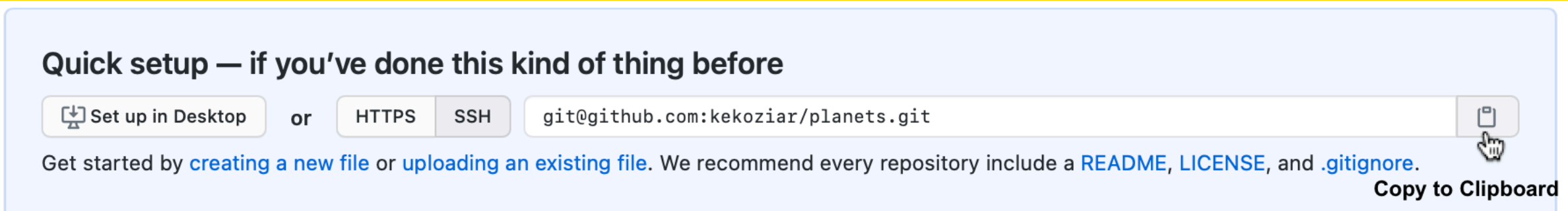
- This effectively does the following on GitHub's servers:
- $ mkdir planets
- $ cd planets
- $ git init

**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.



Connecting two Repos

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



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**

Connecting to GitHub

- Your computer needs authenticate remote repository (GitHub)
- Authenticate access using the command line.
- This method is called **Secure Shell Protocol** (**SSH**). SSH is a cryptographic network protocol that allows secure communication between computers using an otherwise insecure network.
- **SSH** uses what is called a key pair. This is two keys that work together to validate access. One key is publicly known and called the public key, and the other key called the private key is kept private

Connecting to GitHub

$ ssh-keygen **-t** ed25519 **-C** ”your@email.com"

Generating public/private ed25519 key pair. Enter file in which to save the key (/c/Users/Vlad Dracula/.ssh/id_ed25519):

> Enter

Enter passphrase (empty for no passphrase):

Your identification has been saved in /c/Users/Vlad Dracula/.ssh/id_ed25519 Your public key has been saved…
The key's randomart image is:

Connecting to GitHub

ls **-al** ~/.ssh

Output

rwxr-xr-x 1 Vlad Dracula 197121 0 Jul 16 14:48 ./

drwxr-xr-x 1 Vlad Dracula 197121 0 Jul 16 14:48 ../

-rw-r--r-- 1 Vlad Dracula 197121 419 Jul 16 14:48 id_ed25519

-rw-r--r-- 1 Vlad Dracula 197121 106 Jul 16 14:48 id_ed25519.pub

cat ~/.ssh/id_ed25519.pub

Output

ssh-ed25519

AAAAC3NzaC1lZDI1NTE5AAAAIDmRA3d51X0uu9wXek559gfn6UFNF69

yZjChyBIU2qKI vlad@tran.sylvan.ia

Connecting to GitHub

- Go to Github.com > Settings
- On settings page > “SSH and GPG keys”
- Click the “New SSH key”
- Paste your SSH key into the field, and click the “Add SSH key” to complete the setup.

```
$ ssh -T git@github.com
```

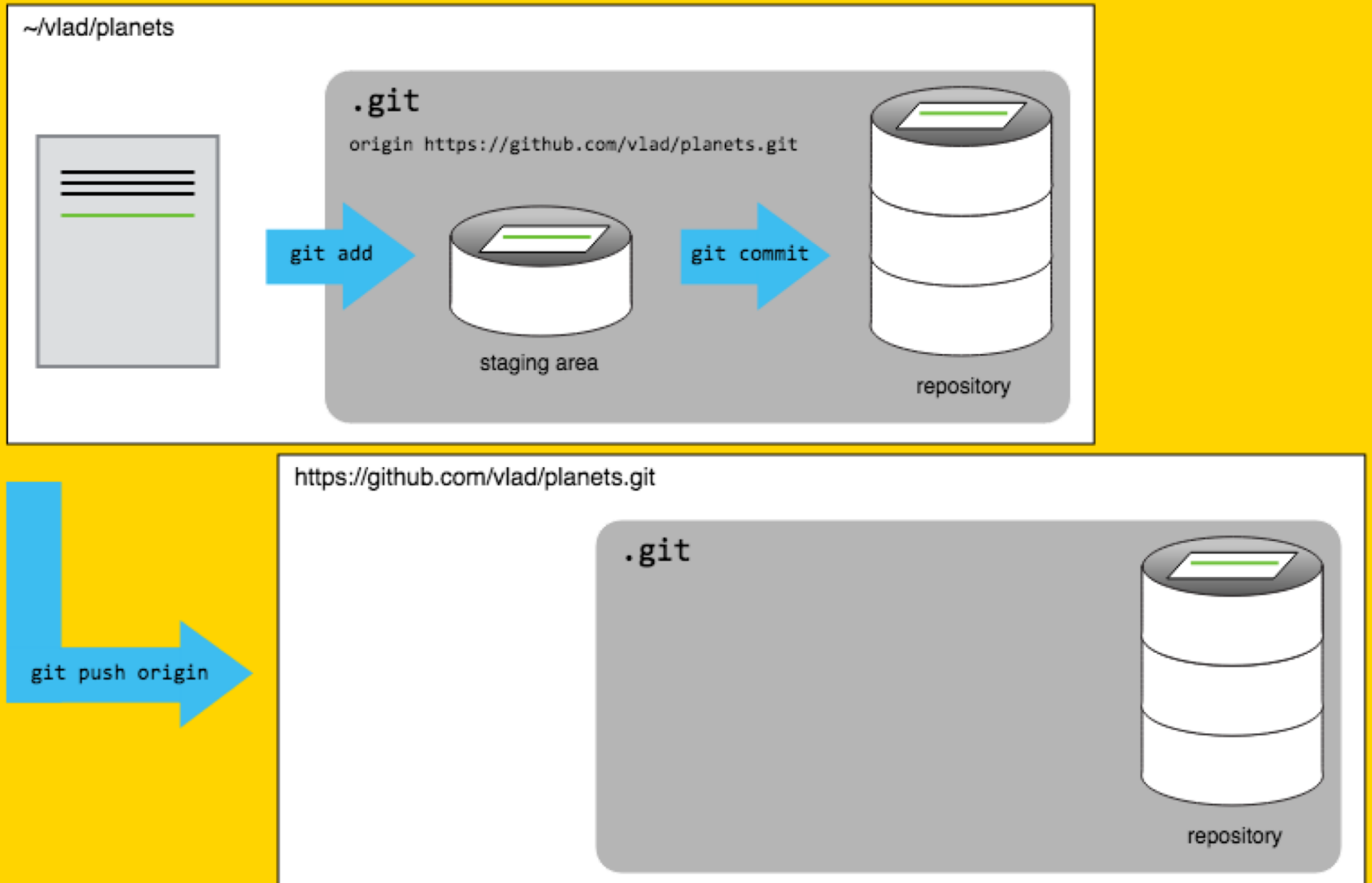
Hi Your Name! You've successfully
authenticated, but GitHub does not provide
shell access.

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
-

Connecting to GitHub



Cloning

- Create a local copy on your computer and sync between the two locations
- Need permission of the owner to push changes

Forking

- Copy of the Repo that you can work on
- Contribute to code where you are not the owner or collaborator of the Repo

Fork vs Clone

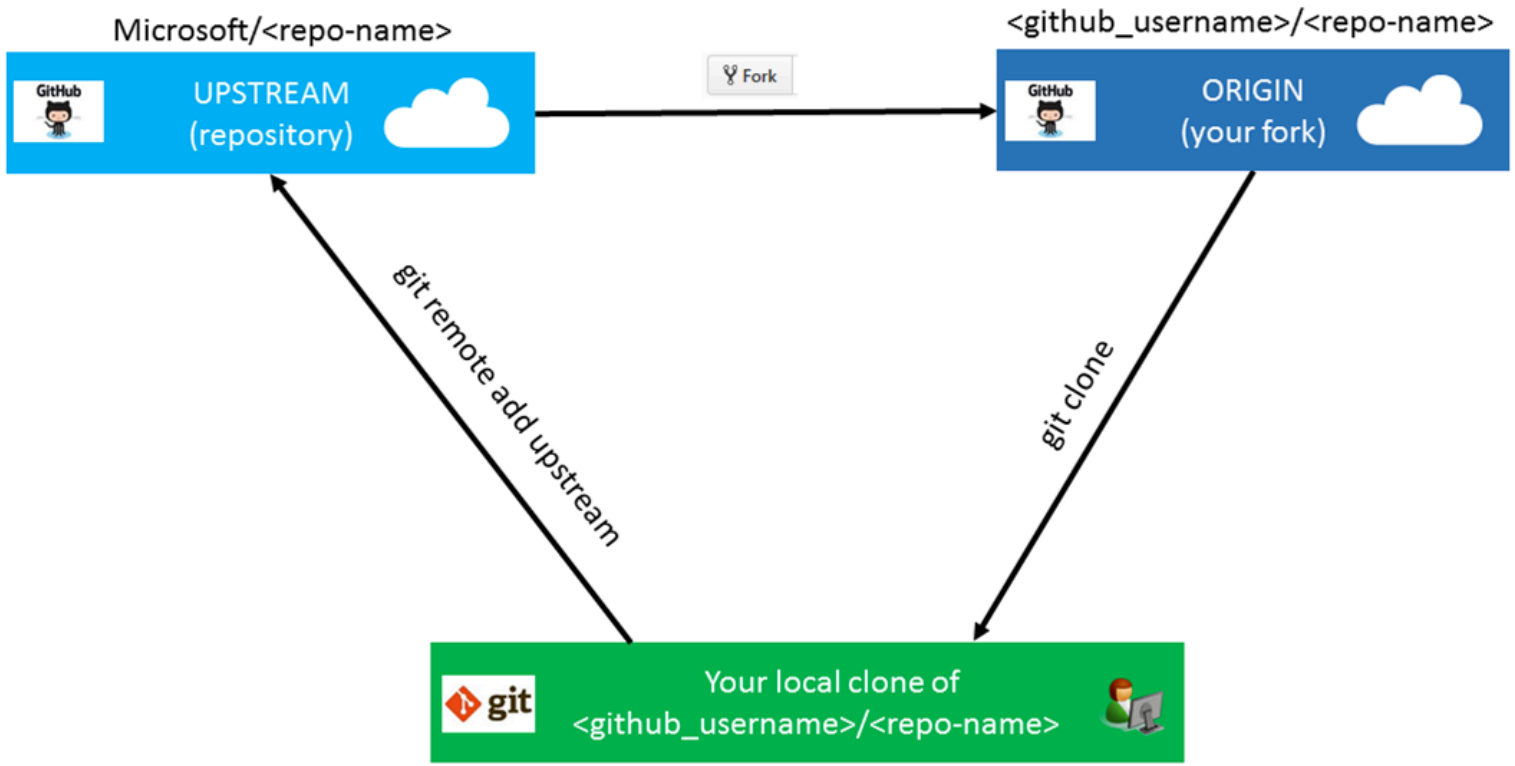
Git Fork vs Clone

Comparison Chart

Fork	Clone
A fork of a repository is nothing but a copy of that repository that you can work on.	A clone is basically a local copy of a remote repository that is stored on your computer.
It allows you to contribute code to the repositories where you aren't the owner or a collaborator.	It allows you to work on the projects, fix some issues or contribute changes to the code.
You do not need the owner's permission to for their repository.	You can push the changes back to the remote repo only if you have the push rights to the repo.

DifferenceBetween.net

Repository first time set up



Pull requests (PRs)

- Submit your contributions
- Accept or reject contributions of others
- Merge changes to base code

GitHub Desktop

Current Repository
vonage-php-sdk-core

Current Branch
hacktoberfest-create-pull...

Fetch origin
Last fetched 2 minutes ago

Changes 1 | History

1 changed file

README.md

Update README.md

Description

Commit to hacktoberfest-create...

README.md

```
@@ -76,7 +76,7 @@ To use [Vonage's SMS API][doc_sms] to send an SMS message, call the
`$client->sm
 required parameters, and a fluent interface provides access to optional parameters.

 ```php
-$text = new \Vonage\SMS\Message\SMS(NEXMO_TO, NEXMO_FROM, 'Test message using PHP cli
ent library');
+$text = new \Vonage\SMS\Message\SMS(VONAGE_TO, VONAGE_FROM, 'Test message using PHP c
lient library');
 $text->setClientRef('test-message');
 ```

@@ -234,8 +234,8 @@ All `$client->voice()` methods require the client to be constructe
d with a `Vona
 ```php
 $basic  = new \Vonage\Client\Credentials\Basic('key', 'secret');
 $keypair = new \Vonage\Client\Credentials\Keypair(
-    file_get_contents((NEXMO_APPLICATION_PRIVATE_KEY_PATH),
-    NEXMO_APPLICATION_ID
+    file_get_contents((VONAGE_APPLICATION_PRIVATE_KEY_PATH),
+    VONAGE_APPLICATION_ID
 );

 $client = new \Vonage\Client(new \Vonage\Client\Credentials\Container($basic, $keypai
r));
@@ -458,7 +458,7 @@ $client->numbers()->purchase('14155550100', 'US');
 To update a number, use `numbers()->update` and pass in the configuration options you
want to change. To clear a setting, pass in an empty value.
```
```