How to Turn your Github into a Lithub

Optionally: How to git gud

What is Github?

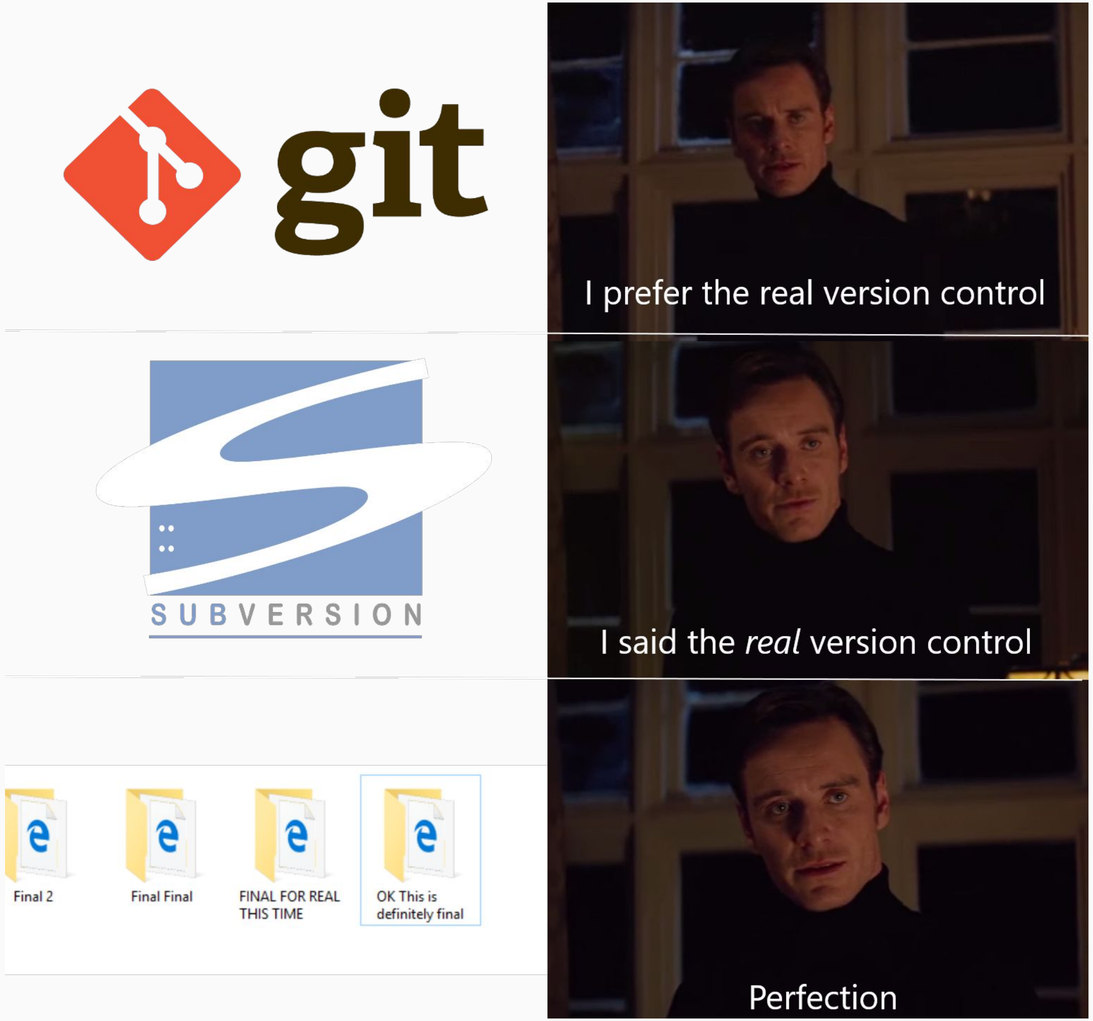
It’s a method to save your code and upload it to the cloud

Version Control

- It remembers your past edits and keeps your project’s history.

Collaborative

- Github makes it very easy to share your programs


git
SUBVERSION
Final 2
Final Final
FINAL FOR REAL THIS TIME
OK This is definitely final
I prefer the real version control
I said the *real* version control
Perfection

Why should you Learn Github?

- All your code is on the cloud.
- Almost every company/group uses Github.
- It works with any language.
- It works with any operating system
- It allows you to share your code with others.
- Two people can program the same project at the same time.

How to create an account

Go to https://github.com/

Please sign up ***with your school email***. This will be important later.

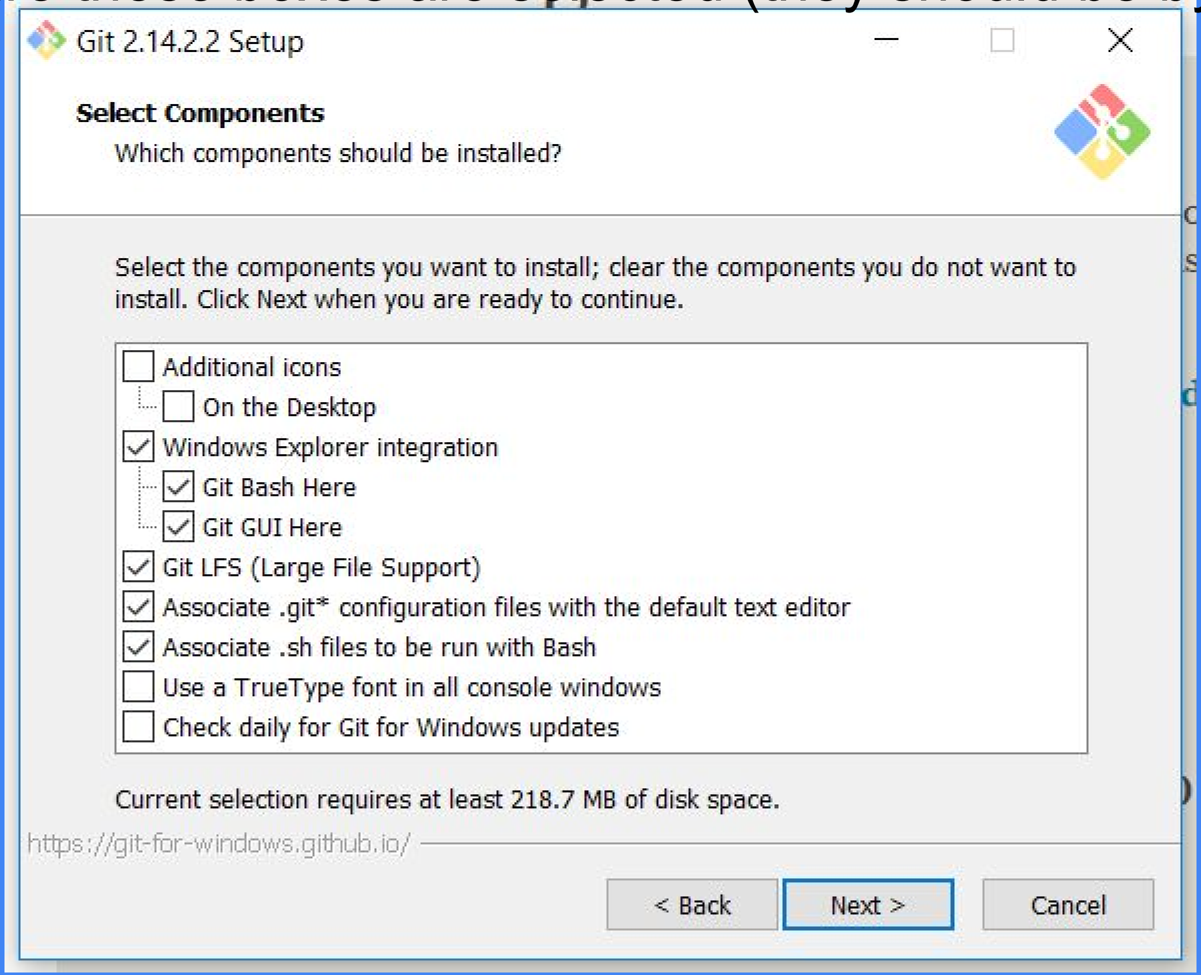
When signing up, pick the free option, you do not need to pay anything to use github.

Download Git Bash

Go to: https://git-scm.com/downloads

- Pick whatever operating system you are using
- The room’s computers should have them installed.

Make Sure these boxes are selected (they should be by default)


Git 2.14.2.2 Setup

Select Components

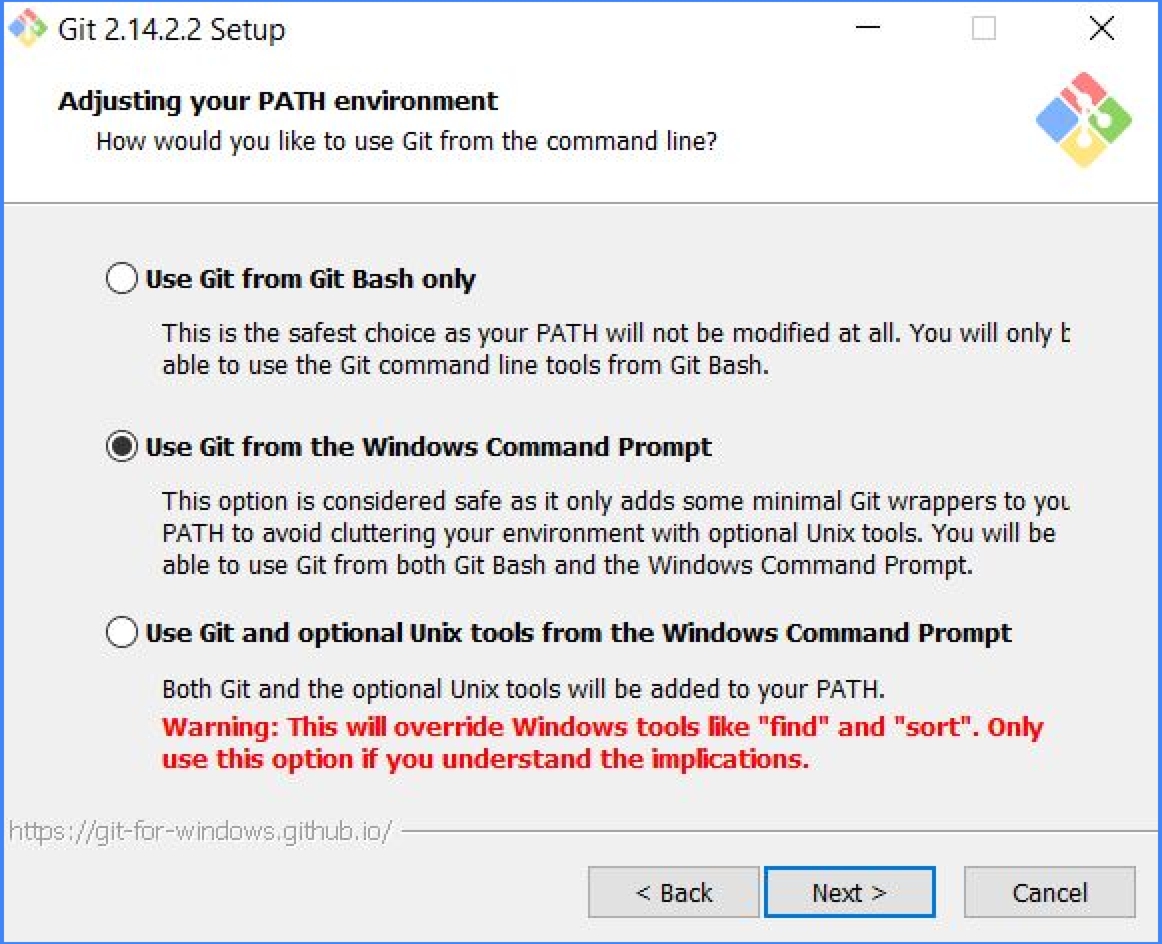
Which components should be installed?

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- [ ] Additional icons
  - [ ] On the Desktop
- [✓] Windows Explorer integration
  - [✓] Git Bash Here
  - [✓] Git GUI Here
- [✓] Git LFS (Large File Support)
- [✓] Associate .git* configuration files with the default text editor
- [✓] Associate .sh files to be run with Bash
- [ ] Use a TrueType font in all console windows
- [ ] Check daily for Git for Windows updates

Current selection requires at least 218.7 MB of disk space.

https://git-for-windows.github.io/


Git 2.14.2.2 Setup

Adjusting your PATH environment

How would you like to use Git from the command line?

○ **Use Git from Git Bash only**

This is the safest choice as your PATH will not be modified at all. You will only b
able to use the Git command line tools from Git Bash.

● **Use Git from the Windows Command Prompt**

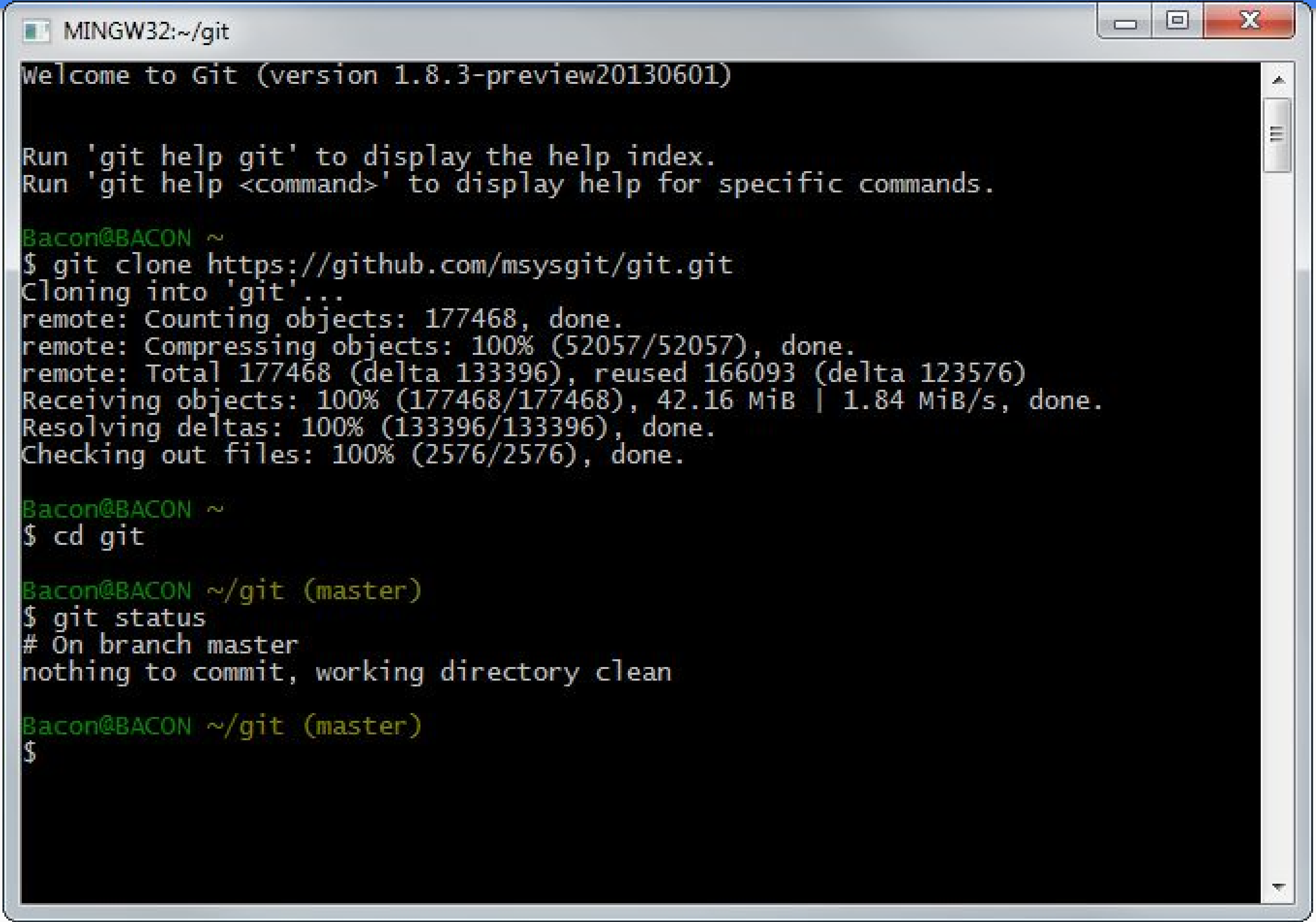
This option is considered safe as it only adds some minimal Git wrappers to you
PATH to avoid cluttering your environment with optional Unix tools. You will be
able to use Git from both Git Bash and the Windows Command Prompt.

○ **Use Git and optional Unix tools from the Windows Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

https://git-for-windows.github.io/

Now for the fun stuff



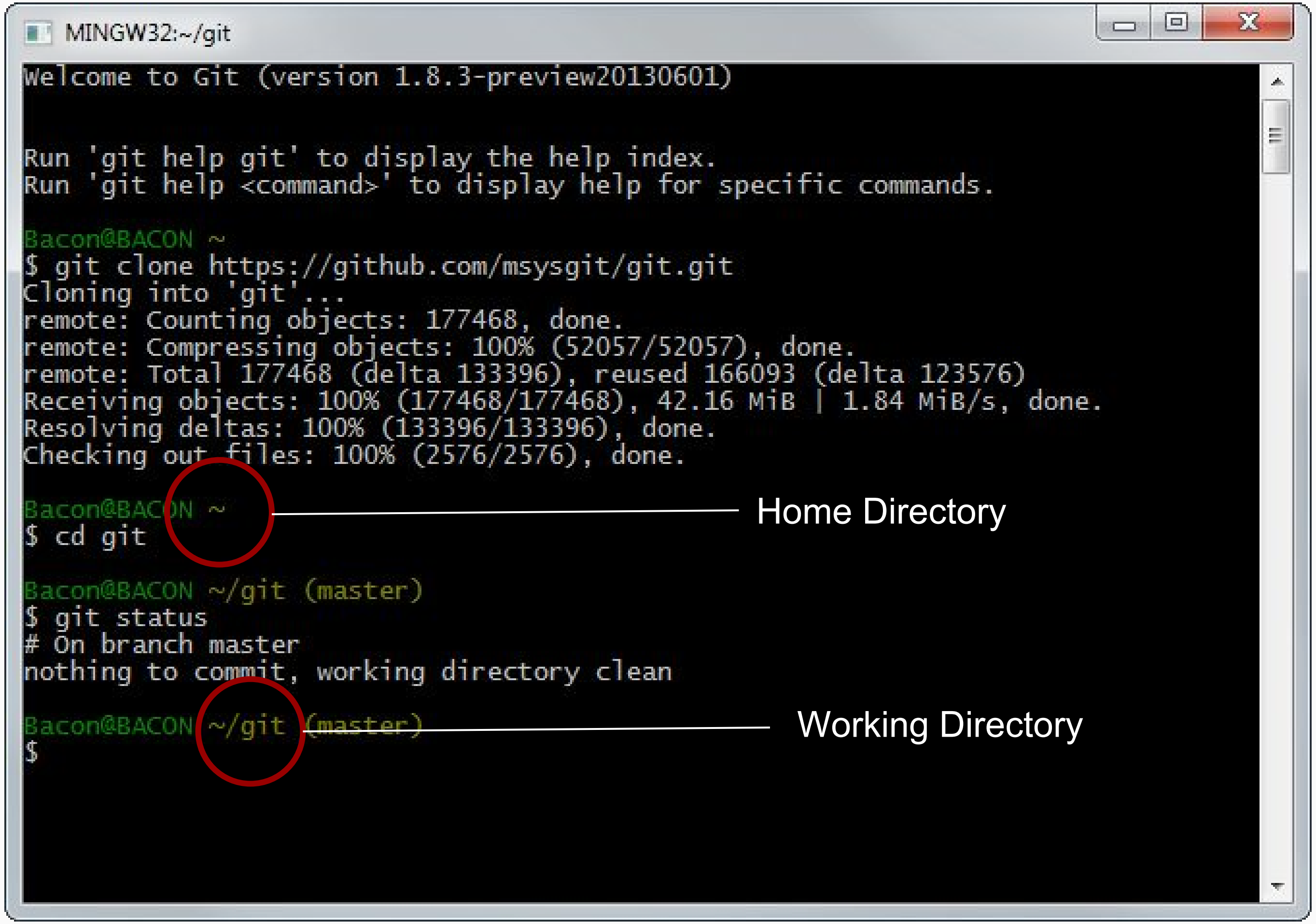
MINGW32:~/git

```
Welcome to Git (version 1.8.3-preview20130601)


Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Bacon@BACON ~
$ git clone https://github.com/msysgit/git.git
Cloning into 'git'...
remote: Counting objects: 177468, done.
remote: Compressing objects: 100% (52057/52057), done.
remote: Total 177468 (delta 133396), reused 166093 (delta 123576)
Receiving objects: 100% (177468/177468), 42.16 MiB | 1.84 MiB/s, done.
Resolving deltas: 100% (133396/133396), done.
Checking out files: 100% (2576/2576), done.

Bacon@BACON ~
$ cd git

Bacon@BACON ~/git (master)
$ git status
# On branch master
nothing to commit, working directory clean

Bacon@BACON ~/git (master)
$
```

Learning how to use a terminal

It’s basically a text based adventure game, without the fun.

Very similar to Window’s command prompt



```
Welcome to Git (version 1.8.3-preview20130601)


Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Bacon@BACON ~
$ git clone https://github.com/msysgit/git.git
Cloning into 'git'...
remote: Counting objects: 177468, done.
remote: Compressing objects: 100% (52057/52057), done.
remote: Total 177468 (delta 133396), reused 166093 (delta 123576)
Receiving objects: 100% (177468/177468), 42.16 MiB | 1.84 MiB/s, done.
Resolving deltas: 100% (133396/133396), done.
Checking out files: 100% (2576/2576), done.

Bacon@BACON ~
$ cd git

Bacon@BACON ~/git (master)
$ git status
# On branch master
nothing to commit, working directory clean

Bacon@BACON ~/git (master)
$
```

Commands to know

mkdir - make a new directory

- mkdir projects

cd - change directory

- cd c:\Users\
- cd .. goes backwards to parent directory c:\

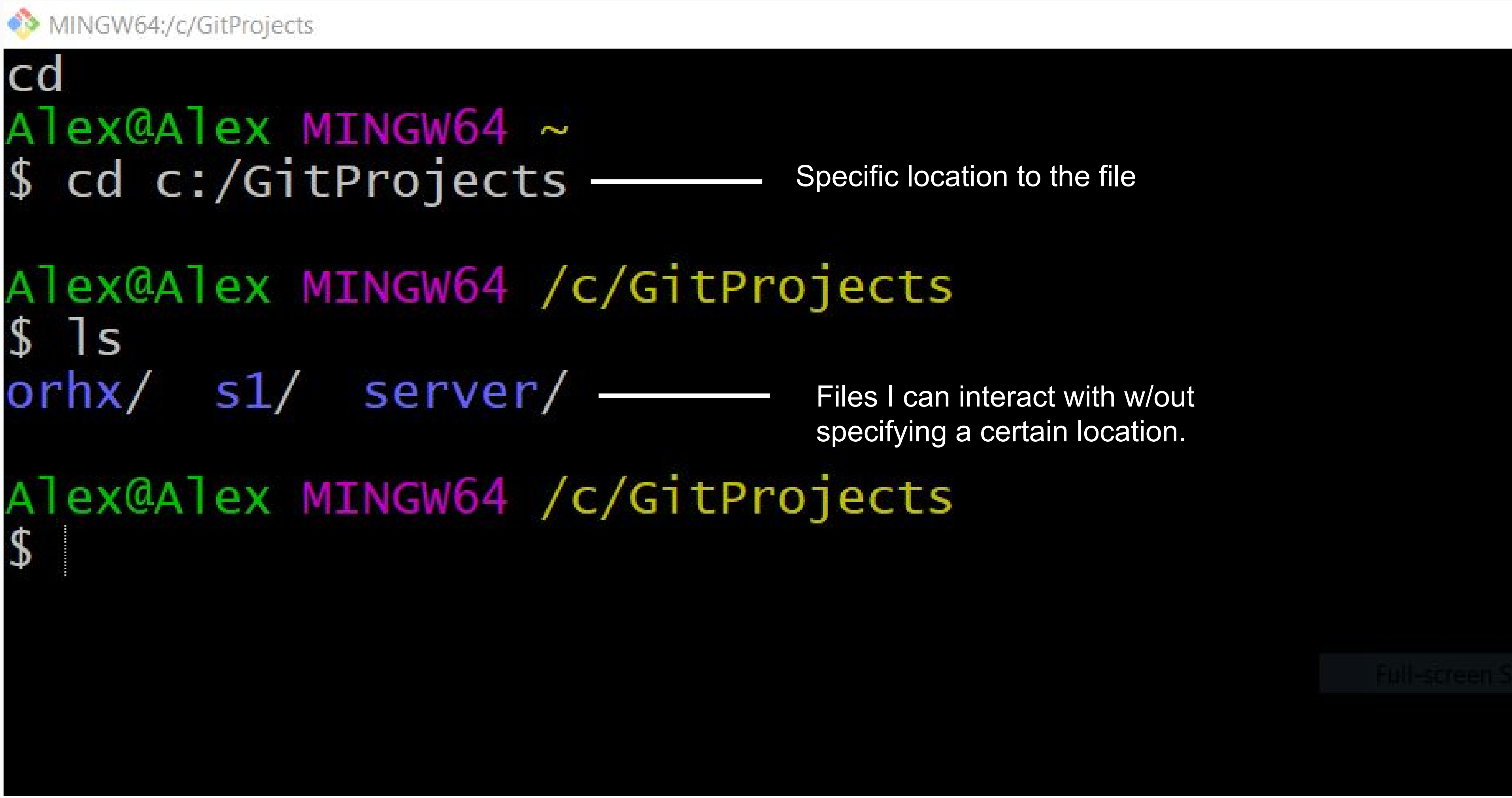
ls - list all files in the directory

Understanding how the terminal works

Think of a directory like a room

- Unless you give the specific location of the item you want to use, you can only access things in that particular room (directory)
- You can only get to some rooms by going through others.

Keep in mind when typing out locations if the location has a space in it, you need to add quotes around the location eg: “C:/Program Files/”


MINGW64:/c/GitProjects

```
cd
Alex@Alex MINGW64 ~
$ cd c:/GitProjects

Alex@Alex MINGW64 /c/GitProjects
$ ls
orhx/  s1/  server/

Alex@Alex MINGW64 /c/GitProjects
$
```

Specific location to the file
Files I can interact with w/out specifying a certain location.

Git-specific commands

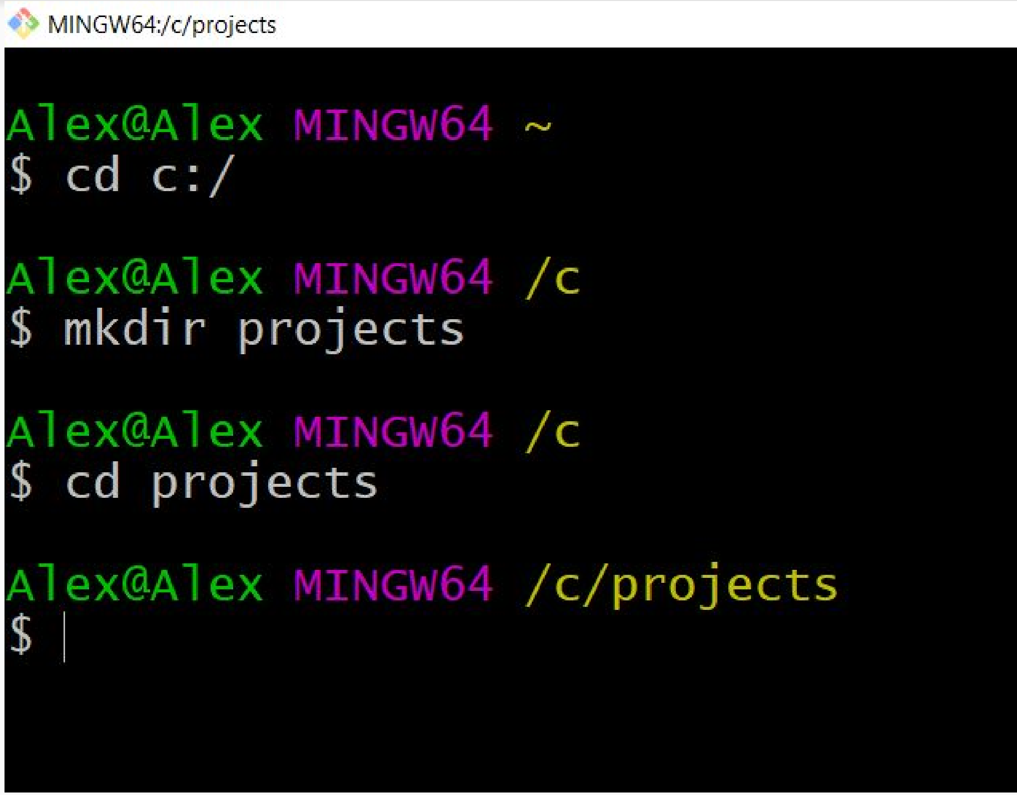
Every git-specific command will have the keyword “git” in front of it.

git init - Creates an empty git repository
git add - Adds files to the staging area
git commit - records changes to the repository
git push - moves local changes to the remote repository (cloud)
git pull - Download changes from remote repository (download and combine)

There are plenty more commands but these are the main five and the ones we’re going to focus on for right now.

Create your first Repository

- Navigate to your C:/ Drive
- Make a new directory and give it a name, something like GitProjects or projects etc. (A projects folder here is optional, we do it at c:\ so we don't have to do too much typing)
- Navigate to that new GitProjects folder

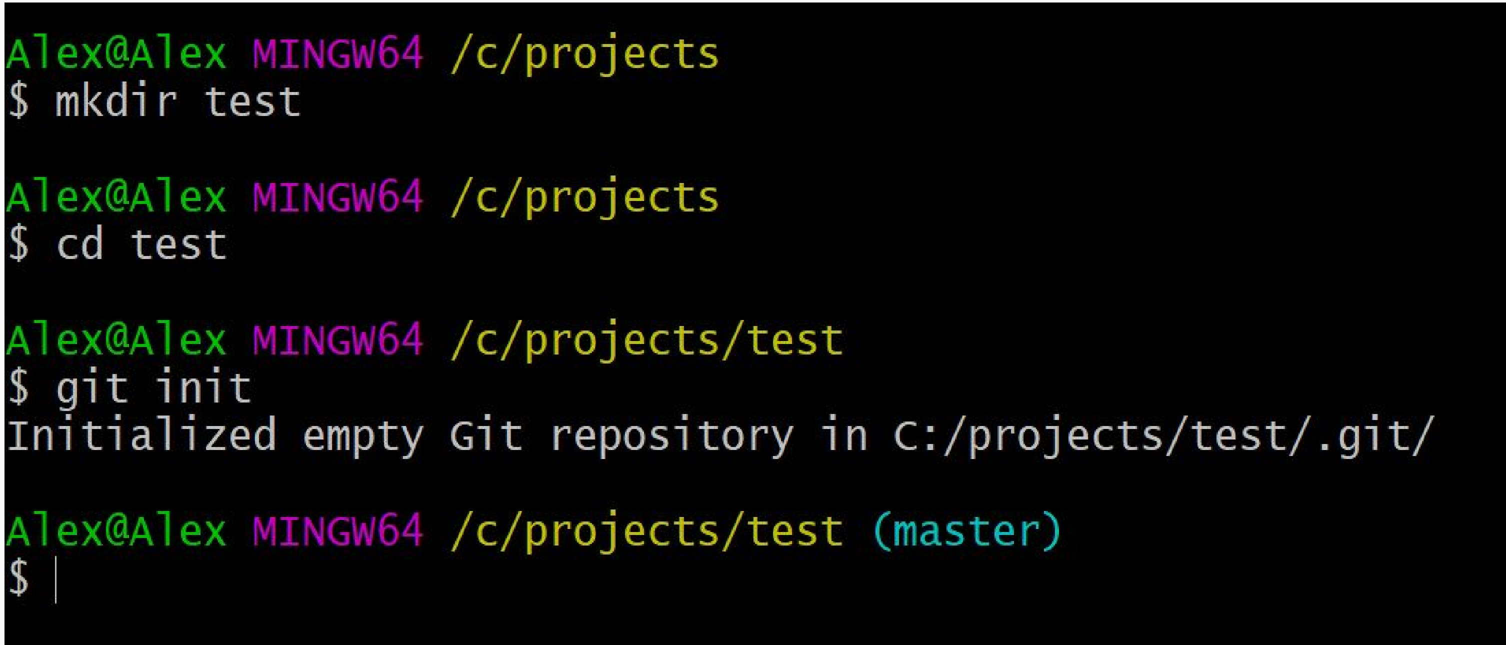


MINGW64:/c/projects

```
Alex@Alex MINGW64 ~
$ cd c:/

Alex@Alex MINGW64 /c
$ mkdir projects

Alex@Alex MINGW64 /c
$ cd projects

Alex@Alex MINGW64 /c/projects
$ 
```

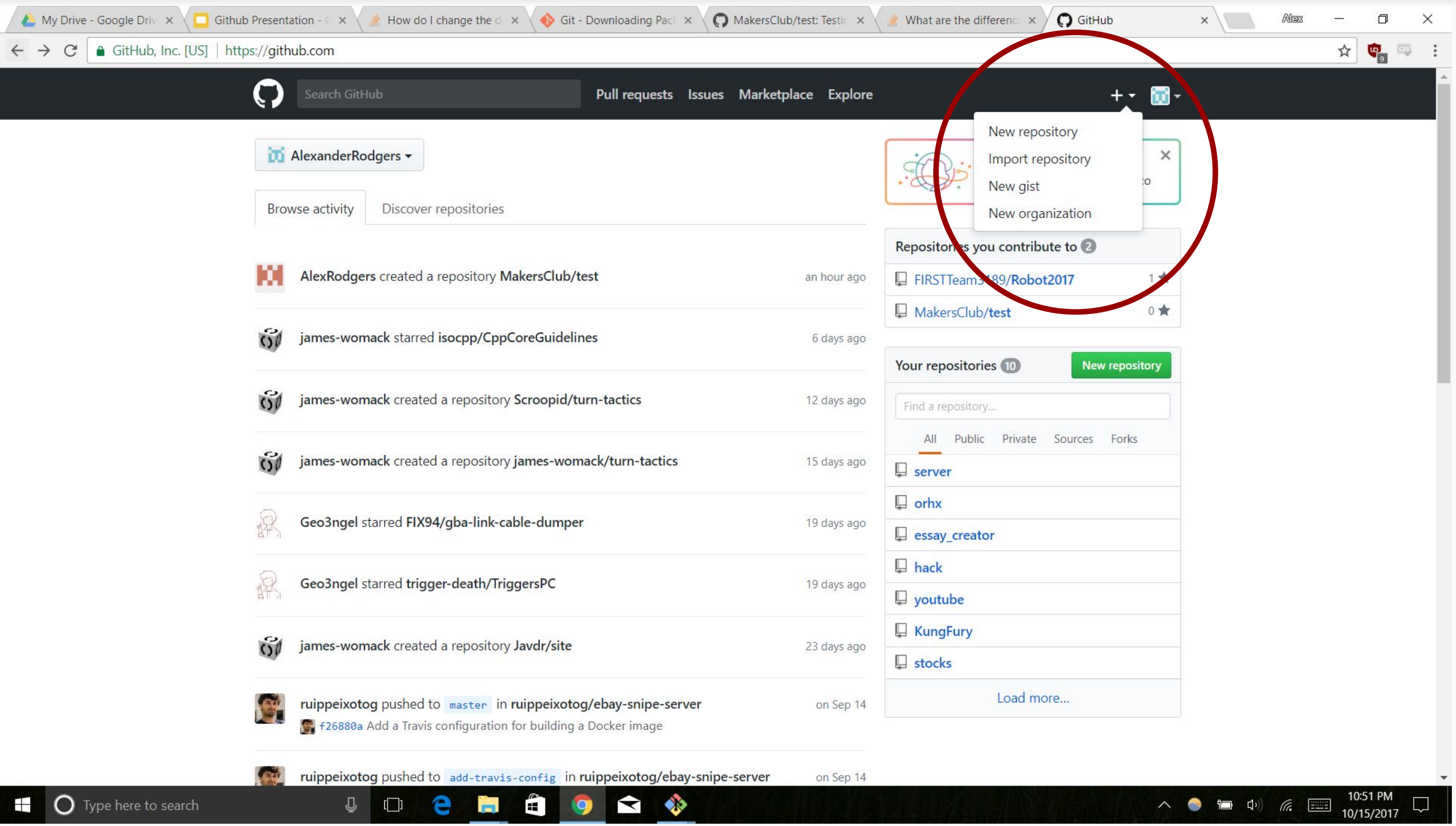
Create Your First Repository

- Make another directory and title it “test”
- Navigate to that test folder
- Initialize an empty git repository



```
Alex@Alex MINGW64 /c/projects
$ mkdir test

Alex@Alex MINGW64 /c/projects
$ cd test

Alex@Alex MINGW64 /c/projects/test
$ git init
Initialized empty Git repository in C:/projects/test/.git/

Alex@Alex MINGW64 /c/projects/test (master)
$ 
```

Create your First Repository

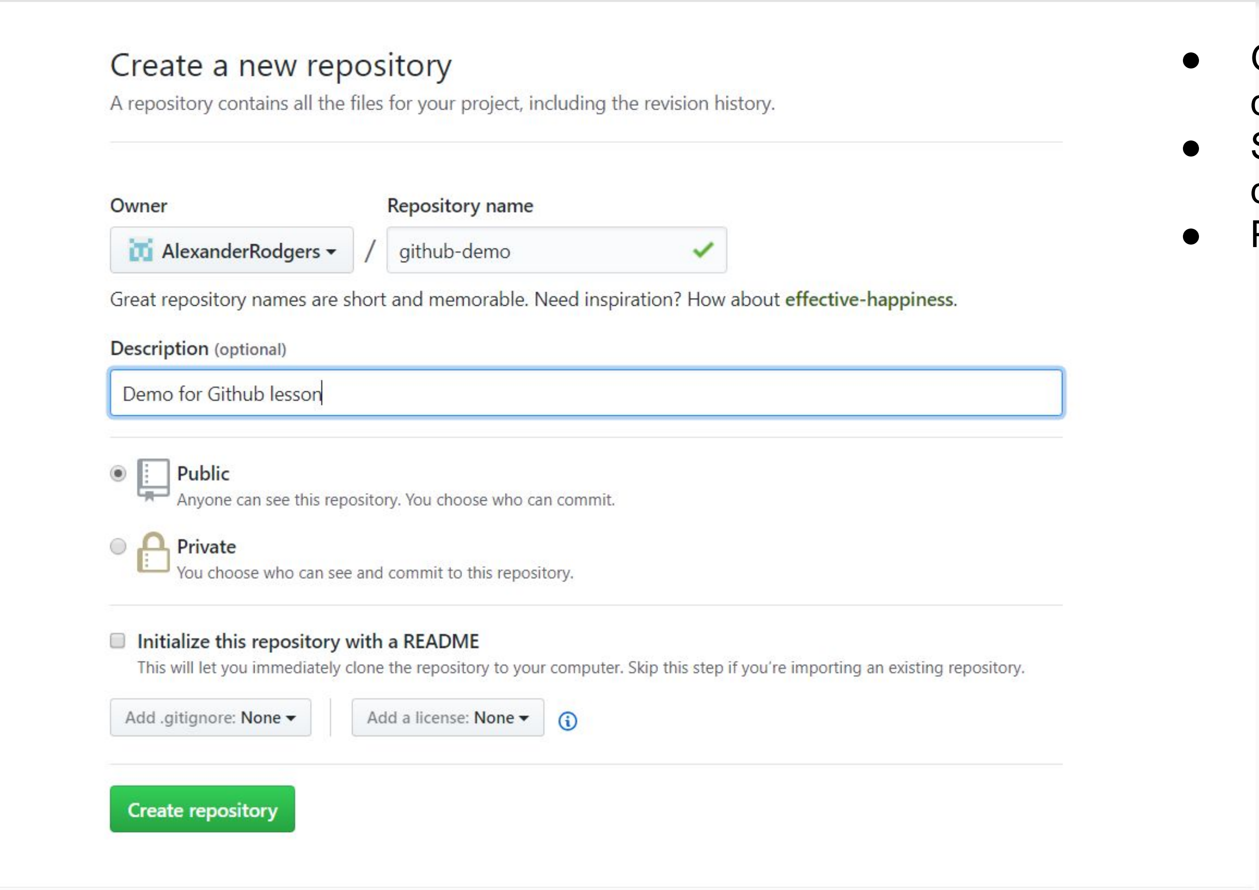


AlexRodgers created a repository MakersClub/test — an hour ago
james-womack starred isocpp/CppCoreGuidelines — 6 days ago
james-womack created a repository Scroopid/turn-tactics — 12 days ago
james-womack created a repository james-womack/turn-tactics — 15 days ago
Geo3ngel starred FIX94/gba-link-cable-dumper — 19 days ago
Geo3ngel starred trigger-death/TriggersPC — 19 days ago
james-womack created a repository Javdr/site — 23 days ago
ruippeixotog pushed to master in ruippeixotog/ebay-snipe-server — on Sep 14
f26880a Add a Travis configuration for building a Docker image
ruippeixotog pushed to add-travis-config in ruippeixotog/ebay-snipe-server — on Sep 14

-Click the + button

-Select New repository

Create your First Repository



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: AlexanderRodgers / Repository name: github-demo ✓

Great repository names are short and memorable. Need inspiration? How about effective-happiness.

Description (optional): Demo for Github lesson

[✓ Public] Anyone can see this repository. You choose who can commit.

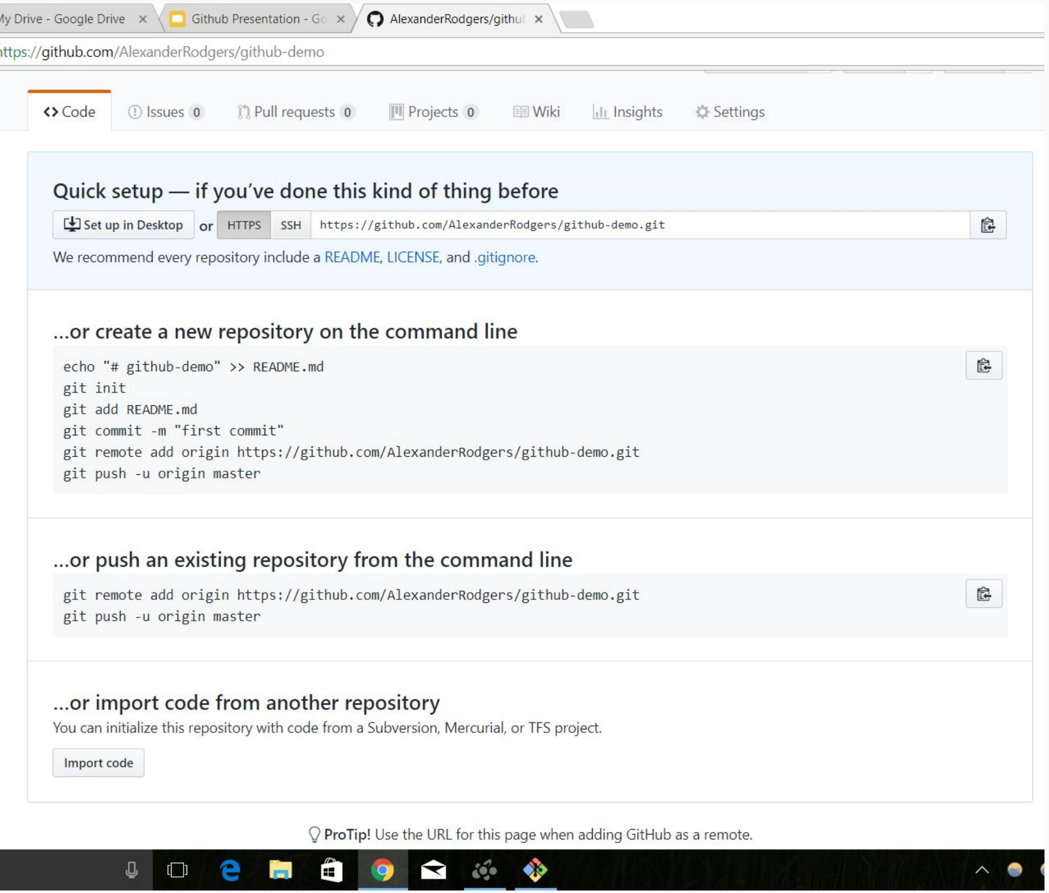
Private You choose who can see and commit to this repository.

☐ Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None | Add a license: None

- Give your repository a name and description.
- Select **public** under the repository description
- Press the green create button

Create your First Repository



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

Set up in Desktop or HTTPS SSH `https://github.com/AlexanderRodgers/github-demo.git`

We recommend every repository include a README, LICENSE, and .gitignore.

…or create a new repository on the command line

```
echo "# github-demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/AlexanderRodgers/github-demo.git
git push -u origin master
```

…or push an existing repository from the command line

```
git remote add origin https://github.com/AlexanderRodgers/github-demo.git
git push -u origin master
```

…or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

ProTip! Use the URL for this page when adding GitHub as a remote.

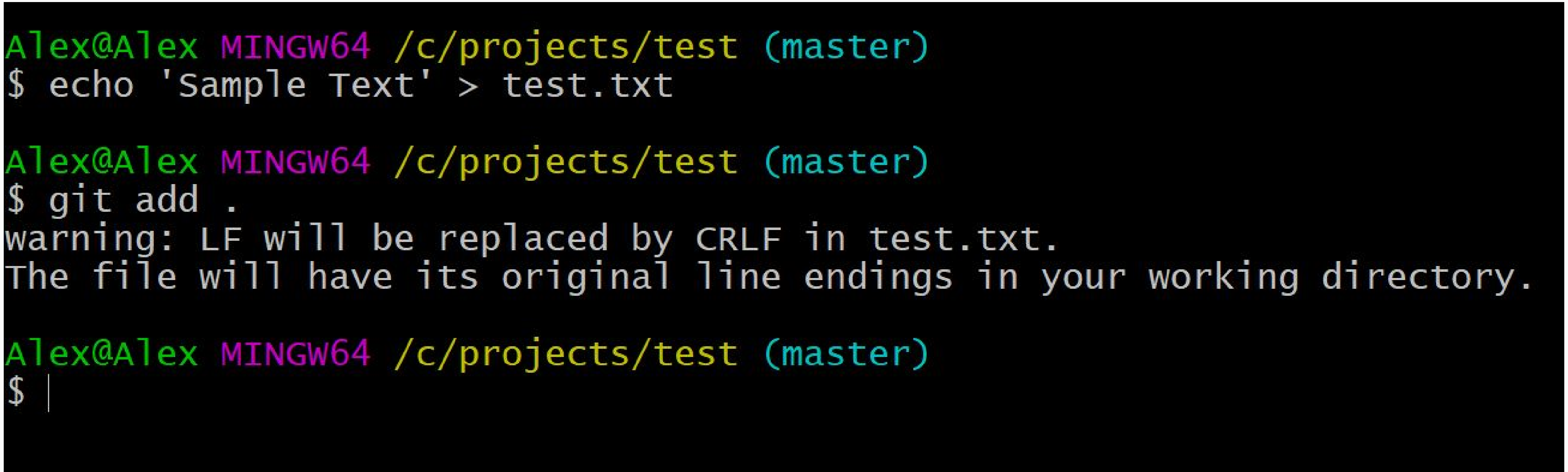
- You should be seeing a webpage that looks something like this.
- Don't exit from this website just yet.

Create your First Repository

Let’s create a file to upload.

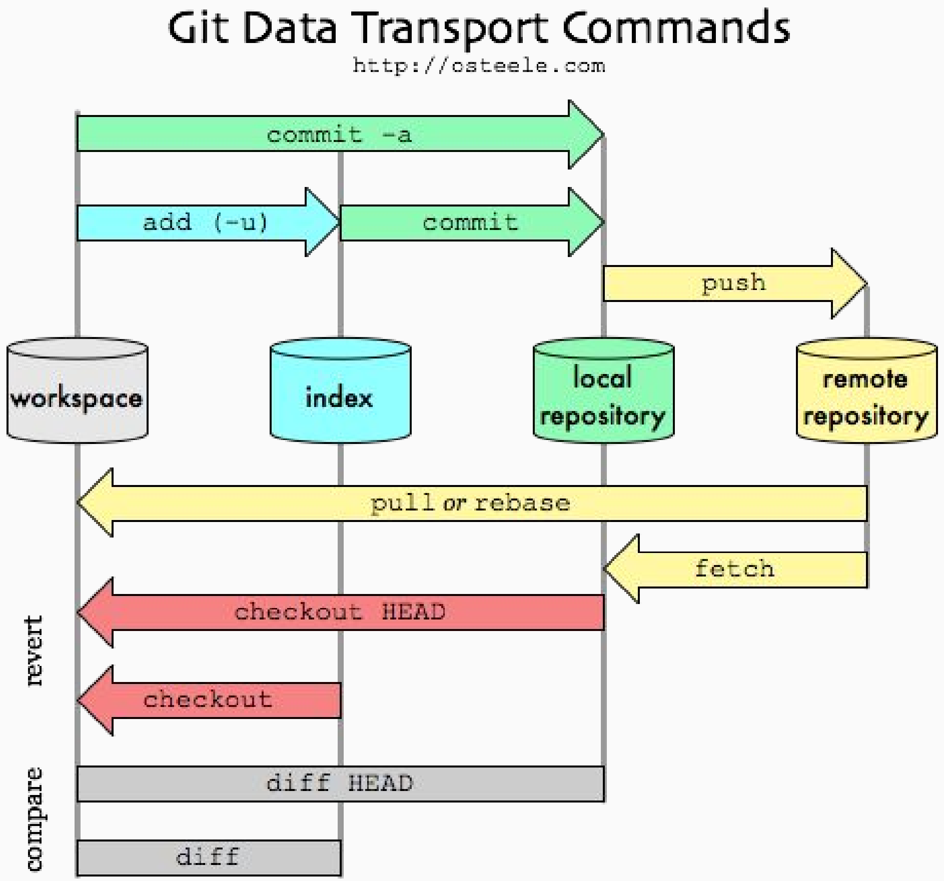
- Create a .txt file through console.
- Add the file to your project (the . means add all new files in the directory)

Don’t worry about the warning, it’s not important.



```
Alex@Alex MINGW64 /c/projects/test (master)
$ echo 'Sample Text' > test.txt

Alex@Alex MINGW64 /c/projects/test (master)
$ git add .
warning: LF will be replaced by CRLF in test.txt.
The file will have its original line endings in your working directory.

Alex@Alex MINGW64 /c/projects/test (master)
$ 
```

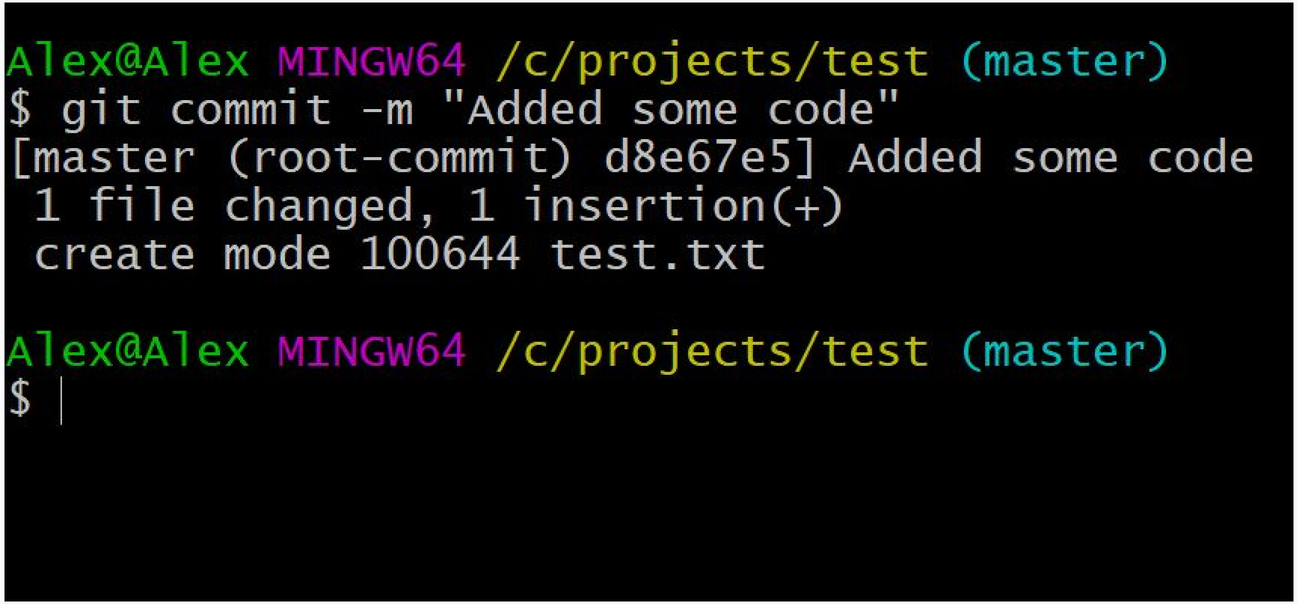
What are we doing?



- We're taking the code from our workspace and adding it to our local index (list of files in the project).
- Basically that means we are specifying what files we want to be part of our project.

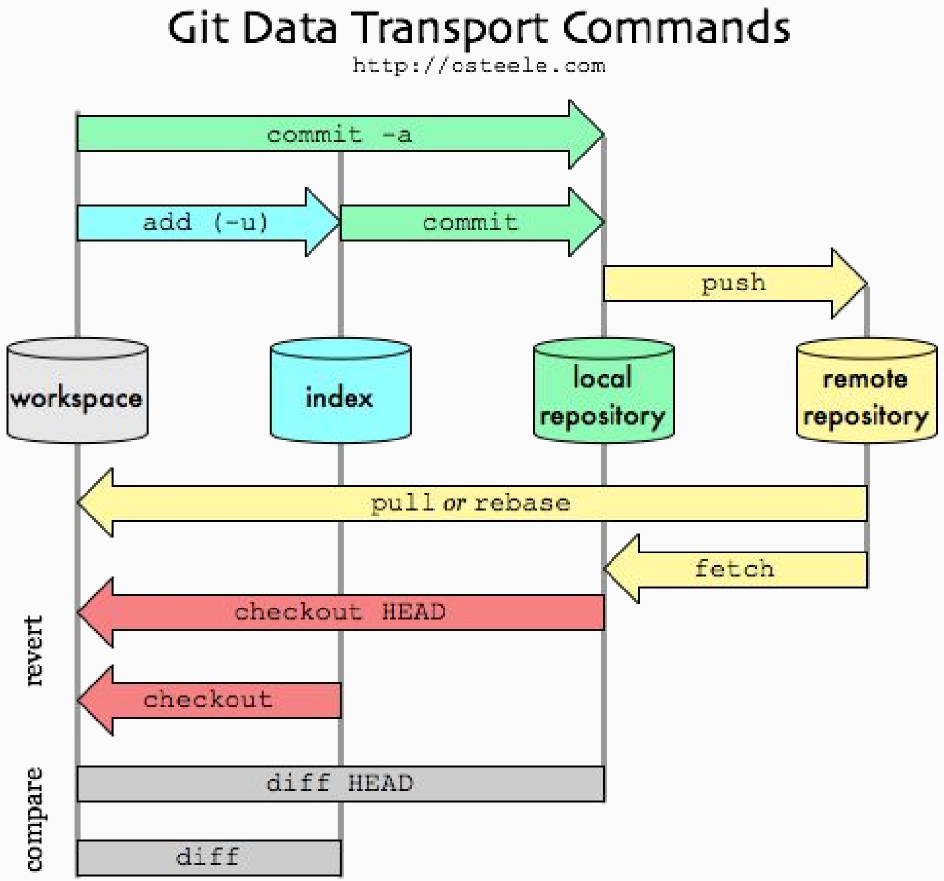
Create your first Repository

- Once you are satisfied with your code commit it by using:
  - git commit -m “*What changes you made to the project*”
  - The -m of this command tells git bash that you are also sending a message about the changes you made to this project.
  - You should ***always*** add a commit message to the changes you made and **be specific**.



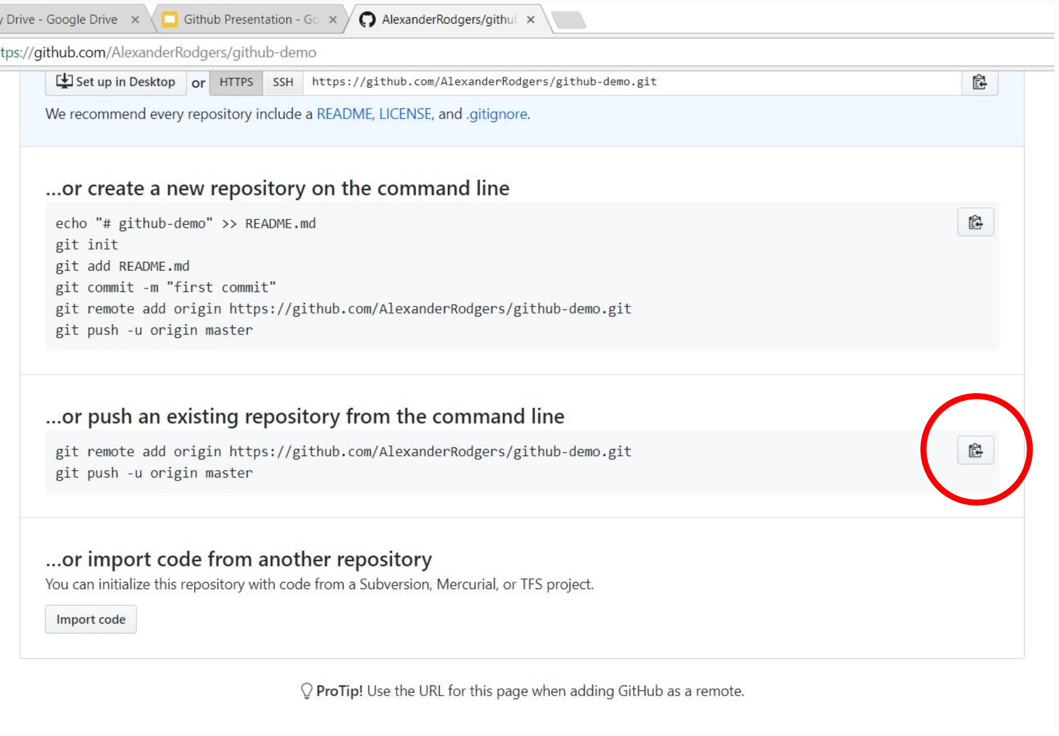
```
Alex@Alex MINGW64 /c/projects/test (master)
$ git commit -m "Added some code"
[master (root-commit) d8e67e5] Added some code
 1 file changed, 1 insertion(+)
 create mode 100644 test.txt

Alex@Alex MINGW64 /c/projects/test (master)
$ 
```

What are we doing?



- We are setting the final version of all our code that will be added to the group project.
- All the code that is committed is saved locally so that even if the code is changed, we can still use this version.

Create your First Repository



https://github.com/AlexanderRodgers/github-demo.git

We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# github-demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/AlexanderRodgers/github-demo.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/AlexanderRodgers/github-demo.git
git push -u origin master
```

...or import code from another repository

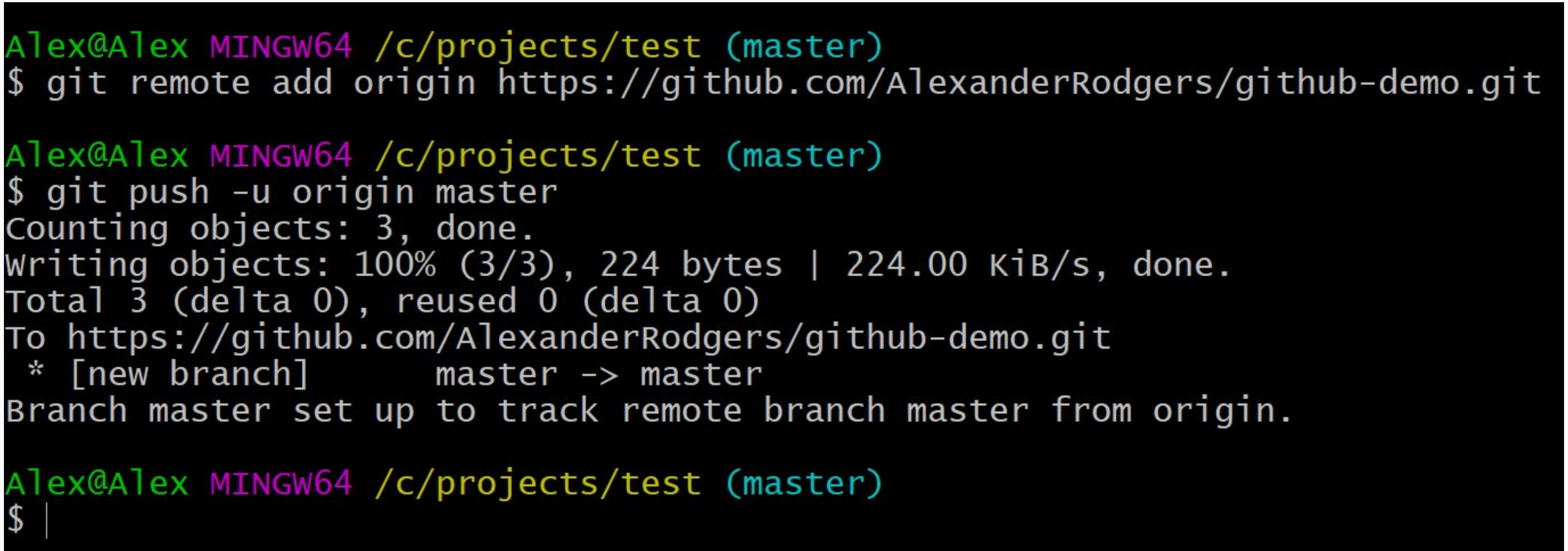
You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

ProTip! Use the URL for this page when adding GitHub as a remote.

- Go back to that website.
- Click on the copy to clipboard button under the “push an existing repository from the command line” section.

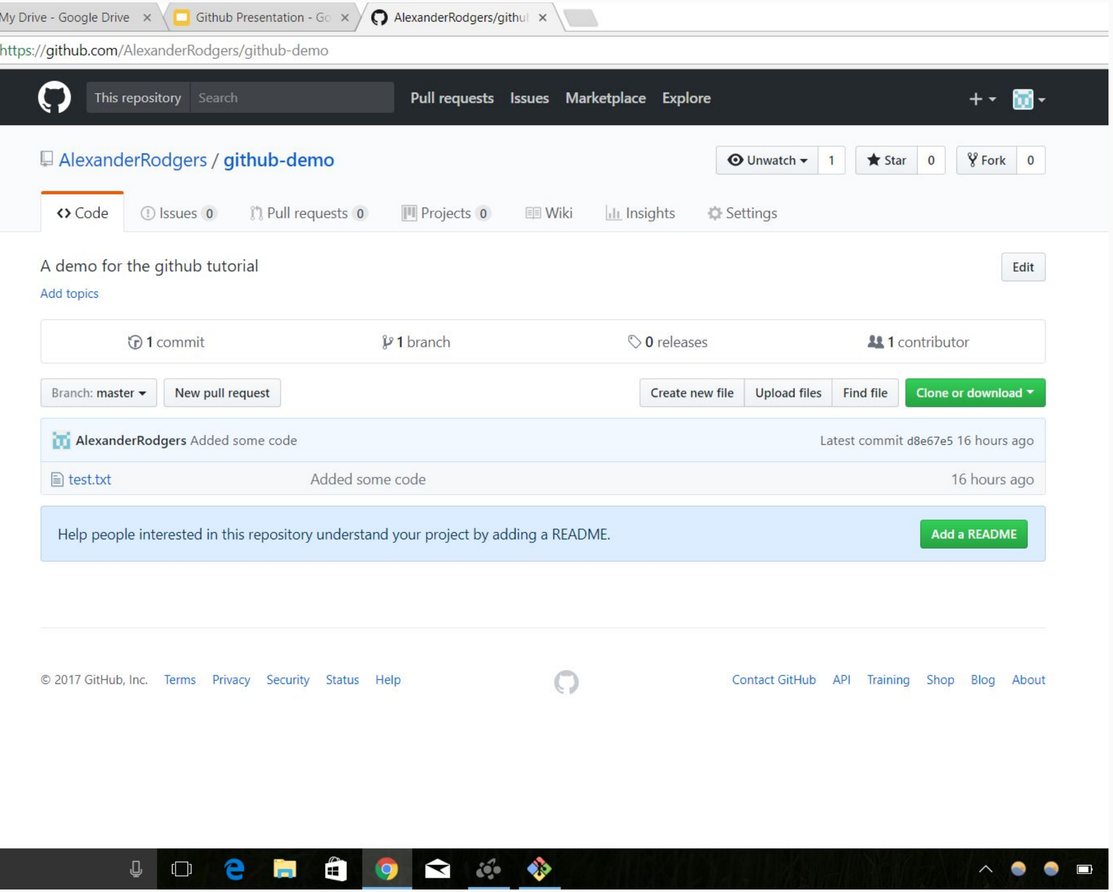
Create your First Repository

- Paste the code you just copied into the terminal.
  - You can paste into the terminal by right clicking with your mouse and selecting copy or using the shift+insert keys
- You may have to sign in. Just follow the instructions on the terminal.



```
Alex@Alex MINGW64 /c/projects/test (master)
$ git remote add origin https://github.com/AlexanderRodgers/github-demo.git

Alex@Alex MINGW64 /c/projects/test (master)
$ git push -u origin master
Counting objects: 3, done.
Writing objects: 100% (3/3), 224 bytes | 224.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/AlexanderRodgers/github-demo.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.

Alex@Alex MINGW64 /c/projects/test (master)
$ |
```

Create your First Repository



My Drive - Google Drive
Github Presentation - Go
AlexanderRodgers/githu

https://github.com/AlexanderRodgers/github-demo

This repository | Search | Pull requests | Issues | Marketplace | Explore

AlexanderRodgers / github-demo

Unwatch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

A demo for the github tutorial

Edit

Add topics

1 commit | 1 branch | 0 releases | 1 contributor

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

AlexanderRodgers Added some code — Latest commit d8e67e5 16 hours ago

test.txt — Added some code — 16 hours ago

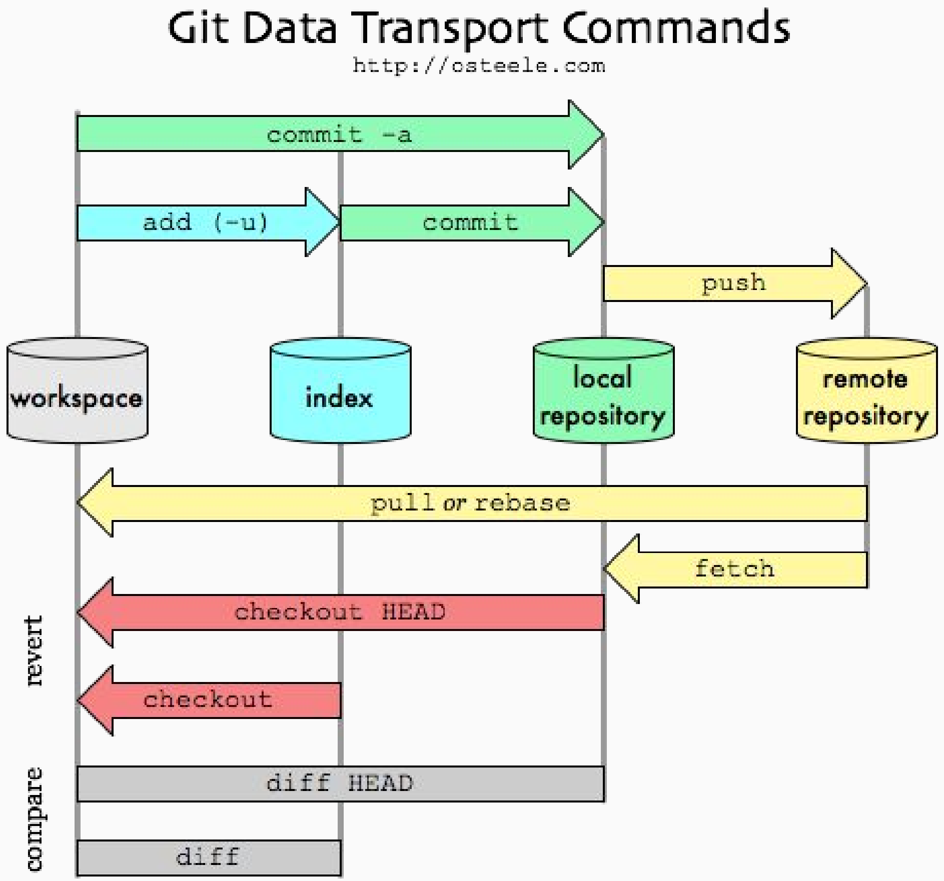
Help people interested in this repository understand your project by adding a README.

Add a README

© 2017 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub API Training Shop Blog About

- Refresh your browser and you should see that your file has been uploaded!

What did we do?



- We uploaded *our* version of the project to the group project online.
- Now anyone apart of the project can take the code you made and download it using git pull.

Sign up for the Github Education Pack

Go to https://education.github.com/pack

- Verify Academic Status
- Describe how you are going to use Github Education Pack (it does not have to be long)

Boom. Free stuff.

Additional Resources

Youtube:
- TheNewBoston
- LearnCode.Academy

https://guides.github.com/

https://try.github.io/levels/1/challenges/1