git

+

GitHub

AGENDA?

What is Git and GitHub?
Why use git?
Installation.
How to use git.

WHAT IS GIT?

- Git is a 'distributed version control system' - eh?

- That basically means...

    - It's a system that records changes to our files over time

    - We can recally specific versions of those files at any given time

    - Many people can easily collaborate on a project and have their own version of project files on their computer

WHY USE GIT?

Project X for client

PROJECT X

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

WHY USE GIT?

Project X for client

PROJECT X
Link 1
Link 2
Link 3
Link 4
Link 5
Lin
click me
random footer

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an un-known printer took a galley of type and scrambled it to make a type specimen book.

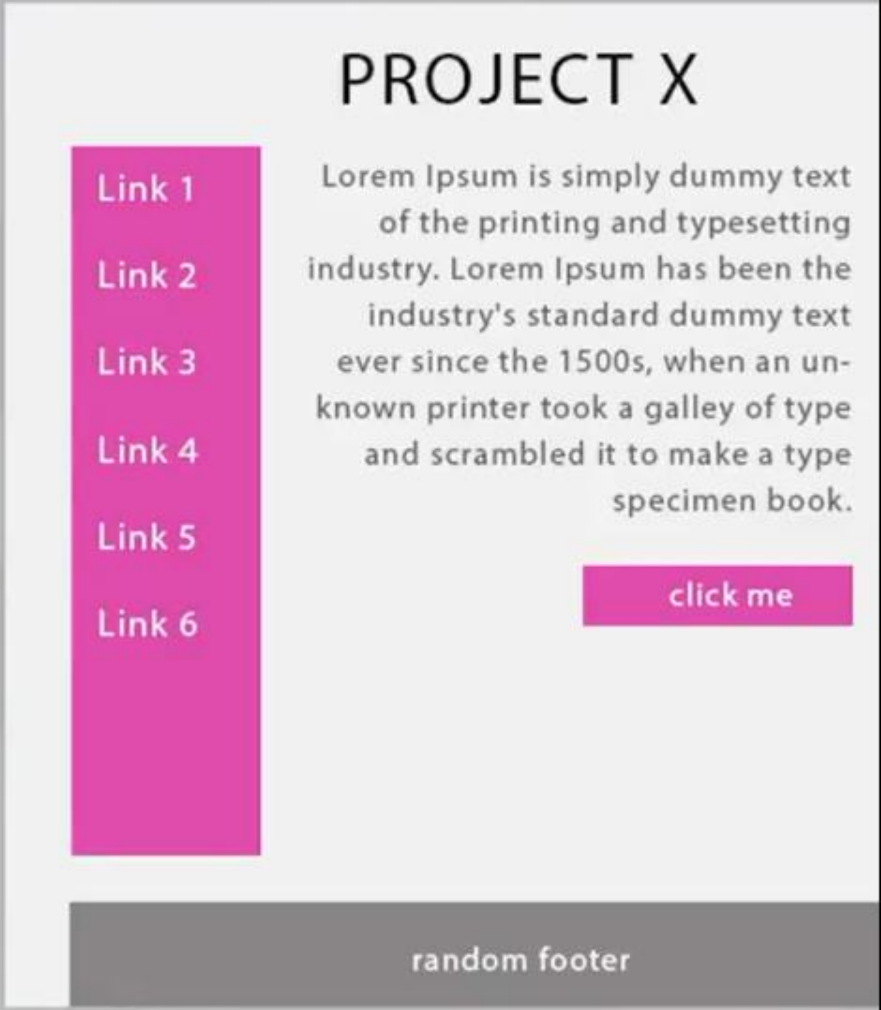
WHY USE GIT?

Project X for client

Lorem Ipsum is simply
of the printing an
industry. Lorem Ipsum
industry's standard
ever since the 1500s,
known printer took a
and scrambled it to
sp

WHY USE GIT?

Project X for client



PROJECT X

Link 1
Link 2
Link 3
Link 4
Link 5
Link 6

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

click me

random footer

WHY USE GIT?

Project X for client

WHY USE GIT?

- Project X for client - v1
- Project X for client - v2
- Project X for client - v3
- Project X for client - v4

WHY USE GIT?

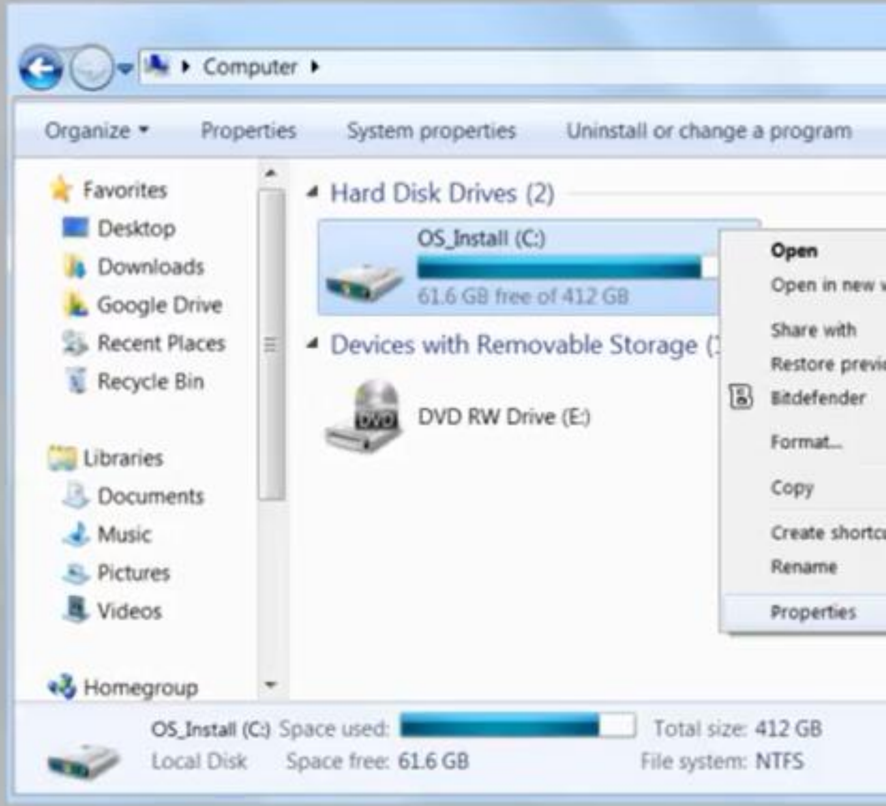
Project X for client - v1

Project X for client - v2

Project X for client - v3

...several days / emails later....

Project X for client - v25 !!!



WHY USE GIT?

- Store revisions in a project history in just one directory

- Rewind to any revision in the project I wanted to

- Work on new features without messing up the main codebase

- Easily collaborate with other programmers

GITHUB

- Online service that hosts our projects

- Share our code with other developers

- Developers can download the projects and work on them

- They can re-upload their edits and merge them with the main codebase

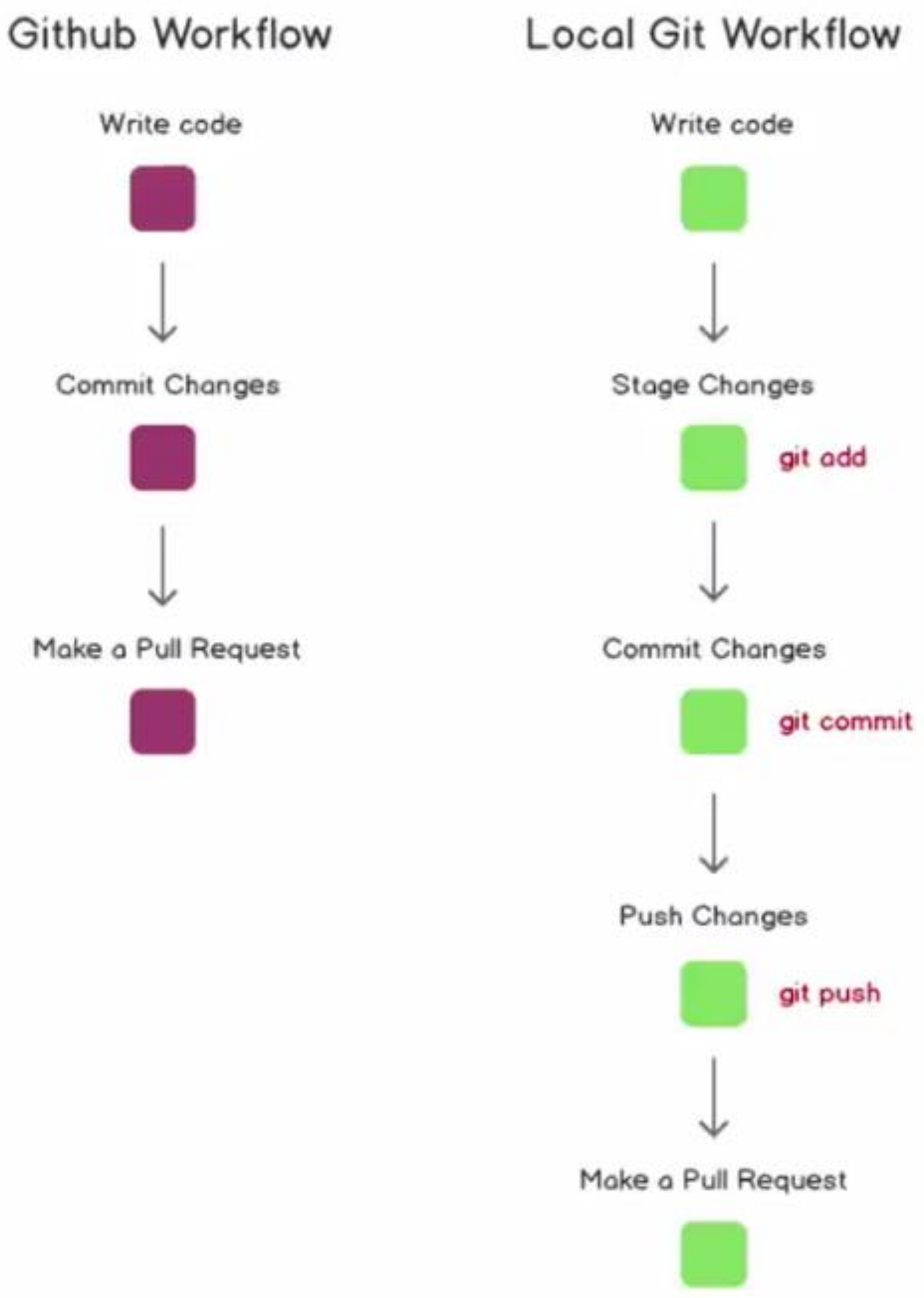
Terms

- Directory -> Folder
- Terminal or Command Line -> Interface for Text Commands
- CLI -> Command Line Interface
- cd -> Change Directory
- Code Editor -> Word Processor for Writing Code
- Repository -> Project, or the folder/place where your project is kept
- Github -> A website to host your repositories online

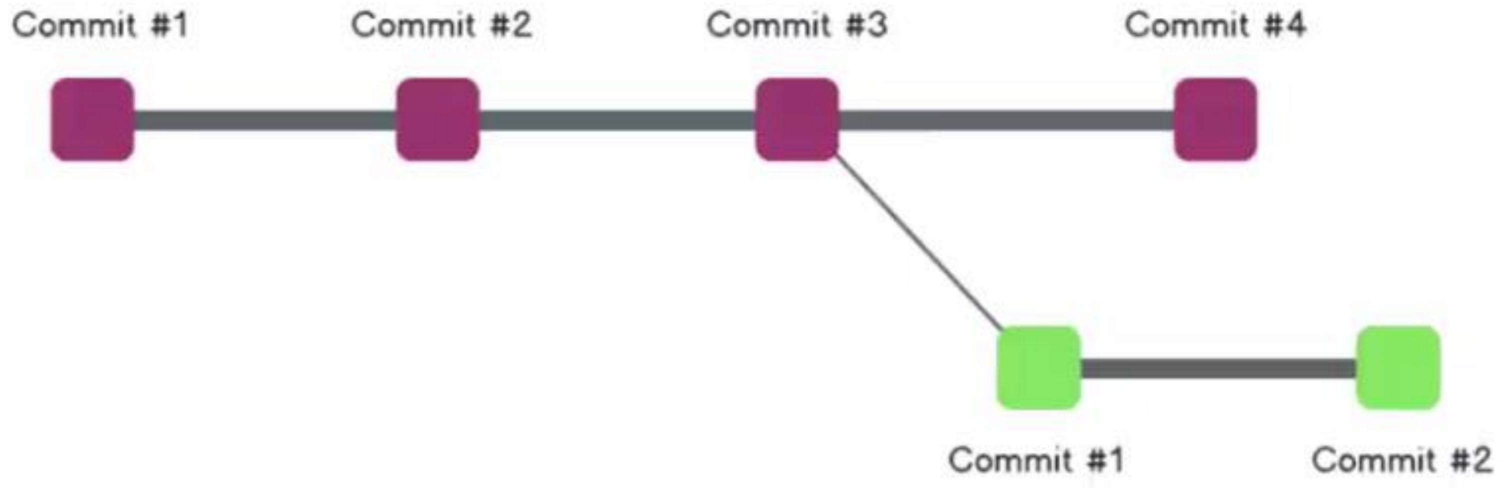
Git Commands

- Clone -> Bring a repository that is hosted somewhere like Github into a folder on your local machine
- add -> Track your files and changes in Git
- commit -> Save your files in Git
- push -> Upload Git commits to a remote repo, like Github
- pull -> Download changes from remote repo to your local machine, the opposite of push

- repo -> repository
- `clone` -> bring a repo down from the internet (remote repository like Github) to your local machine
- `add` -> track your files and changes with Git
- `commit` -> save your changes into Git
- `push` -> push your changes to your remote repo on Github (or another website)
- `pull` -> pull changes down from the remote repo to your local machine
- `status` -> check to see which files are being tracked or need to be commited
- `init` -> use this command inside of your project to turn it into a Git repository and start using Git with that codebase

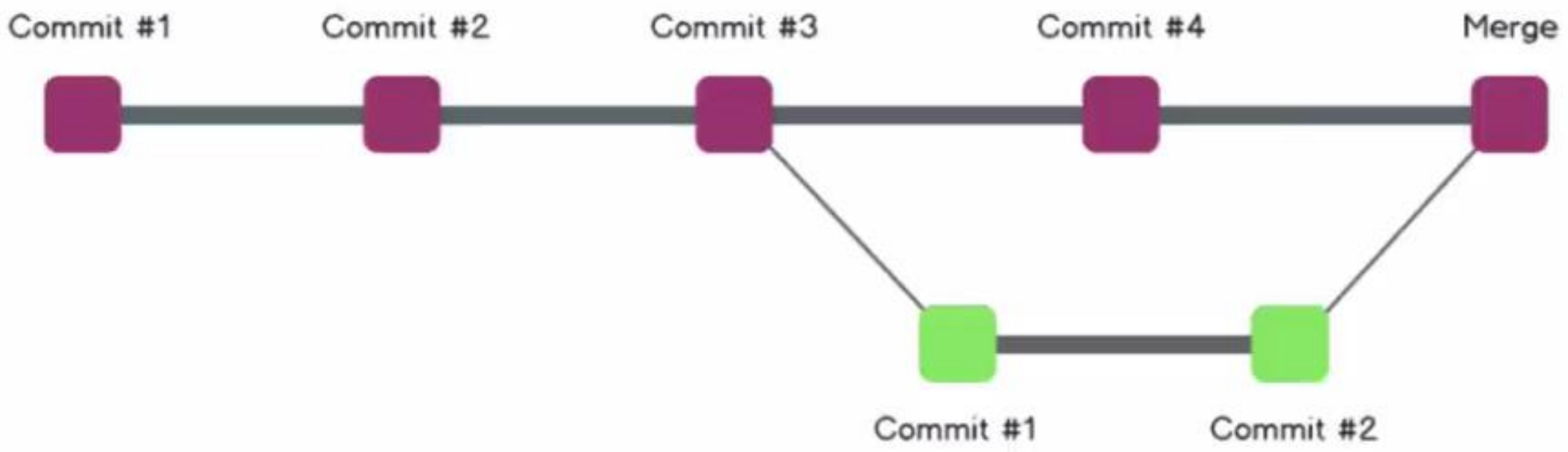

Github Workflow
Write code
Commit Changes
Make a Pull Request
Local Git Workflow
Write code
Stage Changes
git add
Commit Changes
git commit
Push Changes
git push
Make a Pull Request

Git Branching

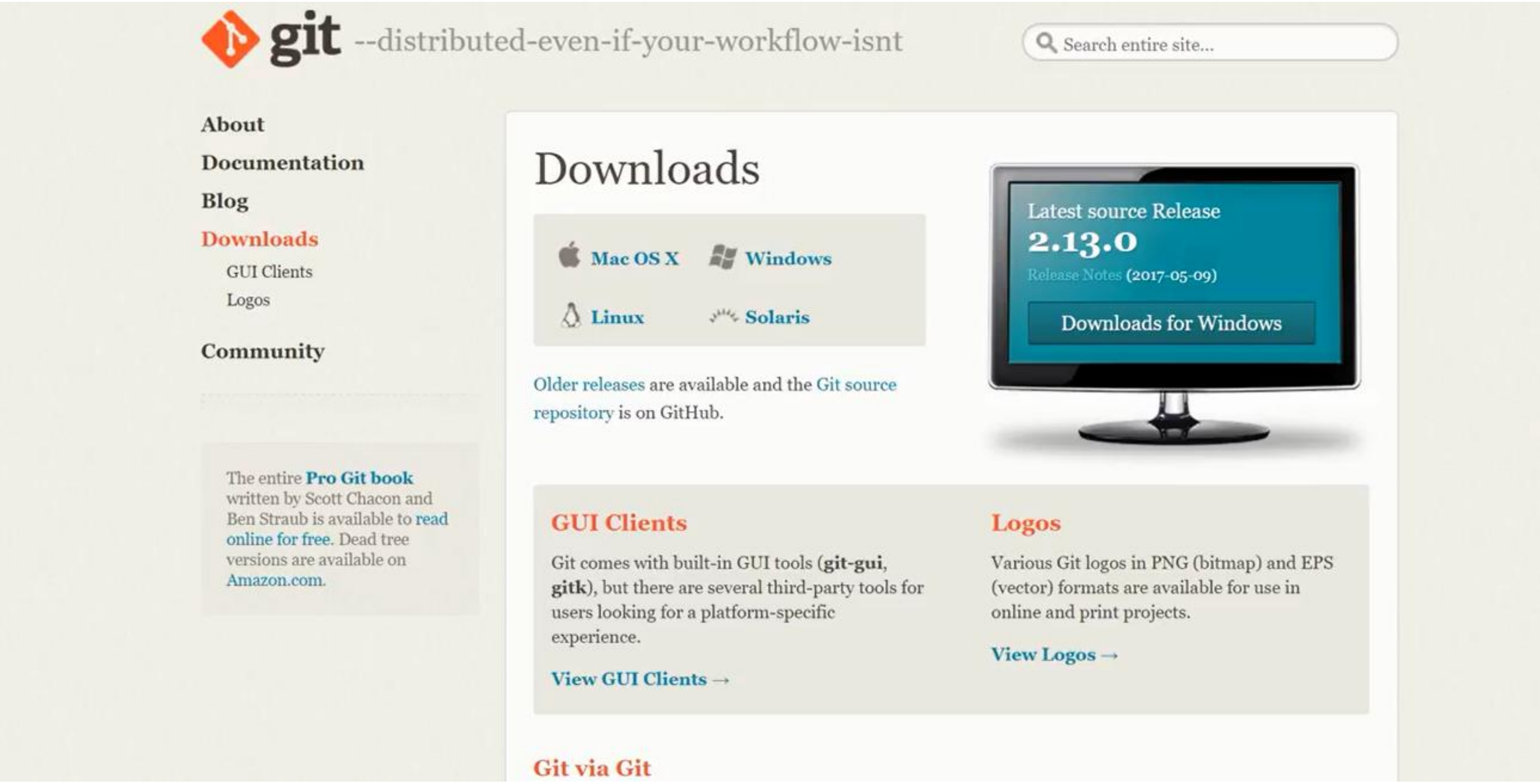


🟪 Master Branch 🟩 Feature Branch

Git Branching



🟪 Master Branch 🟩 Feature Branch



git --distributed-even-if-your-workflow-isnt

Search entire site...

About
Documentation
Blog
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Downloads

Mac OS X **Windows**
Linux **Solaris**

Older releases are available and the Git source repository is on GitHub.

GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

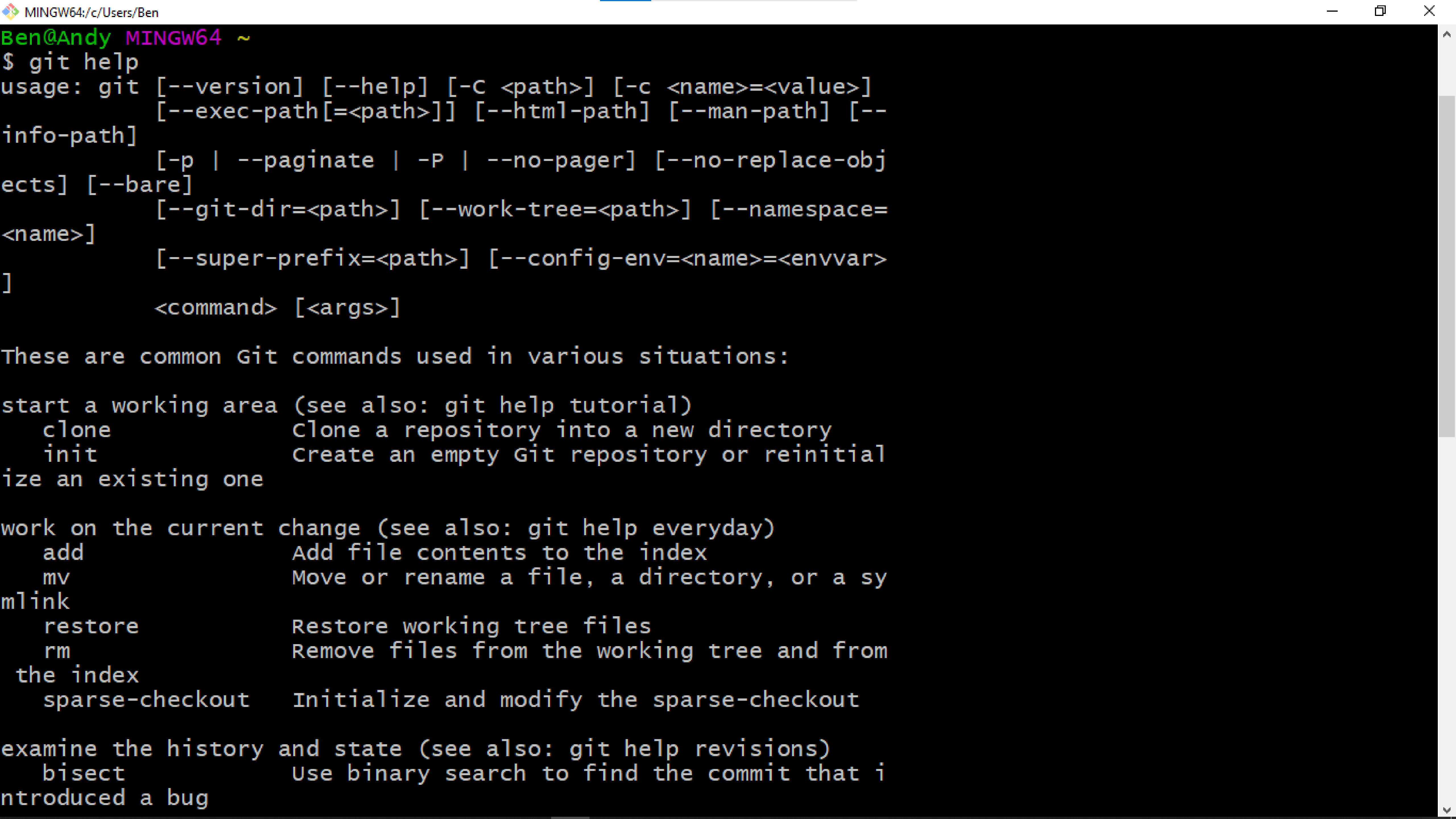
View GUI Clients →

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

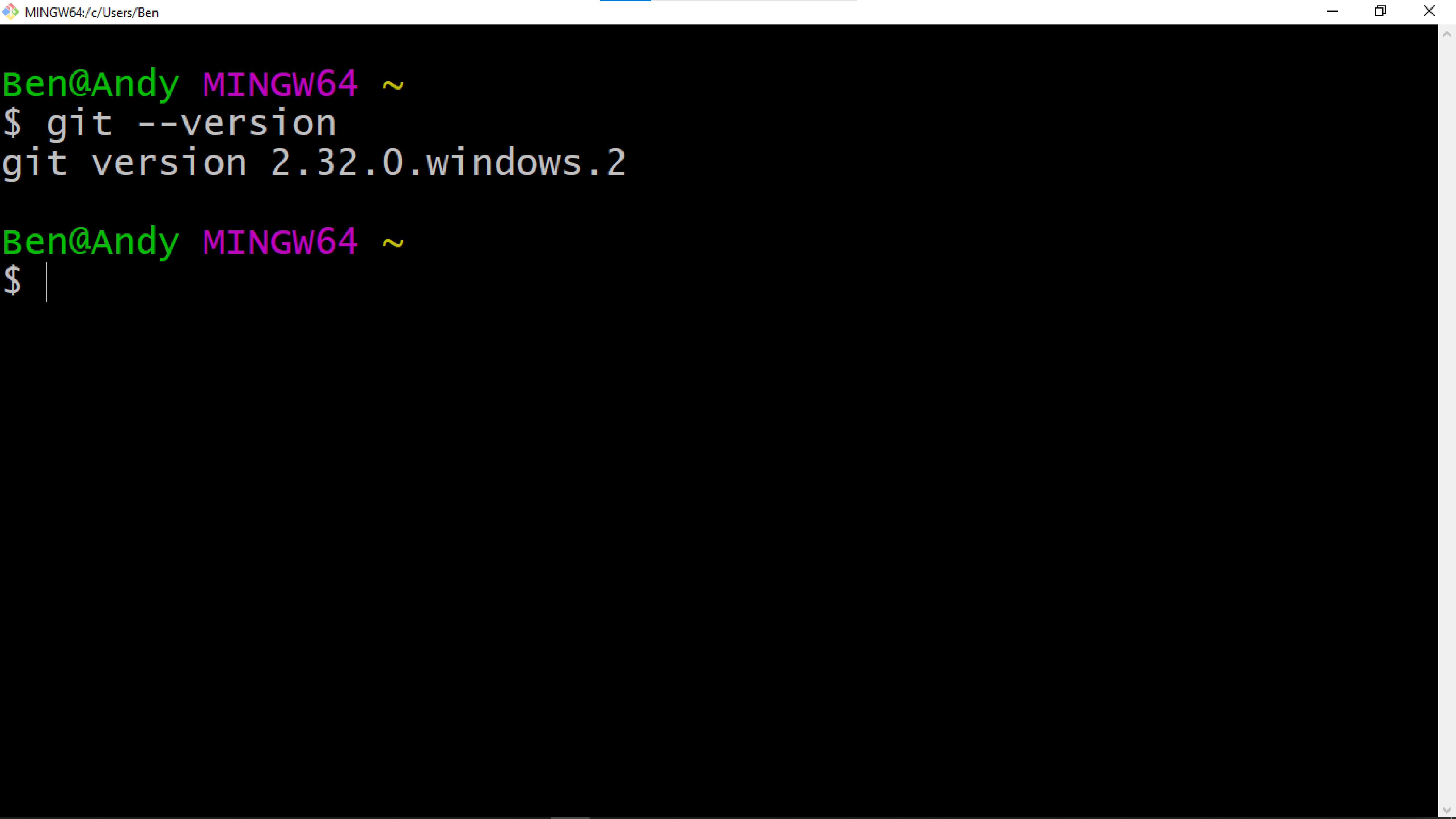
View Logos →

Git via Git



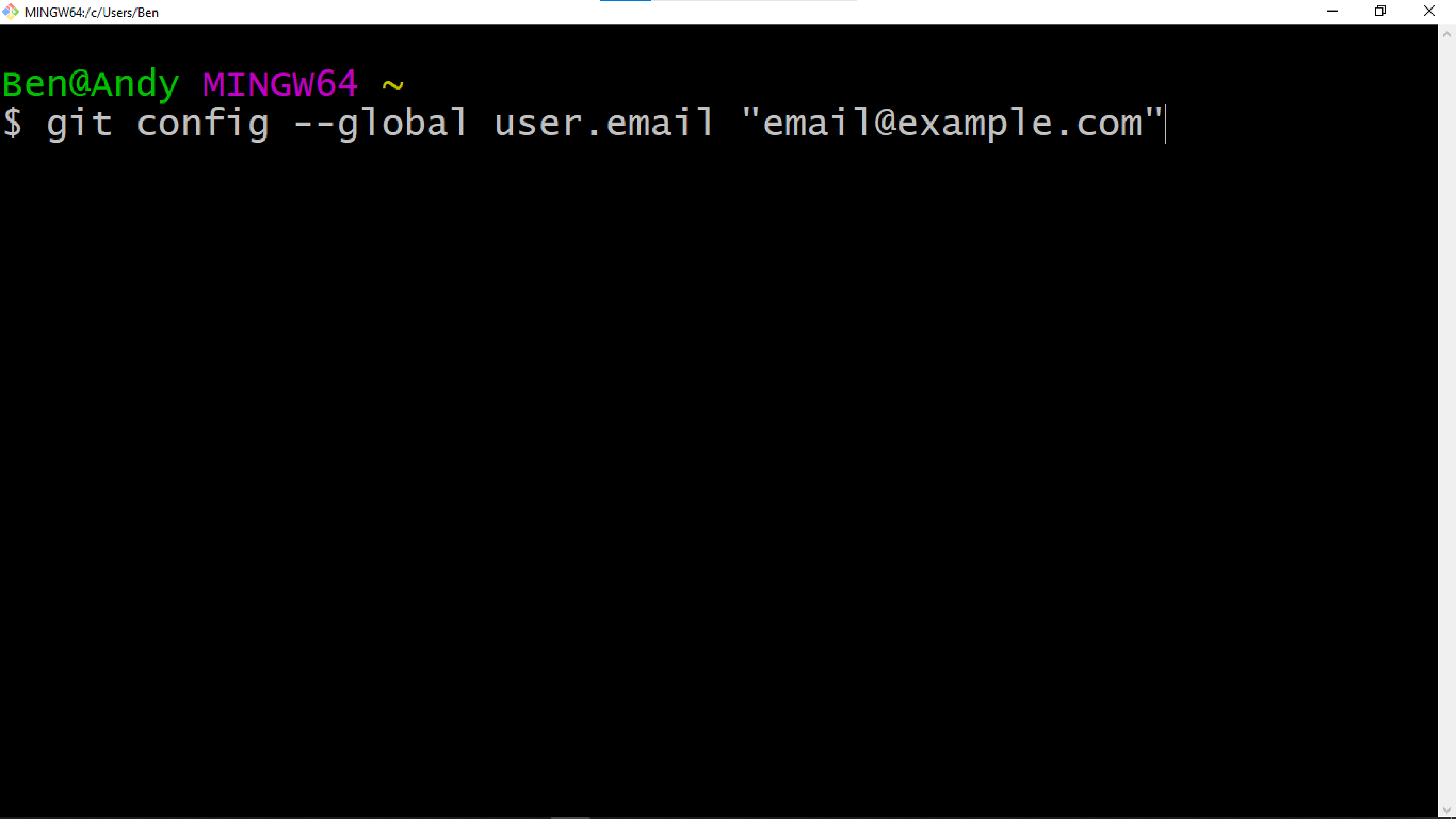
MINGW64:/c/Users/Ben

```
Ben@Andy MINGW64 ~
$ git help
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--
info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-obj
ects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=
<name>]
           [--super-prefix=<path>] [--config-env=<name>=<envvar>
]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone             Clone a repository into a new directory
   init              Create an empty Git repository or reinitial
ize an existing one

work on the current change (see also: git help everyday)
   add               Add file contents to the index
   mv                Move or rename a file, a directory, or a sy
mlink
   restore           Restore working tree files
   rm                Remove files from the working tree and from
 the index
   sparse-checkout   Initialize and modify the sparse-checkout

examine the history and state (see also: git help revisions)
   bisect            Use binary search to find the commit that i
ntroduced a bug
```



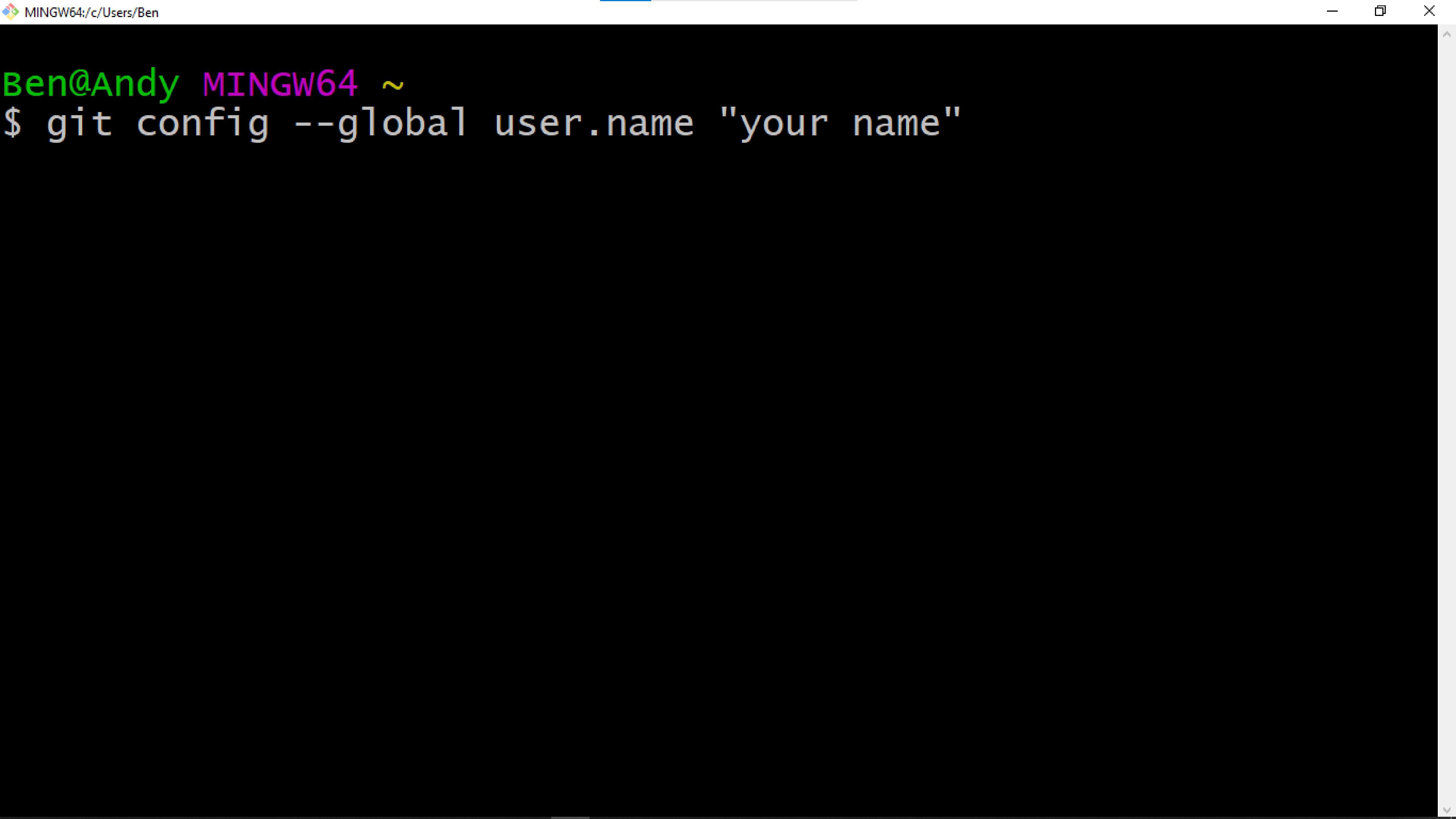
MINGW64:/c/Users/Ben

```
Ben@Andy MINGW64 ~
$ git --version
git version 2.32.0.windows.2

Ben@Andy MINGW64 ~
$ 
```



MINGW64:/c/Users/Ben

```
Ben@Andy MINGW64 ~
$ git config --global user.email "email@example.com"
```


MINGW64:/c/Users/Ben

```
Ben@Andy MINGW64 ~
$ git config --global user.name "your name"
```

Advanced

Create a new branch

Let's try something a little more advanced. Say you want to make a new feature but are worried about making changes to the main project while developing the feature. This is where git branches come in.

Branches allow you to move back and forth between 'states' of a project. Official git docs describe branches this way: ‘A branch in Git is simply a lightweight movable pointer to one of these commits

Advanced

Create a new branch

- git checkout -b <branch-name>
- Switch and merge: git checkout master/git switch master (or main, or your primary branch)
- git merge <branch-name>
- Then git push origin master

Optionally, you can delete the branch

- Locally: git branch -d <branch-name>
- Remotely: git push origin --delete <branch-name>

Roll back to a previous state

Temporarily go back (without changing history)

If you just want to check out a previous commit temporarily:

- git checkout <commit_hash> (get hash with git log –oneline)

Create a new commit that reverts a previous one

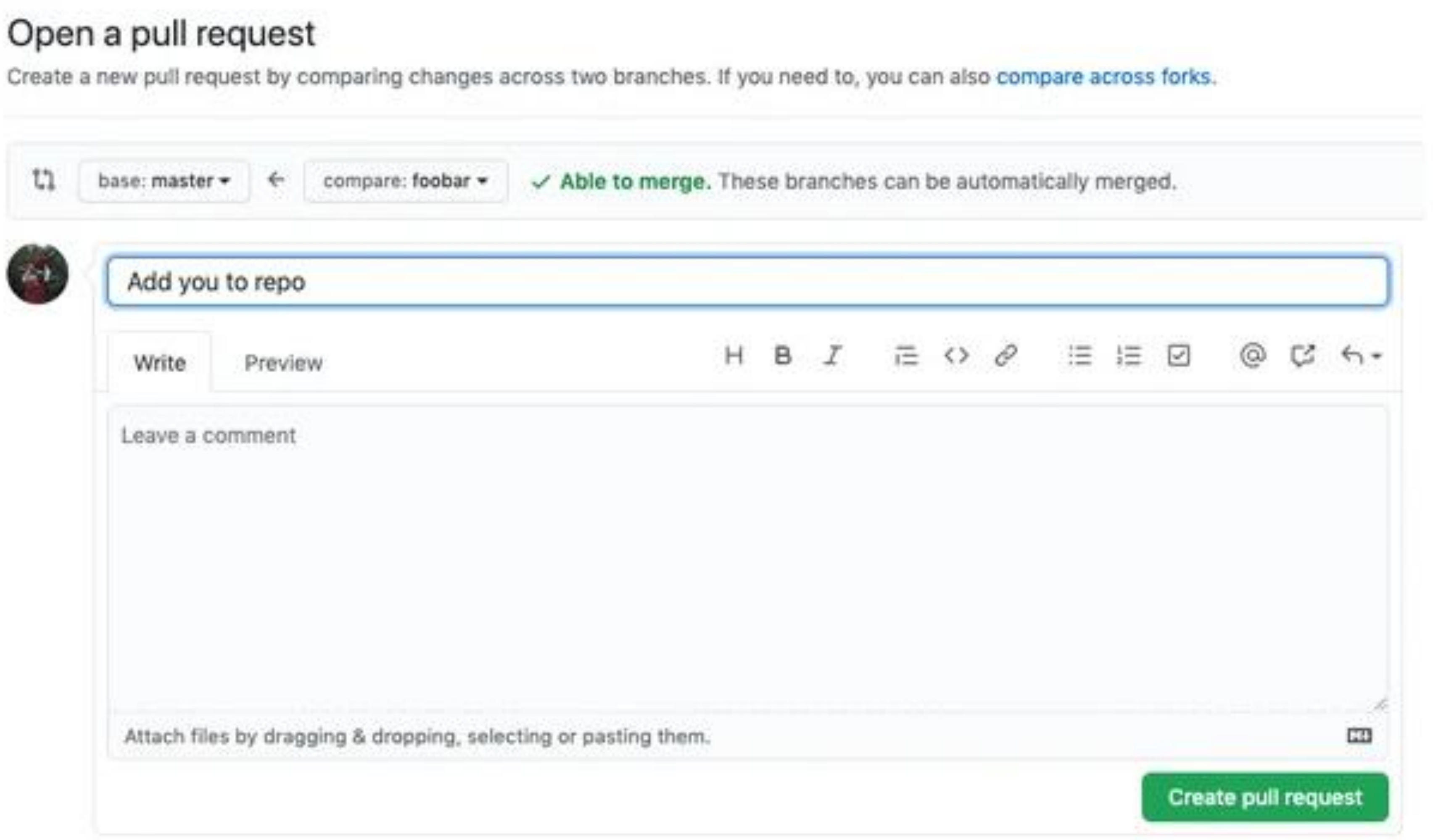
If the commit is already pushed and you want to undo it safely, use:

git revert <1a11e97>. This creates a new commit that reverses the effects of the given commit. Preserves history and is safe for shared branches

Create a pull request (PR)

A pull request (or PR) is a way to alert a repo's owners that you want to make some changes to their code. It allows them to review the code and make sure it looks good before putting your changes on the primary branch.

This is what the PR page looks like before you've submitted it:



Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

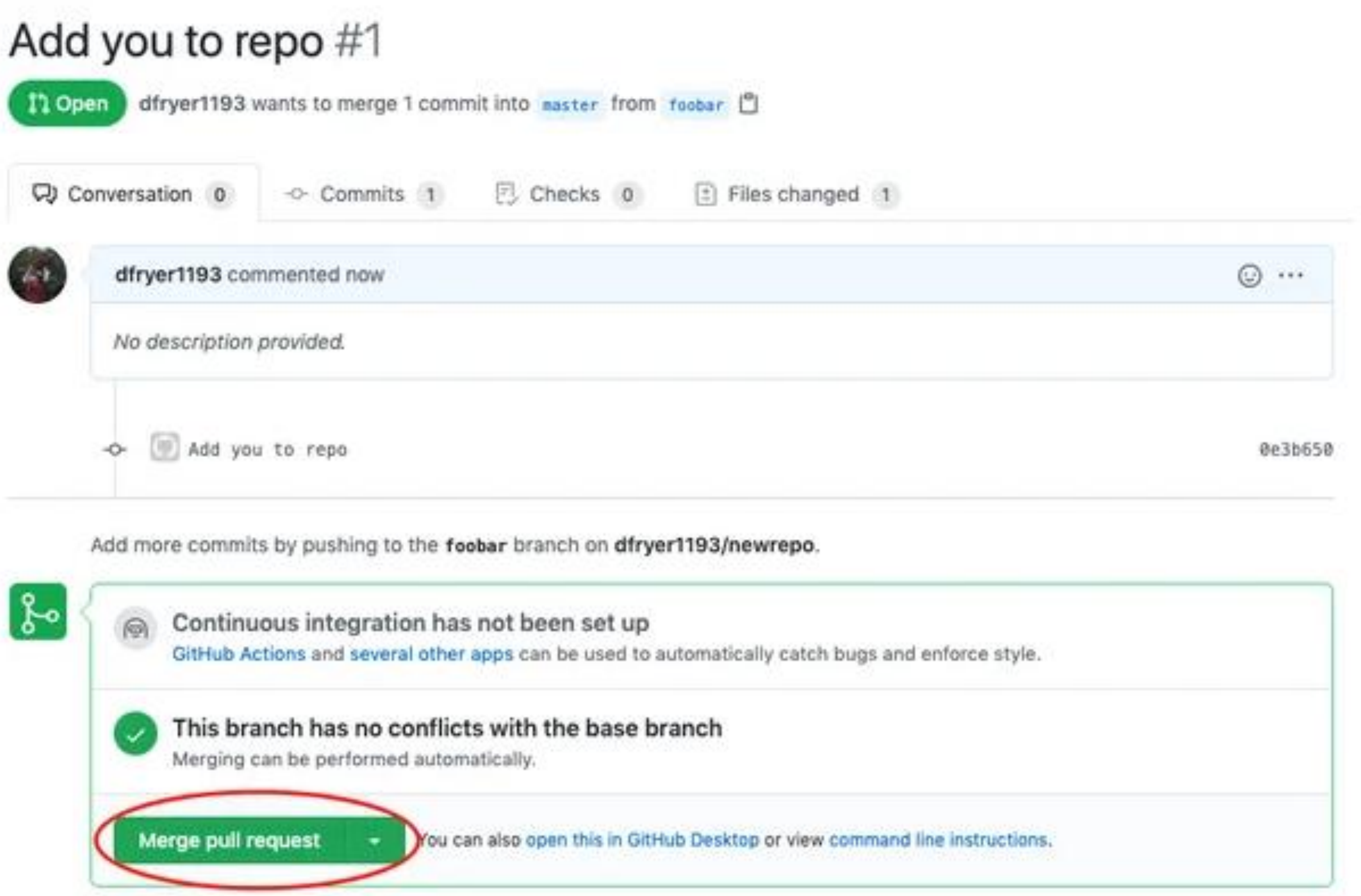
base: master ← compare: foobar ✓ Able to merge. These branches can be automatically merged.

Add you to repo

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

And this is what it looks like once you've submitted the PR request:



Add you to repo #1

Open dfryer1193 wants to merge 1 commit into `master` from `foobar`

Conversation 0 · Commits 1 · Checks 0 · Files changed 1

dfryer1193 commented now

No description provided.

`Add you to repo` `0e3b650`

Add more commits by pushing to the **foobar** branch on **dfryer1193/newrepo**.

Continuous integration has not been set up
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request · You can also open this in GitHub Desktop or view command line instructions.

Get changes on GitHub Locally

Right now, the repo on GitHub looks a little different than what you have on your local machine. For example, the commit you made in your branch and merged into the primary branch doesn't exist in the primary branch on your local machine.

In order to get the most recent changes that you or others have merged on GitHub, use the git pull origin master command (when working on the primary branch). In most cases, this can be shortened to “**git pull**”.

- git pull origin master

QUESTIONS?