# Git/GitHub Introduction

— For open source software sharing and maintenance

Zixuan Liu

# 6 STEPS

01 Git vs GitHub

02 Running GitHub

03 Create a  Repository

04 “add - commit - push”

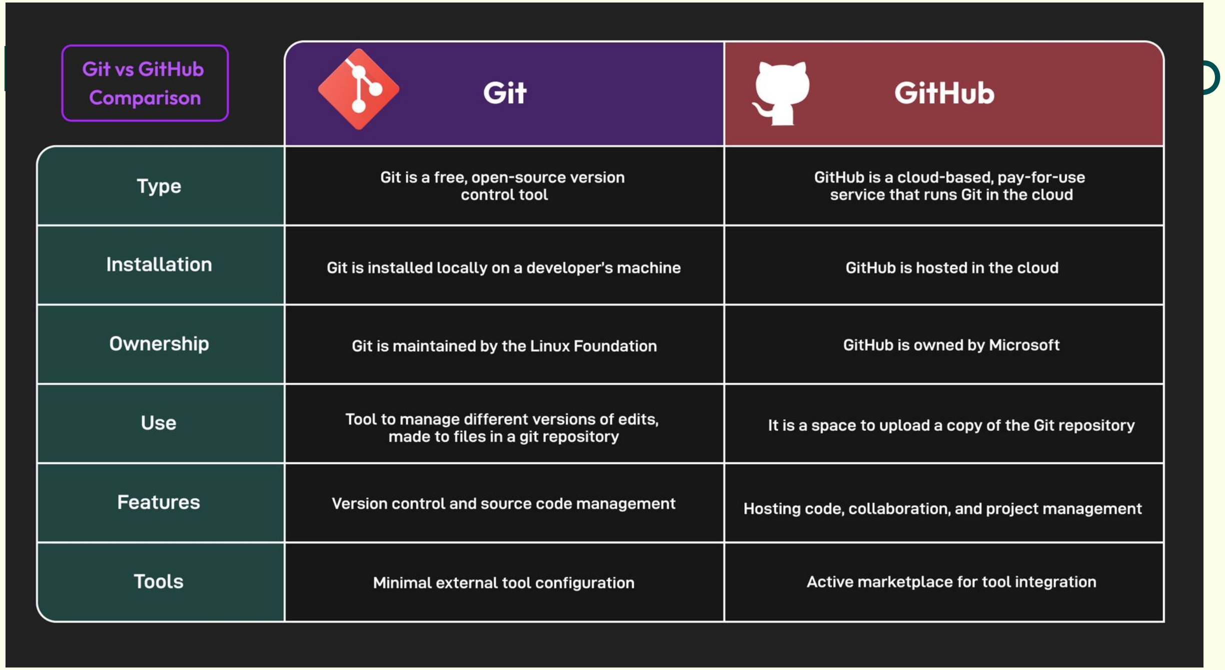
05 Open Sourcing with GitHub

06 Resources

01

## Git & GitHub: Different but related

# Git vs GitHub



| Git vs GitHub Comparison | Git | GitHub |
|---|---|---|
| Type | Git is a free, open-source version control tool | GitHub is a cloud-based, pay-for-use service that runs Git in the cloud |
| Installation | Git is installed locally on a developer's machine | GitHub is hosted in the cloud |
| Ownership | Git is maintained by the Linux Foundation | GitHub is owned by Microsoft |
| Use | Tool to manage different versions of edits, made to files in a git repository | It is a space to upload a copy of the Git repository |
| Features | Version control and source code management | Hosting code, collaboration, and project management |
| Tools | Minimal external tool configuration | Active marketplace for tool integration |

- → Git/GitHub is a software!
- → GitHub is a collaboration tool.
- → GitHub is a social media software developer.
- → GitHub is an ecosystem of version controlled repositories (projects)!

## We will be focusing on GitHub:

## *Web* **and** *Command Line*

## With Git We can:

**The core of Git**

- **Repository(Repo):** Your project - this is where all your files live. Each file in a Git/GitHub repo is version controlled.
- **Version Control:** A system for tracking changes (or versions) of a anything (document, file, image, etc). Version control allows you to go back to previous versions of a file or files.
- **Commit:** A “save” of a version of a file or the repo. Every time you commit a file or files in Git/GitHub, you are leaving a “breadcrumb” of that file or those files to come back to.

## 02 GitHub

## With GitHub We can:

**Features**

- **Sync LOCAL and REMOTE Repo:** With Github your repo can live on your computer and on the web. Which will be a **Folder** on your local machine.
- **Version Control:** the local “commits” will also be synced so everyone sees the history.
- **Open Sourcing and Collaboration :** Because of the version control and sync feature, as well as the social features GitHub offers, it becomes a great place to share your project with others, for open source, peer review, collaboration on development and so on.

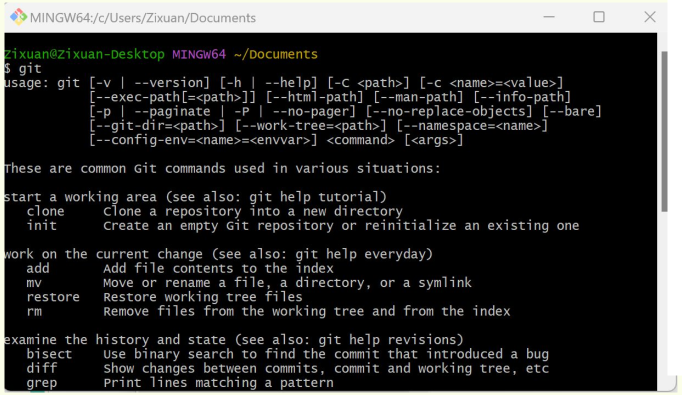
## With GitHub We can:

**Installation**

- **Installation of Git:** https://git-scm.com/ — To use the command lines
- Windows: Download and make sure you click “Add it to Path”, after finish, open git bash terminal.
- Mac OS: Open Terminal and do: `$ brew install git` (The dollar sign is already there) and stay in the terminal.
- **Installation of GitHub Desktop :** https://desktop.github.com/download/ — to use a GUI to interact with GitHub

## With GitHub We can:

### Installation


MINGW64:/c/Users/Zixuan/Documents

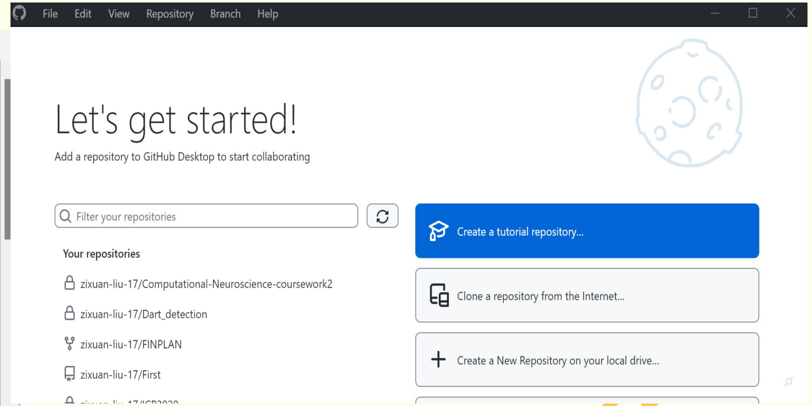
```
Zixuan@Zixuan-Desktop MINGW64 ~/Documents
$ git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--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 reinitialize 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 symlink
   restore   Restore working tree files
   rm        Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect    Use binary search to find the commit that introduced a bug
   diff      Show changes between commits, commit and working tree, etc
   grep      Print lines matching a pattern
```


File
Edit
View
Repository
Branch
Help

Let's get started!

Add a repository to GitHub Desktop to start collaborating

Filter your repositories
Create a tutorial repository...
Clone a repository from the Internet...
Create a New Repository on your local drive...

Your repositories

zixuan-liu-17/Computational-Neuroscience-coursework2
zixuan-liu-17/Dart_detection
zixuan-liu-17/FINPLAN
zixuan-liu-17/First

Git Bash Terminal

GitHub Desktop

## 02 Creating a Repo

## Creating

**Creating a Repo on your local machine:**

```
$ mkdir workspace
$ cd workspace
$ ls -al
$ git init
$ ls -al
```

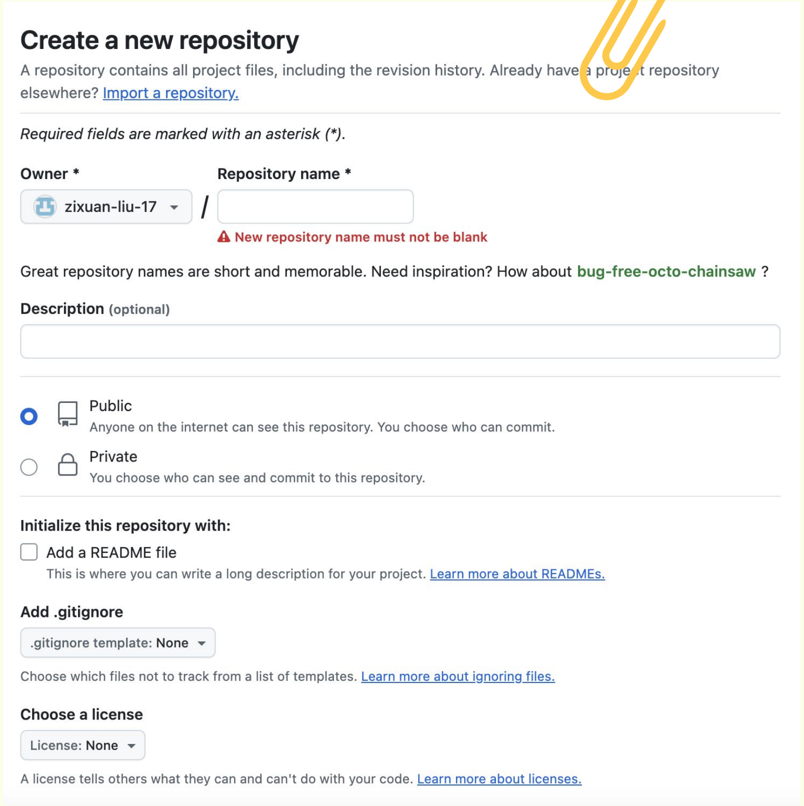
- This create a new folder named as “workspace” under your current working directory
- This on bring you into the new folder, try “pwd” (print working directory)
- ls -al lists all the files in you working directory, including the hidden ones.

- git init: Initialise empty Git repository in ../workspace/.git/

## Creating

### Creating a Repo on GitHub:

- go to github.com
- Sign in
- Click the "+" button



**Create a new repository**

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

*Required fields are marked with an asterisk (*).*

Owner * : zixuan-liu-17 / Repository name * :

New repository name must not be blank

Great repository names are short and memorable. Need inspiration? How about bug-free-octo-chainsaw ?

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:

☐ Add a README file
This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. Learn more about ignoring files.

Choose a license

License: None

A license tells others what they can and can't do with your code. Learn more about licenses.

L

## Demo time

# Connecting Online and Local

**Two ways:**

- **$ git clone :** clone the remote one to local.**Easier**
- You may be asked to enter you github username and password to do so.
  - GitHub now only allow generating personal token separately – go to developer settings, generate one and use it as password
- Don't worry if nothing is shown in the terminal as you key in the password.
- **$git remote add origin REMOTE-URL**
- To verify that you set the remote URL correctly, run the following command.
  git remote -v
- To push the changes in your local repository to GitHub, run the following command.
  $ git push -u origin main
- For this method, do **not** include .gitignore and ReadMe.md in the remote one when creating

L

## Demo time

## 02 “add - commit -push”

## Commits

**Log your changes**

Gif showing the program

The most important ones to us are:

- $ git clone
- $ git pull
- $ git add
- $ git commit
- $ git push

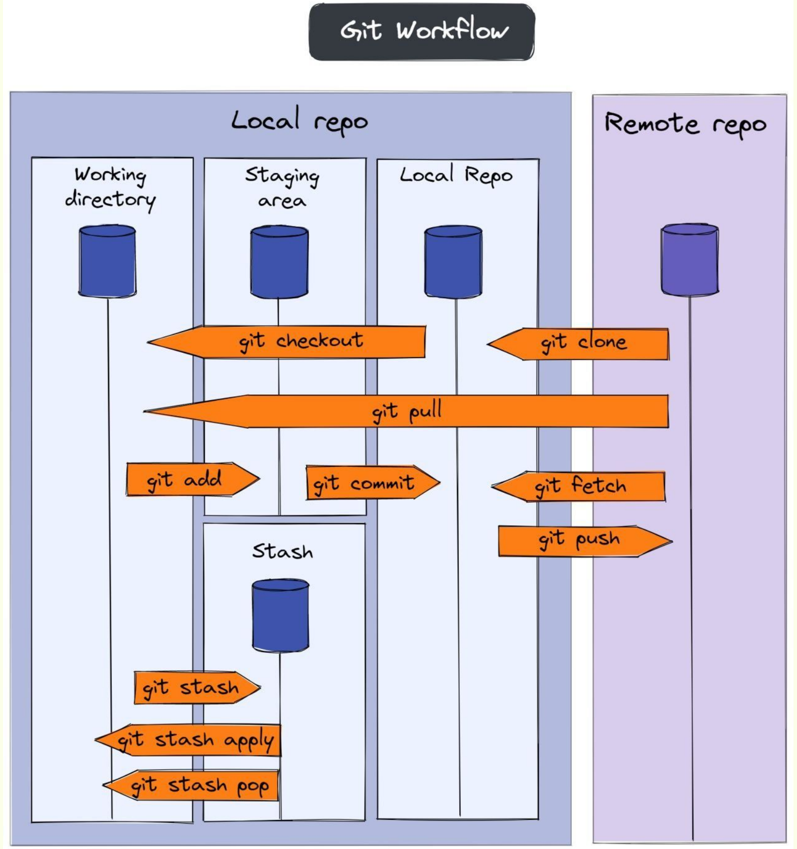


*Image credits: https://x.com/TechWorldWMilan*

## Commits

**Track your changes**

```
$ touch testfile.md
```
(This is equivalent to creating a new file named as testfile.md)

```
$ git status
```
(see the status of the working directory)

```
$ git add testfile.md
```
(adding the change to staging area)

```
$ git status
```
(see what's changed)

```
$ git commit -m "testfile created"
```
(log your changes with a message about it)

```
$ git push
```
(Push it to the remote repo, i.e. GitHub)

## Commits

Log your changes

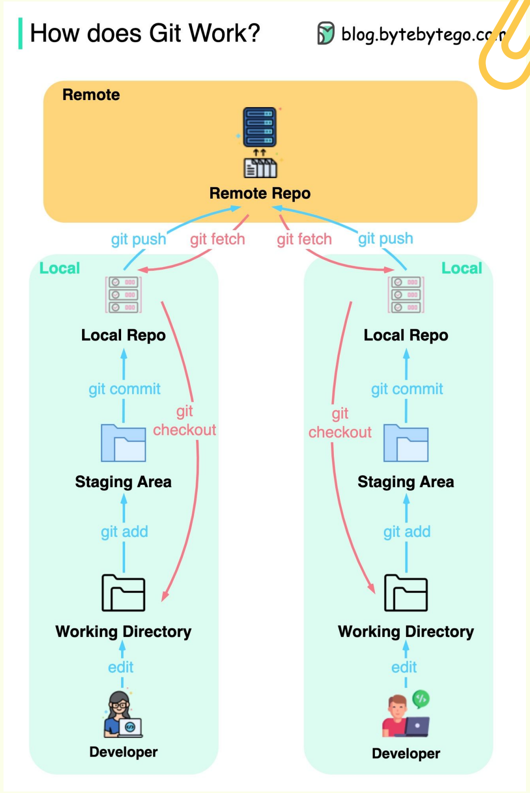
L

## Demo time

## Collaboration

**There can be multiple local repos**

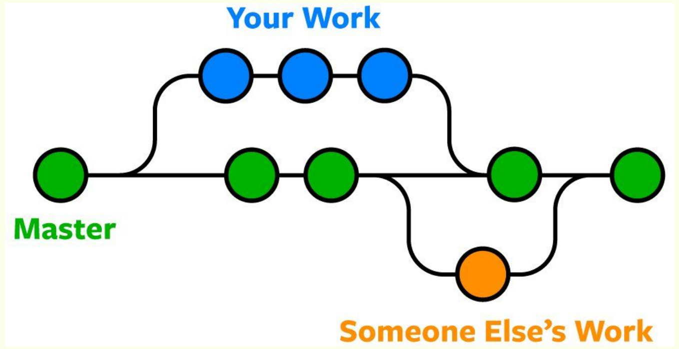
- **Remember to sync regularly :**
- In case of multiple local repos, remember to *$git pull* the remote repo regularly, to avoid too many conflicts.



# Branch and Merge

**Track your changes**

- **Create Branches :** You can create branches to develop features in addition to the main tree, and when you finish, you can bring the features in to the Master branch
- **Developing multiple features by multiple teams:** There can be multiple branches.



Credits: https://www.nobledesktop.com/learn/git/git-branches

L

## Demo time

04

## Open Sourcing with GitHub

# Open Sourcing with GitHub

## Documentation

1. **ReadMe.md** : ReadMe.md is a file in your repo which is shown at the repo's page.
2. **.gitignore:** lists the files that are in your working directory, but are not supposed to be in the repo.

3. **Wiki**: Every repository on GitHub comes equipped with a section for hosting documentation, called a wiki. You can use your repository's wiki to share **long-form** content about your project, such as how to use it, how you designed it, or its core principles. (https://docs.github.com/en/communities/documenting-your-project-with-wikis/about-wikis)

4. **GitHub Pages:** A webpage designed specially for introducing something, p.s. a lot of people use it as their CV.

# Open Sourcing with GitHub

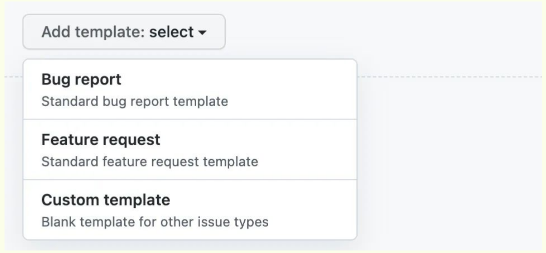
## Discussing Forum

1. **GitHub Issues:** (https://docs.github.com/en/issues/tracking-your-work-with-issues/about-issues)
   Issues can be used to plan and discuss the project.

- Discussing: You can use issue templates to standardise stuff:

  https://docs.github.com/en/communities/using-templates-to-encourage-useful-issues-and-pull-requests/configuring-issue-templates-for-your-repository

- Planning is similar to ClickUp


Add template: select
Bug report
Standard bug report template
Feature request
Standard feature request template
Custom template
Blank template for other issue types

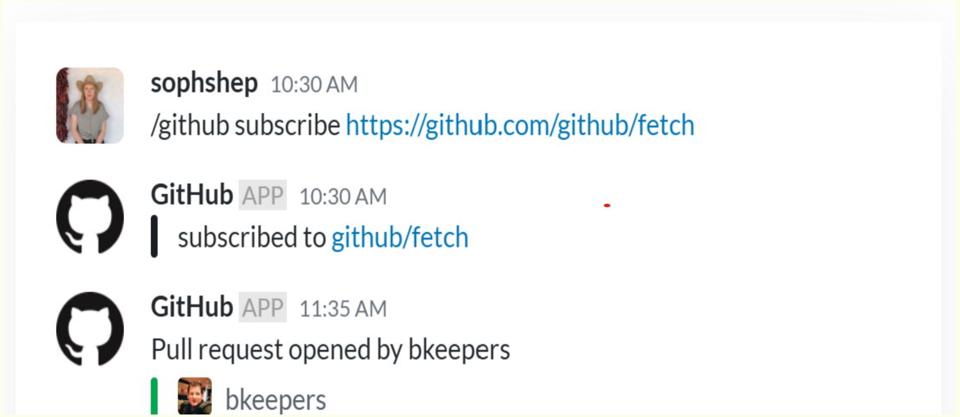
# Open Sourcing with GitHub

## Get GitHub Updates in Slack

1. **GitHub and Slack:** (https://slack.github.com/)

**Stay up to date**

→ Use `/github subscribe [repository name]` in Slack to start receiving updates about activities like:
→ New commits
→ New pull requests
→ New issues
→ Status updates
→ Comments
→ Code reviews



sophshep 10:30 AM
/github subscribe https://github.com/github/fetch

GitHub APP 10:30 AM
subscribed to github/fetch

GitHub APP 11:35 AM
Pull request opened by bkeepers
bkeepers

# THANKS!

Are there any questions?

# RESOURCES

**Docs and Videos:**

- Git Command Cheat Sheet: https://education.github.com/git-cheat-sheet-education.pdf
- Download GitHub Desktop: https://desktop.github.com/download/
- GitHub Markdown course: https://github.com/skills/communicate-using-markdown
- Terminal Cheatsheet: https://www.codecademy.com/learn/learn-the-command-line/modules/learn-the-command-line-navigation/cheatsheet
- Markdown Cheatsheet: https://www.markdownguide.org/cheat-sheet/