

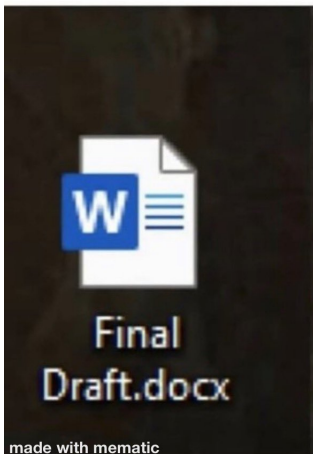
A Brief Introduction to Git and GitHub

Lauren Leek
Website: laurenleek.eu

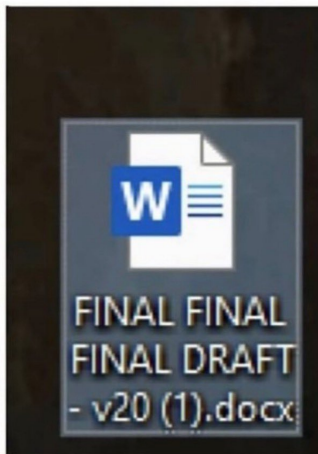
April 19, 2025

Motivation I

How it started



How it's going



Motivation II

**USING
AIRDROP**



**USING
GOOGLE DRIVE**



**USING A
PRIVATE DISCORD**

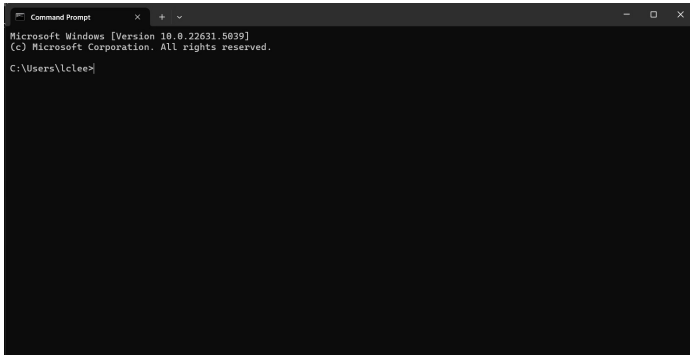


**USING
GITHUB**



Background: the terminal

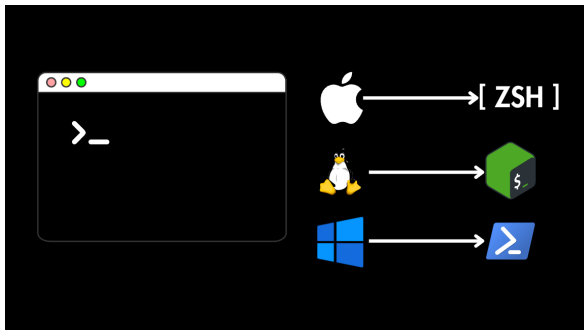
- *To truly master programming, learn how to master the command line first*
- A **terminal**, or **command prompt** or **command line** is a screen or a window that lets you access the Operating System's input and output.
- There are no graphics (images/video) in the terminal, only text

A screenshot of a Windows Command Prompt window. The title bar at the top says "Command Prompt" and has standard window controls (minimize, maximize, close). The main area is black with white text. It shows "Microsoft Windows [Version 10.0.22631.5039]" and "(c) Microsoft Corporation. All rights reserved." followed by the prompt "C:\Users\lclee>".

```
Command Prompt
Microsoft Windows [Version 10.0.22631.5039]
(c) Microsoft Corporation. All rights reserved.
C:\Users\lclee>
```

Background: the shell

- Typically, the terminal runs a program (app) called the **shell**.
- The shell awaits, interprets, processes, executes, and responds to commands typed in by the user.
- Windows has its own thing. For historical reasons, there are two main terminals/shells on Windows these days: CMD & Powershell



How to use the command line

- To use Git we'll be using the terminal or git bash (windows)
- Use "cd" to change your current directory to the destination specified within the command. So to go to the "Users" folder.
- You can navigate directly to the sub-directory.
- If your file path has a space in it, wrap the file path in quotes.



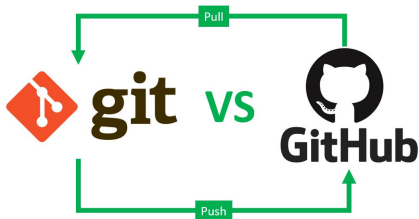
```
MINGW64/c/Users/lclee/OneDrive - Istituto Universitario Europeo/postdoc/co...
lclee@Lauren MINGW64 ~
$ cd "C:\Users\lclee\OneDrive - Istituto Universitario Europeo\postdoc\course_ma
terials\github_intro"

lclee@Lauren MINGW64 ~/OneDrive - Istituto Universitario Europeo/postdoc/course_
materials/github_intro (main)
$ ls
'# How to Use Git & GitHub.md'  git_cmd_guide.md
Images/                         github_config_with_cli.md
VS_code_github_set_up.md

lclee@Lauren MINGW64 ~/OneDrive - Istituto Universitario Europeo/postdoc/course_
materials/github_intro (main)
$ |
```

Theory: What is Git?

- GitHub is a code sharing site for programmers: cloud storage platform & back-up service
- At the centre of GitHub is **Git**, an open source project started by Linux creator Linus Torvalds.
- Git is a Version Control System (VCS)
- GitHub is a Git repository hosting service. A repository (or repo) tracks all changes made to files in your project, building a history over time
- Contribute to existing projects too



Installing Git & GitHub Setup

Installation Options

macOS

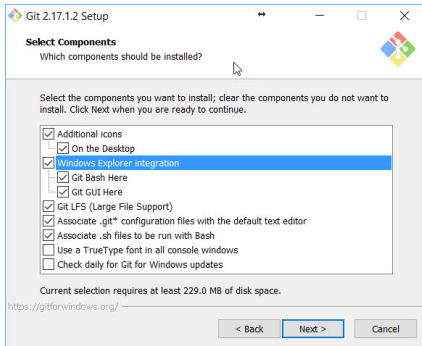
- Open Terminal and type `git` — follow prompt to install developer tools
- Or download:
git-scm.com/download/mac

Windows

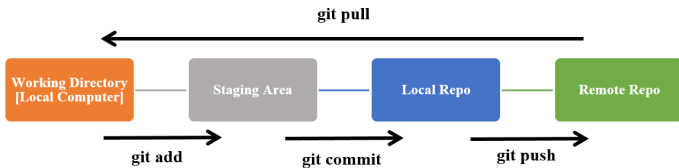
- Download from:
git-scm.com/download/win
- Or install GitHub Desktop:
desktop.github.com (includes Git)

GitHub Account Setup

- Sign up at: github.com
- **Optional:** Apply for student benefits at:
education.github.com/benefits

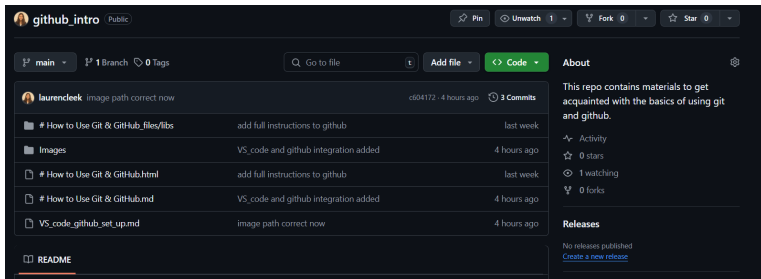


Theory: Git workflow



Practical I: Cloning a Repository

- Navigate to the repository on GitHub
- Click the **"Code"** button
- Copy the HTTPS or SSH URL



```
# Clone repository  
git clone <repository-url>
```

Practical II: cloning a repository - authentication

- You will be prompted for your GitHub username and password
- For password authentication, you need a **personal access token**
- Some users may get a popup window for authentication (skip token)

To create a personal access token:

- 1 Click on your profile icon on GitHub
- 2 Go to: Settings → Developer settings → Personal access tokens
- 3 Click **"Generate new token"**
- 4 Configure it:
 - Name: e.g., "command line"
 - Choose expiration date
 - Select "repo" scope
- 5 Click **"Generate token"** and copy it immediately
 - Enter your GitHub username
 - Paste the token as your password when prompted
 - Once authenticated, Git will finish cloning the repository
 - You won't need to re-authenticate until the token expires

Basic Git Commands

```
# Clone repository
git clone <repository-url>

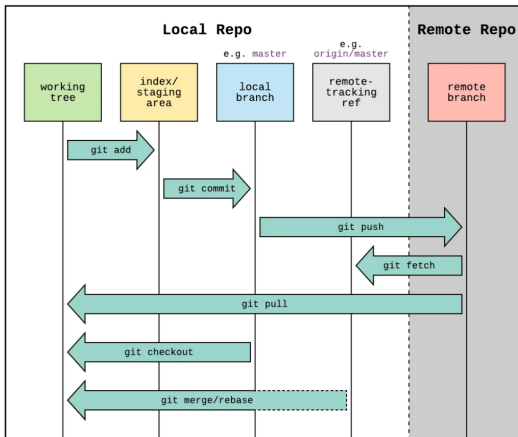
# Initialize new repository
git init

# Stage changes
git add .

# Commit changes
git commit -m "Your commit message"

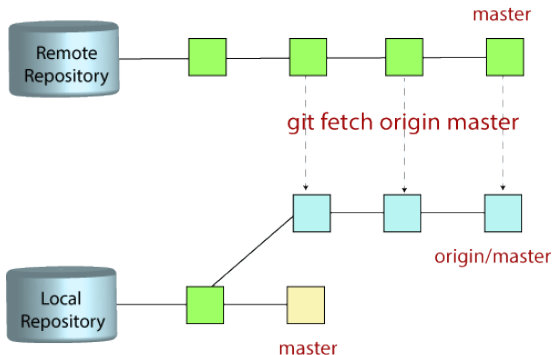
# Push changes
git push
```

Extra commands



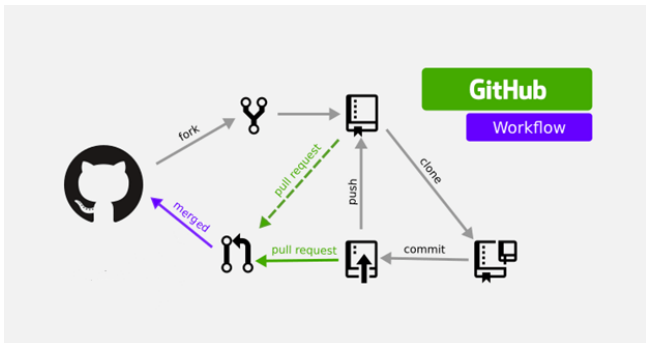
Extra commands: fetching

- Retrieves changes from a remote repository
- Downloads commits, branches, and tags without modifying your local work
- Keeps your local view of the remote repository up to date
- Use in combination with 'git merge' to update your local branch



Extra commands: forking

- Creates your own copy of someone else's repository on GitHub
- Changes you push only affect your fork, not the original repository
- Useful for contributing to open source projects
- Different from cloning which creates a local copy



Extra commands: branching & merging

Branching

- Create parallel versions of your code
- Useful for developing features without affecting the main branch

Create a new branch:

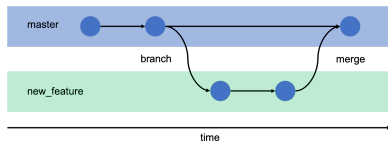
```
git branch feature-name  
git checkout feature-name
```

Or do both in one step:

```
git checkout -b feature-name
```

Merging

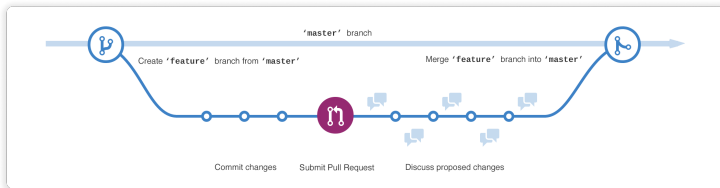
- Combines changes from another branch into your current one
- ```
git merge branch-name
```
- May need to resolve conflicts if the same files were changed





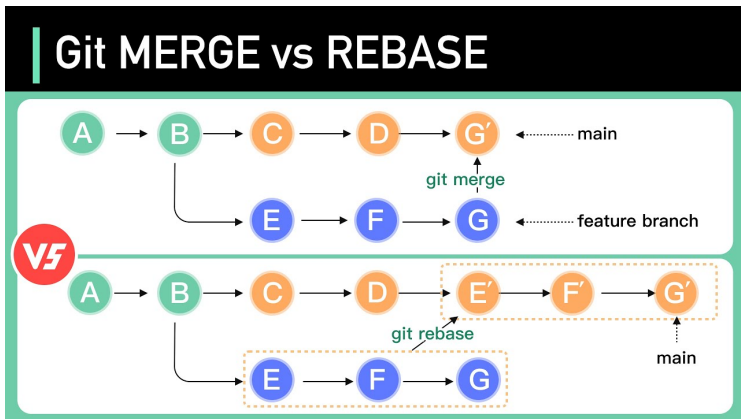
# Extra commands: pull requests

- GitHub feature to propose changes from your fork or branch
- Steps to create:
  - 1 Push changes to your fork/branch
  - 2 Go to original repository on GitHub
  - 3 Click "New Pull Request"
  - 4 Select branches to compare
  - 5 Add description of changes
  - 6 Submit pull request
- Repository maintainers can review, comment, and merge



## Extra commands: rebase

- Reapplies commits from one branch onto another base branch
- Creates a cleaner, linear project history (no merge commits)
- Commonly used to update a feature branch with changes from 'main'
- Be careful: rewriting history can be risky on shared/public branches



# Review of Key Commands

| Command                          | Description                     |
|----------------------------------|---------------------------------|
| <code>git clone ...</code>       | Download repo to local computer |
| <code>git status</code>          | See status of files             |
| <code>git add .</code>           | Stage all changes               |
| <code>git commit -m "..."</code> | Record staged changes           |
| <code>git push</code>            | Upload local changes to remote  |
| <code>git pull</code>            | Update local repo with remote   |

# Questions?

Feel free to reach out:  
`laurencaroline.leek@eui.eu`

`laurenleek.eu`