# Lecture #4: Git SCM

Presented by Jamal Bouajjaj

2023–11–05

For University of New Haven's Fall 2023 CSCIxx51 Course

# Intro

# Git

Git is a Source-Control Management (SCM) system.

Started by Linus Torvalds himself, it is the most widely used version control software.

# How popular?

From `https://openhub.net/repositories/compare`:

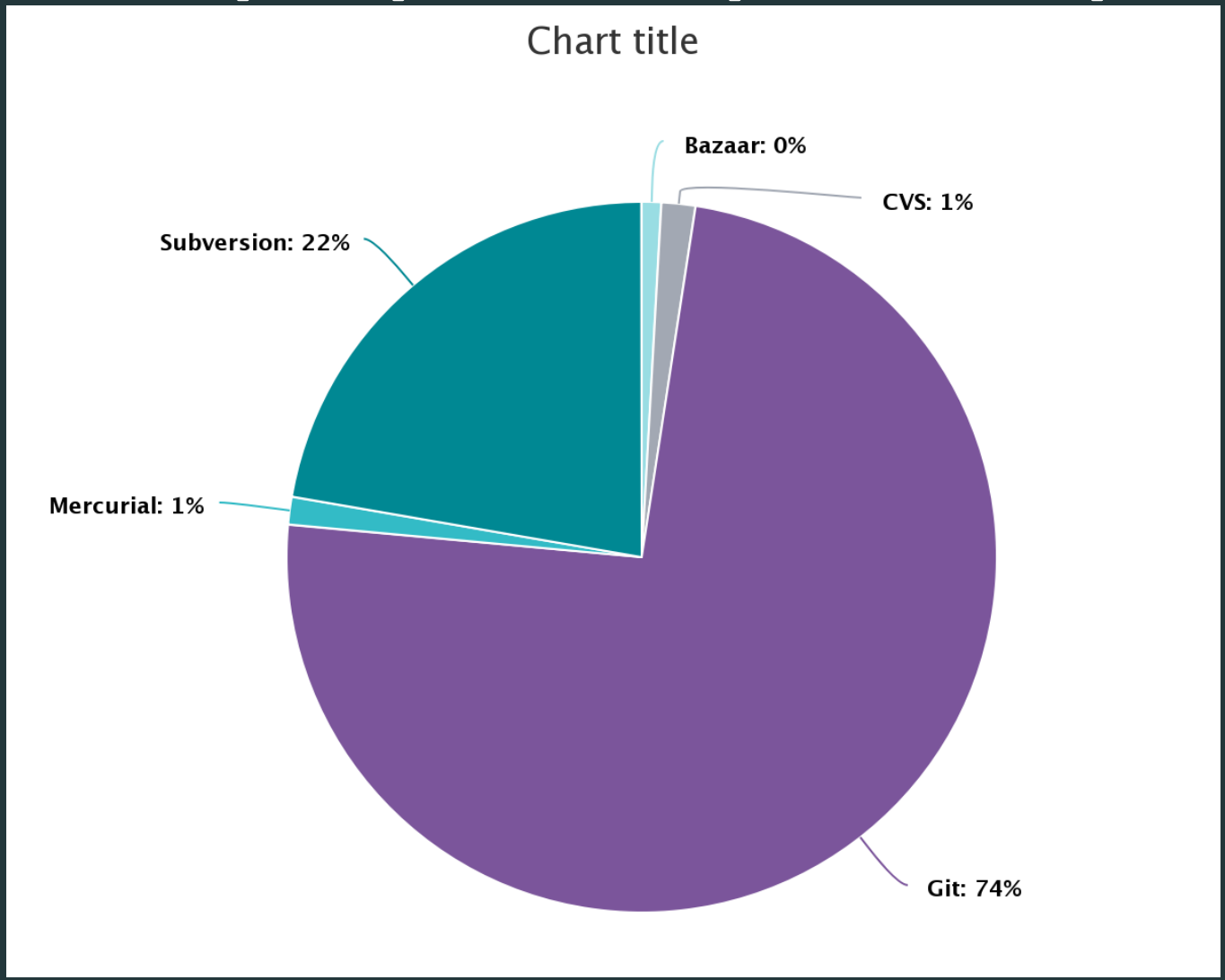



Image Source: `https://openhub.net/repositories/compare`

# So Github?

Q: *So is this just Github?*

# So Github?

Q: *So is this just Github?*

A: **NO**

While Git and Github is related, the former is the version control software, and the latter is a really nice git server.

# Why??

- For tracking changes throughout a project

# Why??

- For tracking changes throughout a project
- For making changes to code with a team or world-wide

# Why??

- For tracking changes throughout a project
- For making changes to code with a team or world-wide
- For blaming bad code (git blame)

# Installation

# Installation

It can be downloaded from `https://git-scm.com/`. The setup process is pretty easy.

# Free Book and Cheat Sheet

If you want a good reference, there is a free book that can be downloaded right from git's webpage. There is also a cheat sheet for quick command references:

`https://git-scm.com/book/en/v2`

`https://training.github.com/downloads/github-git-cheat-sheet.pdf`

# Concepts

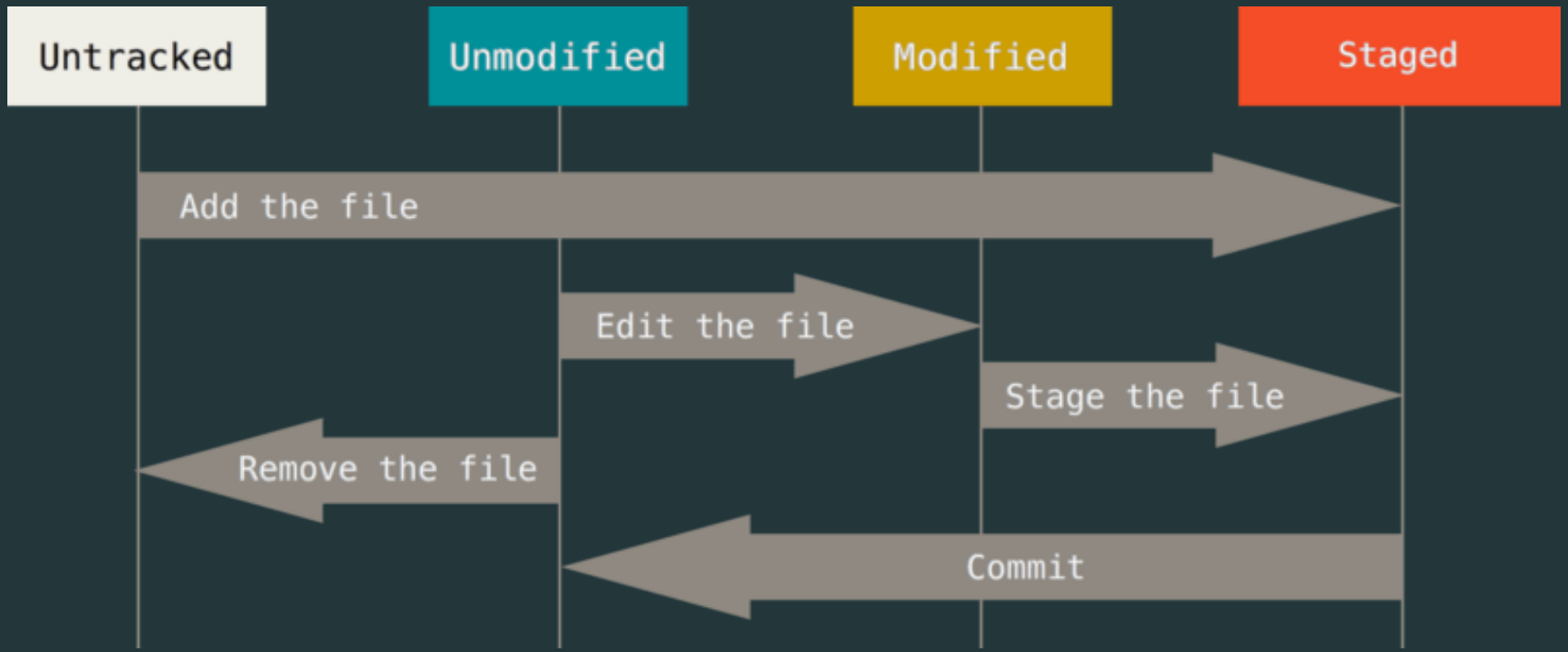
# Decentralized

If you have a git repo, you have it's entire history since it's inception! This makes it decentralized: you are not reliant on some centralized server, rather it's a convenience.

# Some terminology

- **Commit**: A snapshot in time of your tracked files
- **Tracked Files**: Files which git will look at. All others are ignored
- **Branch**: Think of your repo as a tree. A branch is just that, but of code
- **Remote**: The server the changes are pushed unto

# Files Before Commit




Some files can also be ignored with a .gitignore file

Image Source: https://www.git-scm.com/book/en/v2/Git-Basics-Recording-Changes-to-the-Repository

# Github Workflows

# Flows

How you organize your branching and structure depends on your preference. There are two main schools of thought regarding this:

# Git Flow

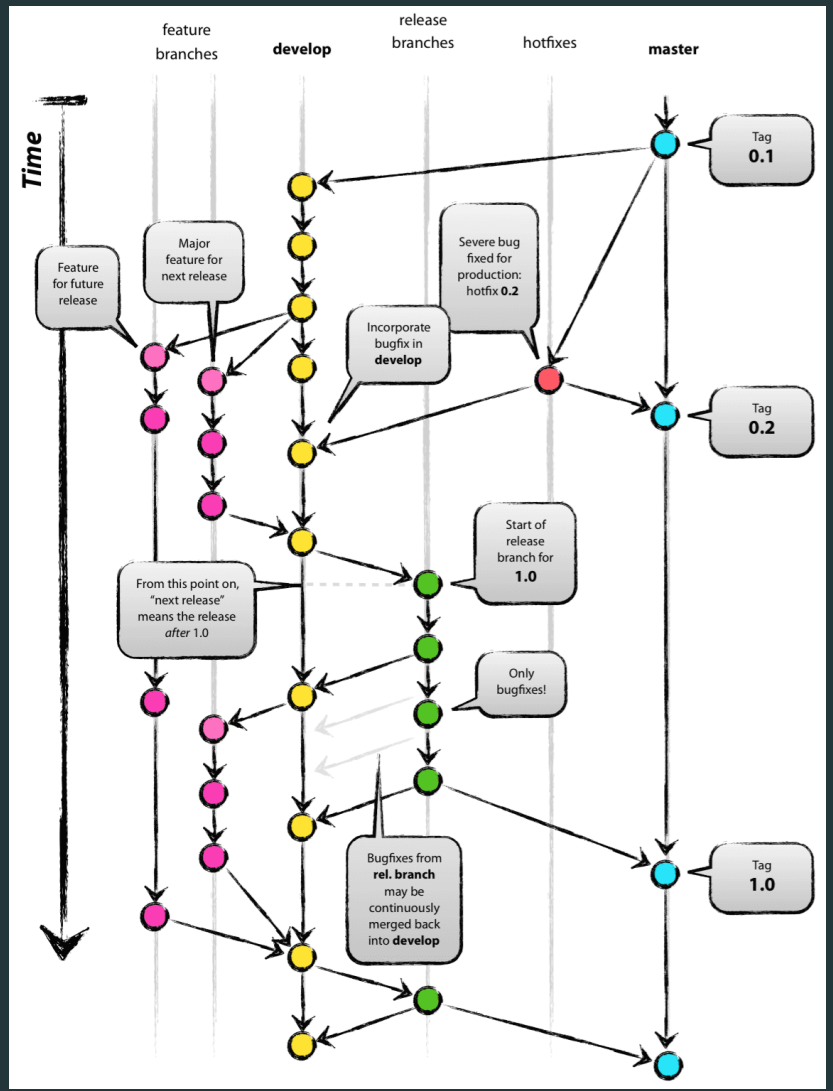



Image Source: https://nvie.com/posts/a-successful-git-branching-model/

# Gitub Flow

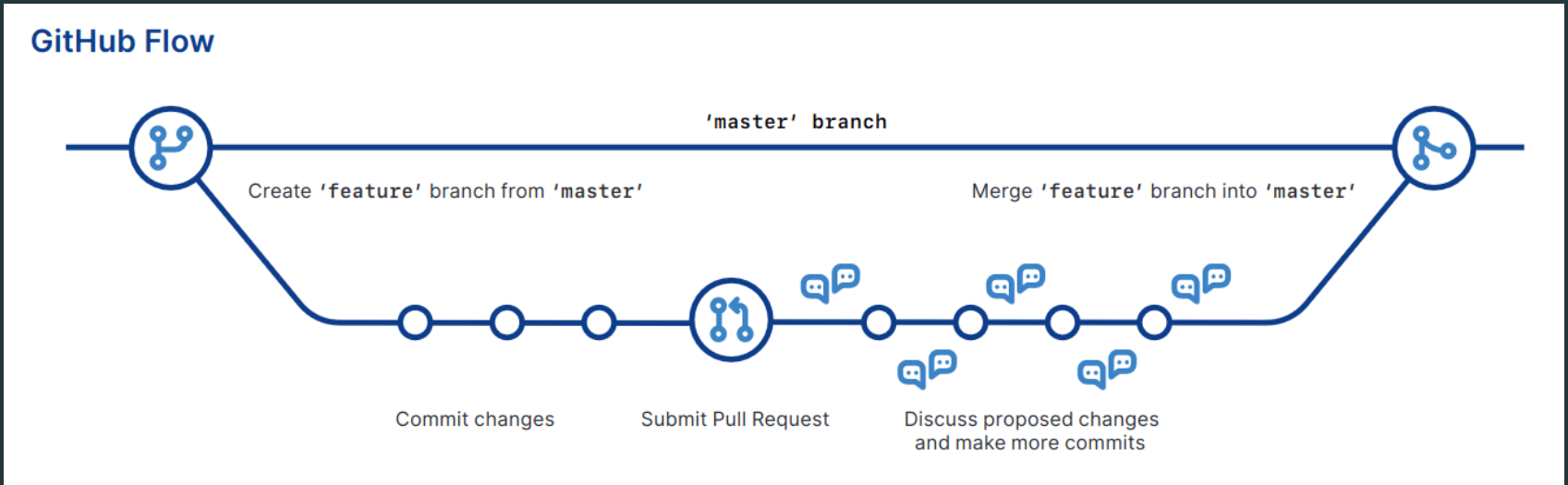



Image Source: https://training.github.com/downloads/github-git-cheat-sheet.pdf

# Commands

See cheat sheet by Github (it's actually nice!)

# PyCharm

PyCharm has git integrated into it, and we'll be demo-ing it in class!

# Good practices

- Commit often
- Add useful commit messages
- If working on a team, it's better to have individual development branches that gets merged and pulled from a master branch.
- Only track source code: all generated stuff can be generated from the code
- Don't track IDE settings unless they are really needed (for example .idea)

# End

**The end**