Intro to Git & GitHub

1. **What is Git?**
2. **Basics of Git**
3. **What is GitHub?**
4. **Basics of GitHub**
5. **Additional Resources**

What is Git?

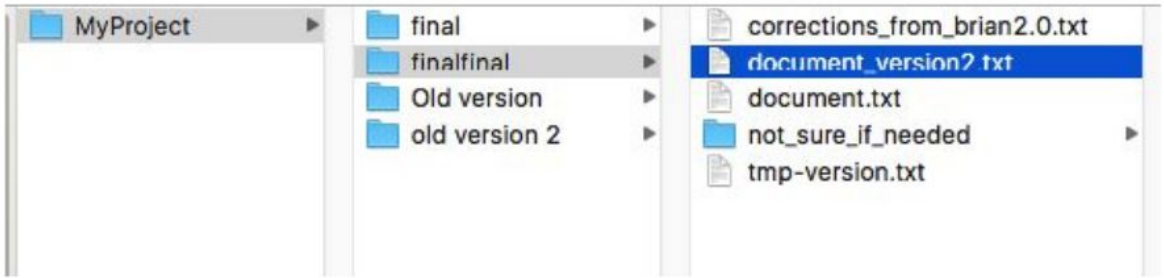
Git is an open-source distributed version control system.

With Git, you can **back up** a version of **a set of files** and later **restore** the files to that version.

It enables multiple users to **edit the same file simultaneously** and later **merge the changes**.

We will be using Git mostly as a way to **interface with GitHub** for the purposes of **autograding.**

Why Version Control?



Version control

Tracks & logs changes in your files with…

- Author
- Timestamp
- Description

Allows…

- Restoring old versions
- Having multiple parallel versions
- Analyzing your code

[slide by Lukas Kalbertodt]

Basics of Git

1. **Create a new Git repository:**

   git init <NEW-REPO-NAME> OR **git clone <EXISTING-REPO-URL>**

2. **After you changed a file in the repo, stage the changes you made:**

   git add <FILE-NAME>

3. **Commit the changes you made:**

   git commit -m “Message detailing the changes I made”

Basics of Git

4. View the history of all commits in this repo:

   git log

5. View the current status of all files in this repo:

   git status

6. After you changed a file mistakenly, reset it to the latest committed version:

   git checkout <FILE-NAME>

What is GitHub?

GitHub is a service that hosts Git repositories online.

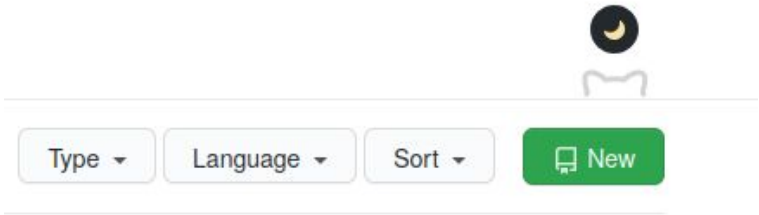
This makes it possible to **back up your local files to an online storage**.

It also makes **collaboration with others** on a project easy.

In our case, the tools GitHub provides can be used to **automatically grade homeworks**.

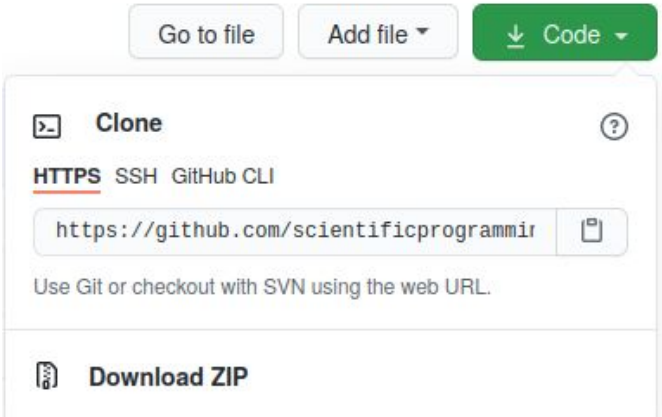
Basics of GitHub

1. **Create a new GitHub repository:**



2. **Clone the online GitHub repository to your local system with Git:**

git clone <REPO-URL>



Basics of GitHub

3. **After you made some changes and committed them with Git, push them to GitHub:**
   git push

4. **If someone else made changes and pushed them to GitHub, pull them with Git:**
   git pull

Best Practice: If you collaborate with others, **always pull before making changes** in order to avoid merge conflicts (two users making contradictory changes)

Additional Resources

Git & GitHub have much more to offer.

Additional features include **different branches**, **pull requests** and **continuous integration**.

Many tutorials for Git & GitHub can be found online, since they are **popular among developers**.

The Git Handbook on **guides.github.com** is a good start for more information.