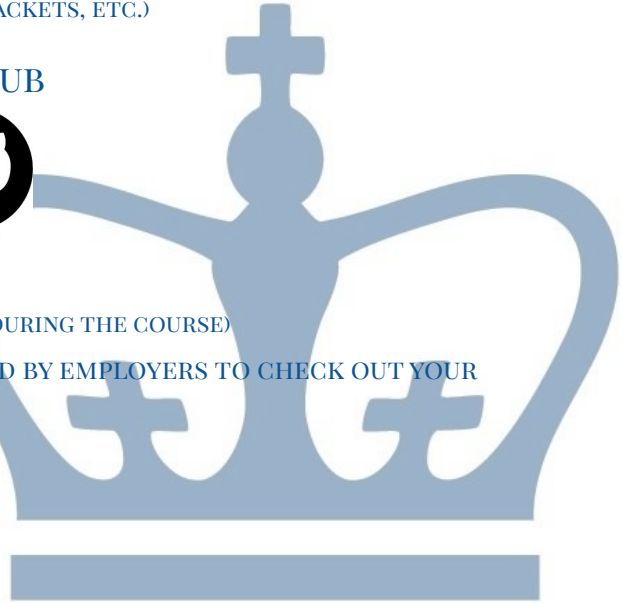
Git vs GitLab vs GitHub

Git - distributed version control system: a software that provides versioning functionality
It's not the only one in the market - there are Subversion, Mercurial, etc.

Git is invisible but we can communicate with it using different tools:

- Built-in consoles: Terminal (for MacOS) or CMD (for Windows)
- Desktop GUI: Git Bash, SourceTree, GitKraken, etc.
- Remote repositories with UI: GitHub, GitLab, etc.
- Built-in plugins in IDE (integrated development environment: WebStorm, VS Code, Brackets, etc.)

GitLab  or GitHub 



They are roughly the same

(they have some differences but we don't use those features during the course)

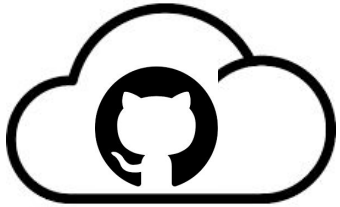
We recommend using GitHub for your projects since it is widely requested by employers to check out your coding style

For the class purposes:

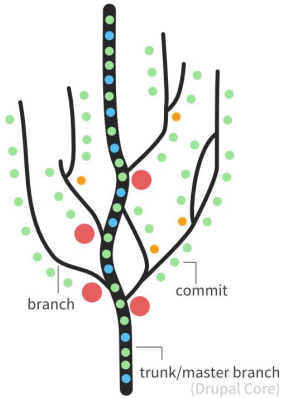
 Pull new activities from GitLab

 Push your own work to GitHub

GitHub or GitLab store remote copy of your project including its git history

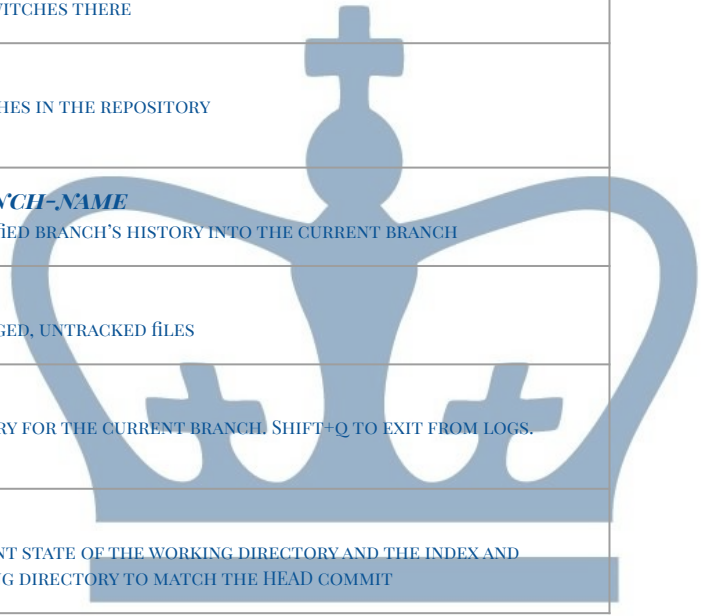


Project's git tree



GIT COMMANDS

GIT INIT Turns a directory into a git repository; it creates a hidden folder .git where it keeps tracking	**GIT BRANCH *BRANCH-NAME*** Creates the new empty branch with the name of your choice
GIT CLONE [URL] Clones a repository that already exists located at the URL	**GIT CHECKOUT *BRANCH-NAME*** Switches to the specified branch
GIT ADD . (SYNONYM: GIT -A) Adds changes to the index - the view of the working directory that is ready for commit	**GIT CHECKOUT -B *BRANCH-NAME*** Copies the current working branch, creates a new branch with the name of your choice, and switches there
GIT COMMIT -M "YOUR DESCRIPTIVE MESSAGE" Records changed files in version history. The message should be descriptive so once you see the message you get an idea about what it was made for	**GIT BRANCH** Lists all the branches in the repository
GIT PUSH Uploads commit to the remote repository	**GIT MERGE *BRANCH-NAME*** Combines the specified branch's history into the current branch
GIT PUSH ORIGIN *BRANCH-NAME* Uploads commit to the branch of your choice (master by default)	**GIT STATUS** Lists staged, unstaged, untracked files
GIT PUSH -U (SYNONYM: GIT PUSH --SET-UPSTREAM) Sets the connection between the local branch and remote one so you don't have to write the branch name every time when you push	**GIT LOG** Lists version history for the current branch. Shift+Q to exit from logs.
GIT PULL Updates the current local working branch with all new commits from the corresponding remote branch	**GIT STASH** Records the current state of the working directory and the index and reverts the working directory to match the HEAD commit



You can find more commands here: https://git-scm.com/docs

Irina Plaksina for Columbia University. 2020

Scenarios

To turn the directory to a repository

1. Make sure you're inside the directory
2. **git init**
3. .git folder is added into the directory
   If it's hidden - make hidden files to be shown

   For Windows For Mac

To clone the repository

1. Make sure you're inside the directory you want to clone the repository to
2. **git clone *link-to-the-remote-repository***
   Ex. `git clone git@github.com:account/repo.git`

For class purposes:

3. **git checkout -b**
   branch-name-of-your-choice
   Ex. `git checkout -b my-branch`

To get repository updates

1. Make sure you're inside the repository
2. **git pull**

NB! You pull from the remote branch with the same name as your local branch

To merge updates to another branch

1. Make sure you're inside the branch to merge to
2. **git merge *branch-name-your-merge-from***
   Ex. `git merge master`

To switch between branches

1. **git checkout *branch-name***
   Ex. `git checkout my-branch`

To see the branches list

1. **git branch**

To add your changes to remote repo

1. **git add .** or **git add -A**
2. **git commit -m "*your descriptive message*"**
   Ex. `git commit -m "added submit button to the form"`

NB! If you push from master:

3. **git push**

NB! If you push not from master:

3. **git push -u origin *branch-name*** or
   git push --set-upstream *branch-name*
   Ex. `git push -u origin my-branch`

If you work together on the project

1. **git add .** or **git add -A**
2. **git commit -m "*your descriptive message*"**
3. **git checkout master**
4. **git pull**
5. **git checkout *your-branch***
6. **git merge master**
7. **git add .** or **git add -A**
8. **git commit -m "*your descriptive message*"**
9. **git push**

You can find more commands here: https://git-scm.com/docs

Irina Plaksina for Columbia University. 2020

Oopses

You worked in a master branch instead of your own and you need to push

1. **git checkout -b *branch-name-of your-choice***
2. **git add .** or **git add -A**
3. **git commit -m "*your descriptive message*"**
4. **git push**

You worked in a master branch of your class repo and you need to pull

1. **git add .**
2. **git stash**
3. **git pull**
4. **git checkout *your-working-branch-name***
5. **git merge master**

If you want to keep your changes:

6. **git stash pop**

You added wrong changes

1. **git reset**

You added and committed wrong changes

1. **git reset HEAD~1**

NB! Will take you back to the one commit but keep changes you made before unstaged

OR

1. **git reset --hard HEAD~1**

NB! Will take you back to the one commit and remove changes you made

You can find more commands here: https://git-scm.com/docs

Irina Plaksina for Columbia University. 2020