EPFL

EMBEDDED SYSTEMS LABORATORY

Introduction to git part 1

EPFL - Embedded Systems Laboratory

2025

These videos

- nothing → work with confidence
- 4 chapters
- 4 steps:
  - Theory
  - I do a demo
  - Recap and tips
  - You re-do the demo
- There will be a cheatsheet

Introduction to git

introduction and basic use

Working alone

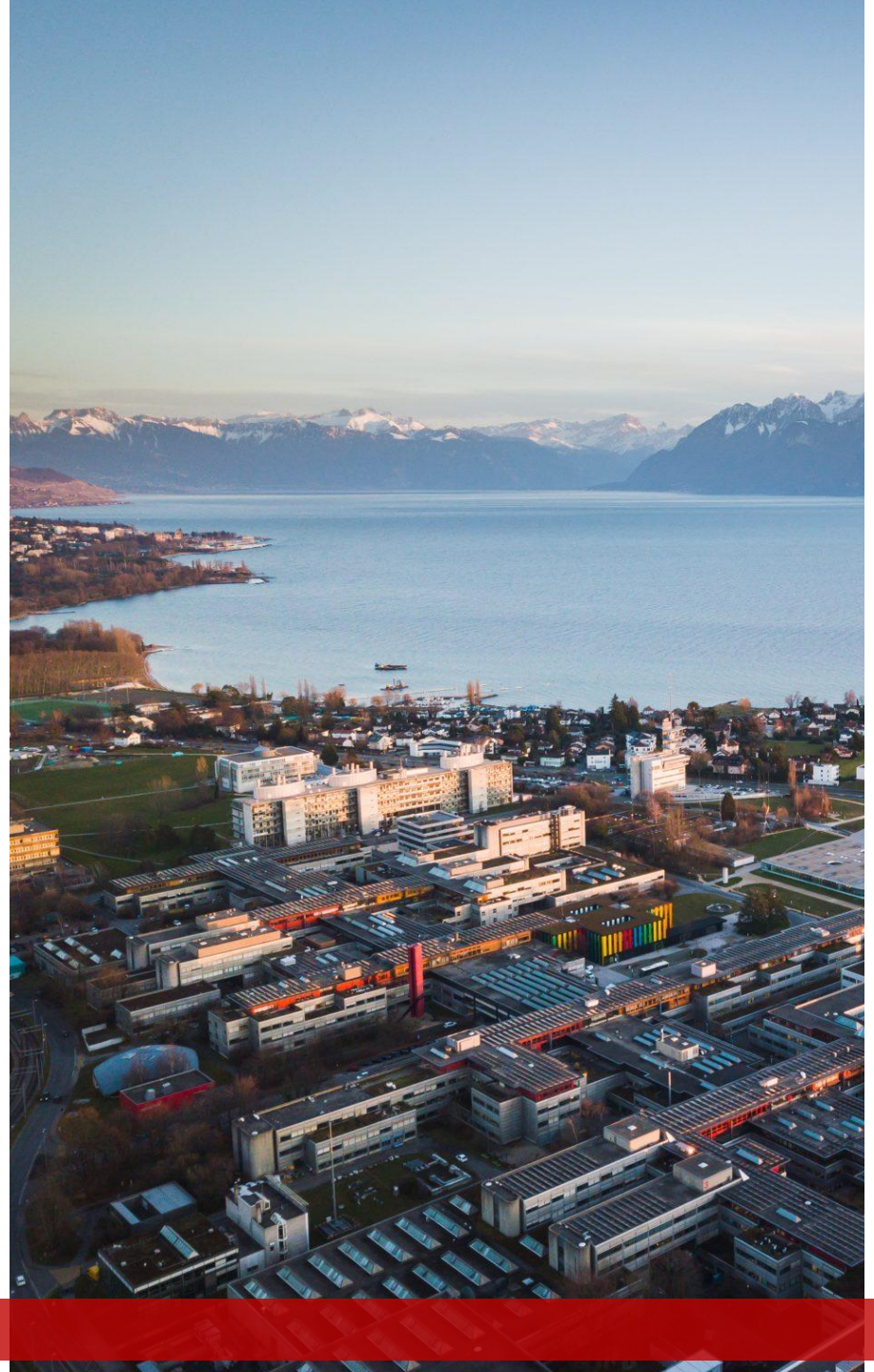
for any project you have

Using other’s work

e.g. the practical works

Working with others

e.g. your final project



What is it?

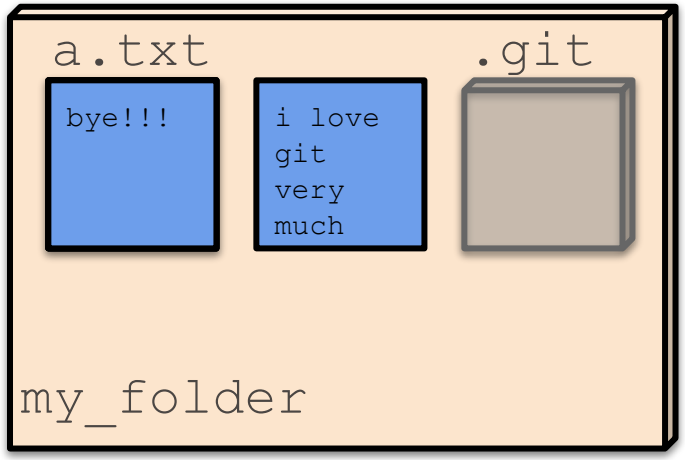
- Linus Torvalds (inventor of Linux)
  working for the Linux Kernel Project (massive project)
- Decided to stop using their Version Control System
- In a **few weeks** created his own: **git -** it's just that simple!
- Easy, reliable, free and open-source version control

- Keep track of the **history** of a project
- Manage different **versions**
- Allow a team to **cooperate**

without having to send code-final-version-2-fix-bug-xjdha.zip over Whatsapp

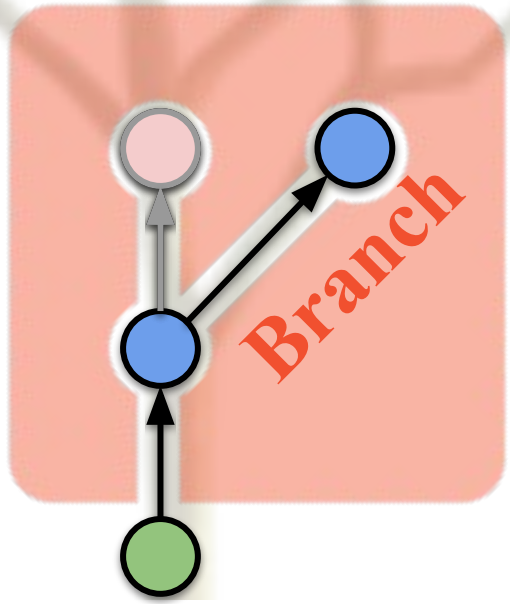
How does it work?

Repository (repo)




```
- Removed b.txt
Modified b.txt
 + very much
Modified a.txt
 + !!!
Modified a.txt
 - hello
 + bye
+ Added a.txt
+ Added b.txt
```

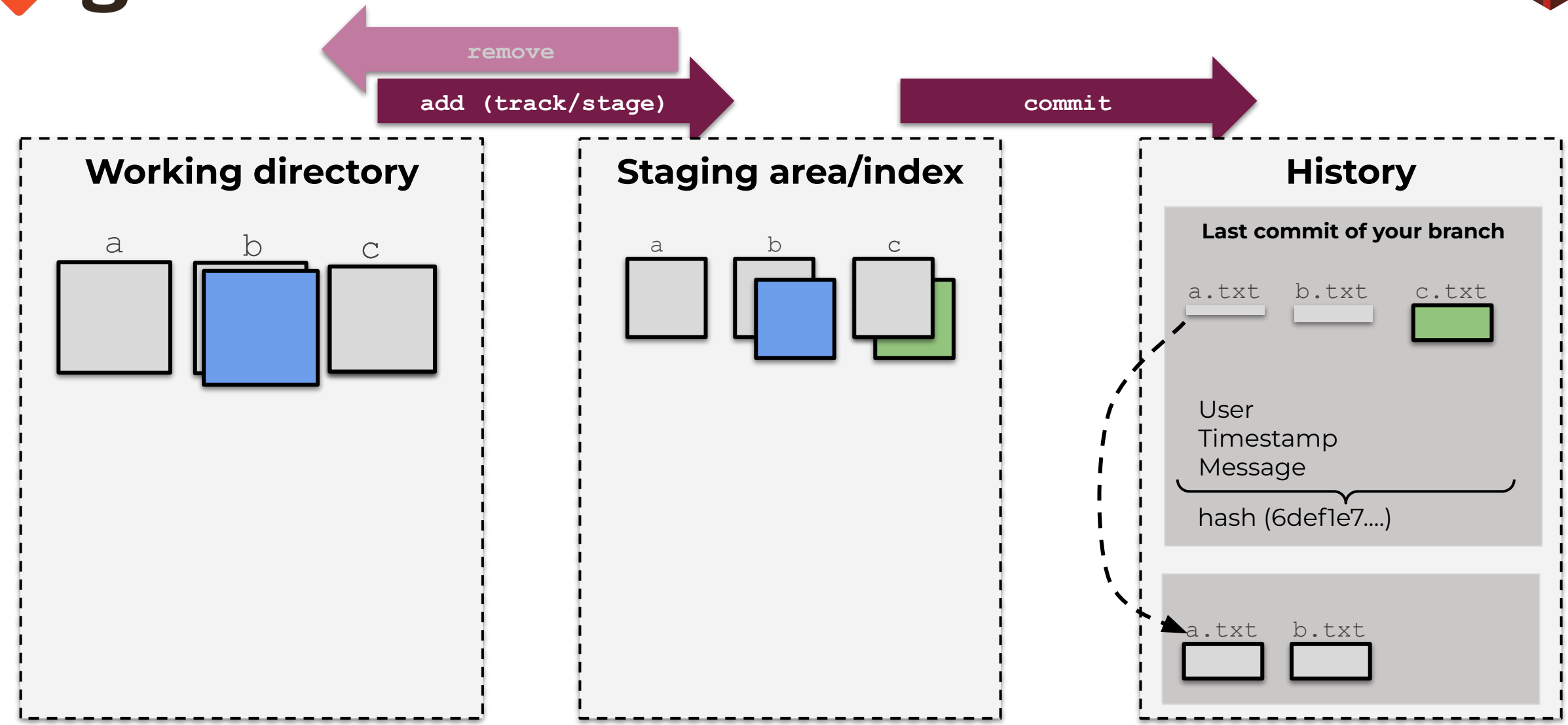
History (log)



Commit

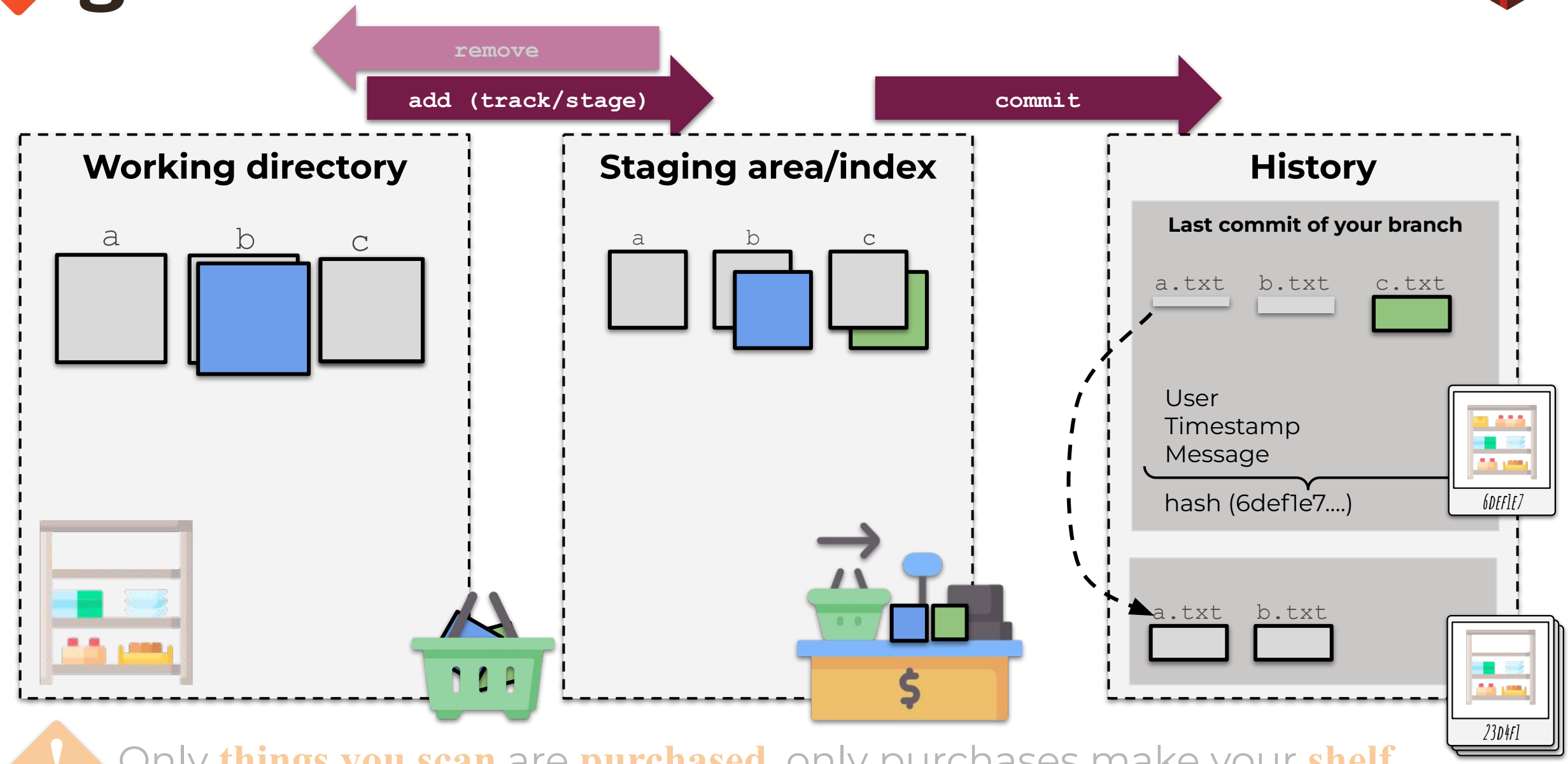
- Git keeps the **history** of **commits** done on a **repo**
- Allows you to have different **branches** of your work

Under the hood



Only **staged** changes are **committed**, only commits make your **history**

Under the hood



Only **things you scan** are **purchased**, only purchases make your **shelf**

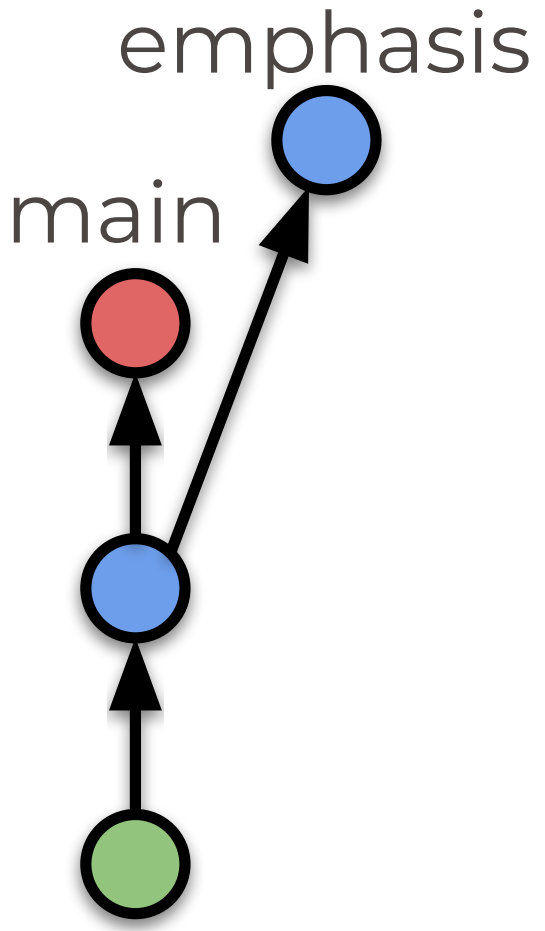
Demo

Set-up

- Install git
- Config credentials

First steps

- Create a folder and turn it into a repository
- Make changes and commits
- See the history
- Create branches



Demo - recap

- `git init`
  - Turns a folder into a repository (adds a `.git` folder)
- `git config --global credential.helper store`
  - Stores your credentials in **`~/.git-credentials`** IN PLAIN TEXT
  - ⚠️Do not enter your passwords!
- `git config --global user.name "Juan Sapriza"`
  `git config --global user.email juan.sapriza@epfl.ch`
  - Save your username and password (globally, can also use **`--local`** for only this project)

- `git status`
  - See the status of the repository (files with changes)

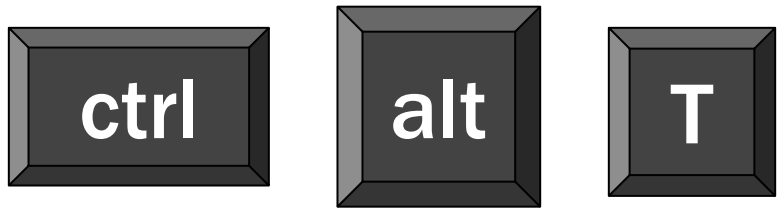
Demo - recap

- `git diff`
  - See the **unstaged** changes w.r.t. the last commit
- `git restore <file>`
  - Removes changes from working directory, back to last commit
- `git add <file>`
  - Starts **tracking** `<file>`, and **stage** its changes
- `git restore --staged <file>`
  - Removes changes from the staging area, kept in working directory
- `git rm --cached <file>`
  - Removes <file> from the index, but keeps it in the working directory
- `git commit -m "Added a test file"`
  - Create a new commit with all the staged files with a **m**essage

Demo - recap

- `git log [--all --oneline --graph]`
  - See the history, your name, commit and it's **hash**
- `git checkout [-b <branch name> [<hash>]]`
  - Create a new **b**ranch called `<branch_name>` from the commit `<hash>`
- `git branch`
  - See a list of branches
- `git branch -m main`
  - Set the na**m**e of the current branch to `main` from the default `master`

Demo - recap



- `cd <somewhere>`
  - **C**hange **D**irectory
  - `~` is my home
  - `..` is one directory back
  - `../..` two directories back ...
  - `-` is the last directory
  - `.` here
- `ll`
  - **L**ist files and folders, in a **L**ong list
- `mkdir`
  - **M**a**k**e a new **Dir**ectory
- `code <folder/file>`
  - Open the selected folder/file with VSCode

Demo - cheatsheet

```
git --version                                   # Check if git is installed
sudo apt install git                            # Install git
git config --global credential.helper store     # Store your credentials in ~/.git-credentials
git config --global user.name Juan              # Introduce yourself to git :)
git config --global user.email my@email.ch
cd ~/Desktop                                    # Change Directory to the Desktop
mkdir my_folder                                 # Make a new Directory called my_folder
l                                               # List the contents of the Desktop
cd my_folder
git init                                        # Turn my folder into a repo
l -a                                            # To see hidden folders
l -a .git
code .                                          # Open VScode in this folder
        # Create a.txt and b.txt
git status                                      # Show the status of my repo
git add a.txt                                   # Start tracking a.txt
git status
git add .                                       # To add ALL files (be careful!)
git rm --cached b.txt                           # To remove from the stage area
git commit -m "Added files to have something"   # Commit changes
git status
git log                                         # See the history of my branch
git branch -m main                              # Rename the current branch
        # Modify a.txt "hello" -> "bye"
git status
git diff                                        # Show the differences between UNSTAGED
                                                # changes and the last commit
git restore a.txt                               # Restore changes to the last commit
        # Modify a.txt "hello" -> "bye"
git add a.txt
git diff                                        # Shouldn't show anything, a.txt is staged!
git restore --staged a.txt                      # Remove the changes from the staging area
                                                # (file is still tracked, ≠ rm --cached)
git add .
git commit -m "Changed to bye because i'm leaving"
git log
        # Remove b
git status
git restore b.txt
        # Remove b
git add .
git commit -m "b is no longer necessary because it's clear"
git log
git checkout --help                             # See the help about a command
git checkout -b emphasis <hash>                 # Create a new branch from <sha>
git log
        # Make modifications in a and b
git add .
git commit -m "Added emphasis, because it was not clear enough"
git log
git log --all                                   # See also other branches (in chronological order)
git branch                                      # See list of branches
git checkout main                               # Switch to branch "main"
git log --all --oneline --graph                 # See the history with branches
```

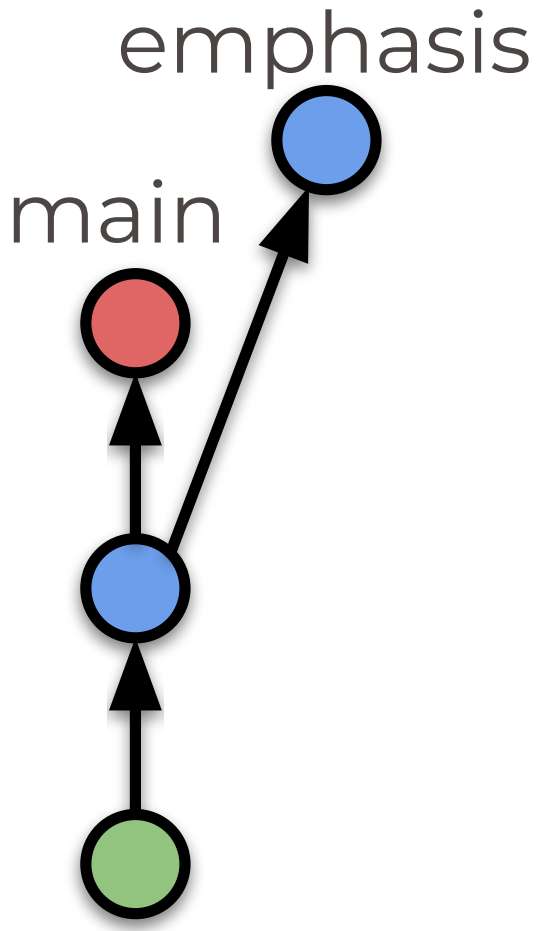
Demo - try it yourself

Set-up

- Install git
- Config credentials

First steps

- Create a folder and turn it into a repository
- Make changes and commits
- See the history
- Create branches



Tips

- Repeat `git status` often to get a grasp of the effect of your changes
- Use commit messages that describe **WHY** you did changes
  - ✓ `Added _____ feature to improve _____`
  - ✓ `Fixed _____ bug by doing _____`
  - ✗ `wip`
  - ✗ `changes`
- Press **TAB**⇥ to autocomplete files and folders when using the terminal
- Check `git help`: it's pretty well documented!
  - ✓ `git help <command e.g. log, checkout>`
  - ✓ `man git`
  - ✓ `git help tutorial` = git-scm.com/docs/gittutorial

Thank you!

EPFL - Embedded Systems Laboratory