Git 101

Jonas Jakobsen

AALBORG UNIVERSITY
DENMARK

AAU SATLAB

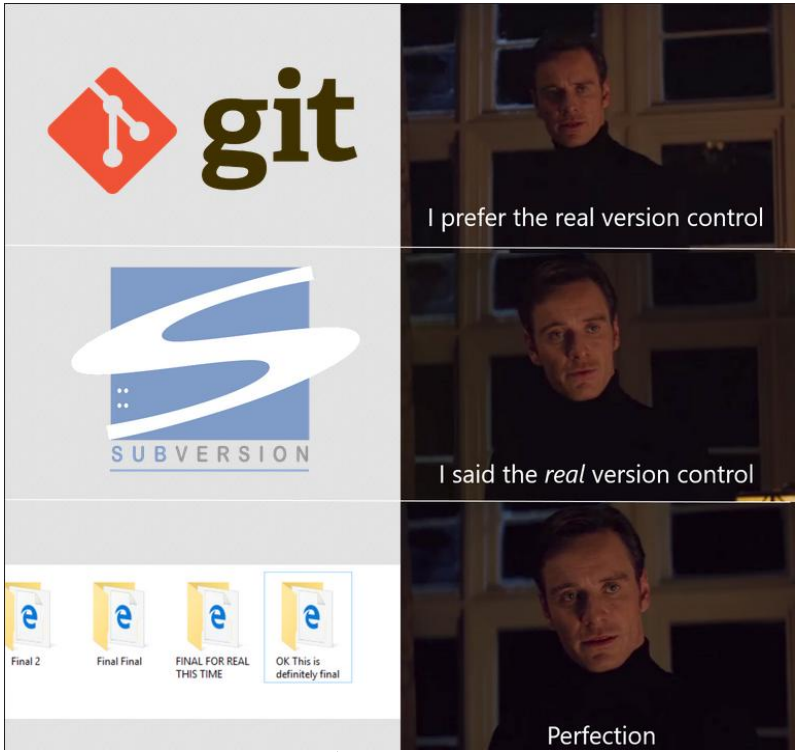
Git history

- Version control system (VCS)
- Created for the Linux Kernel
- Goals when designing Git:
  - Speed
  - Simple design
  - Good for non-linear development
  - Handle large projects

©Britannica – Linus Torvalds

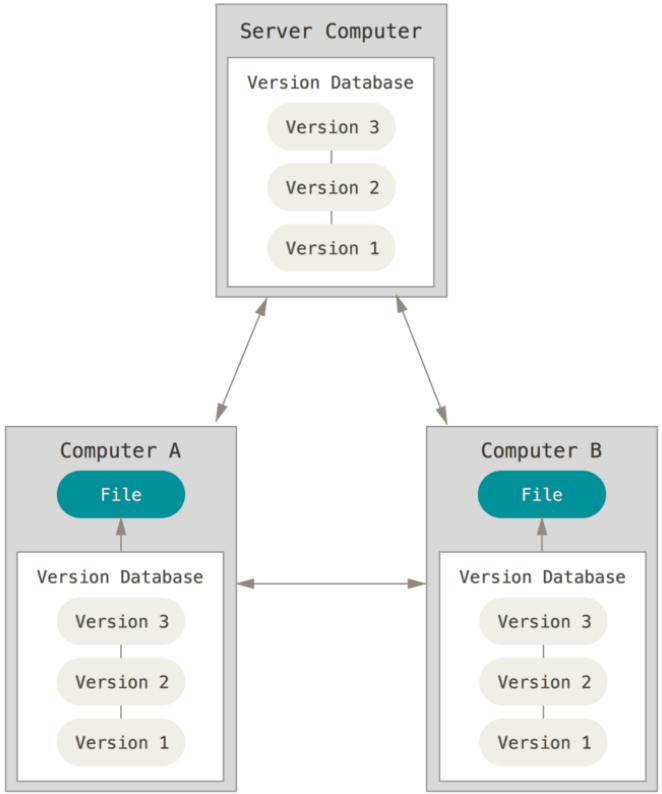
Why bother with version control?

- Keep track of changes
  - Possibility to revert back
- Avoid messes



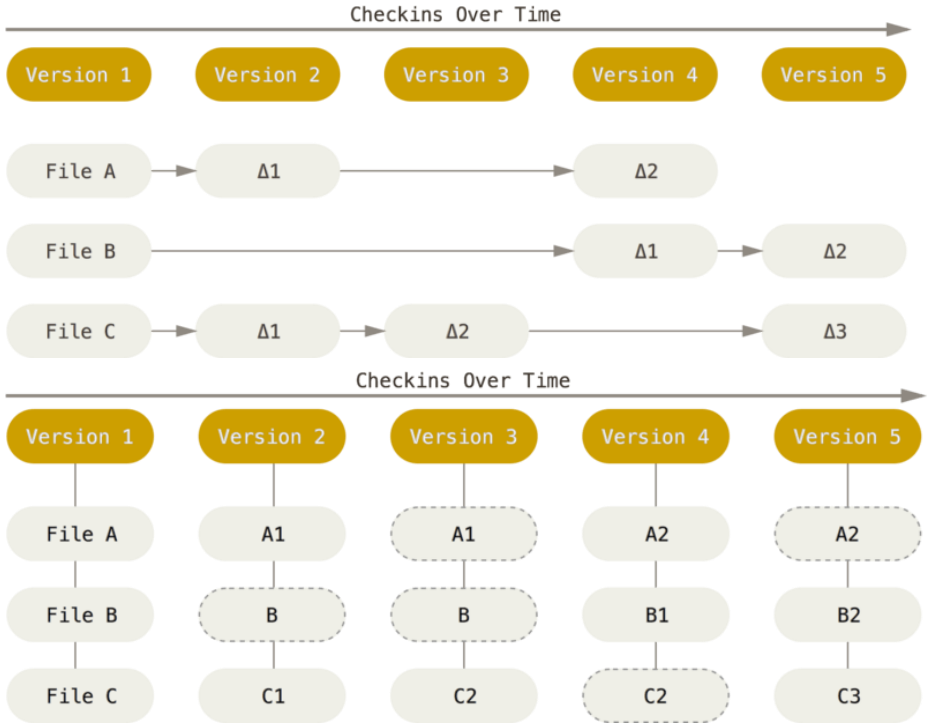
What is a VCS

- VCS Version Control System
- LVCS Local VCS
- CVCS Centralized VCS
- DVCS Distributed VCS



Differences between VCS

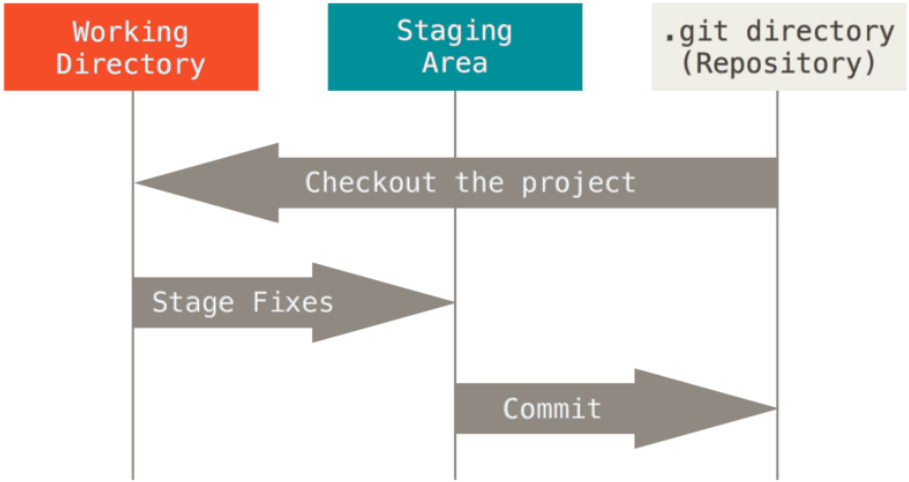
- Delta-based
  - Track file changes over time.
- Snapshot-based
  - Like taking a “picture” of the current files
- Git is snapshot based



©git-scm.com – Delta based (Top) and Snapshot based (bottom)

The stages of Git

- Modified
- Staged
- Committed



©git-scm.com – Working Tree, staging area and Git directory

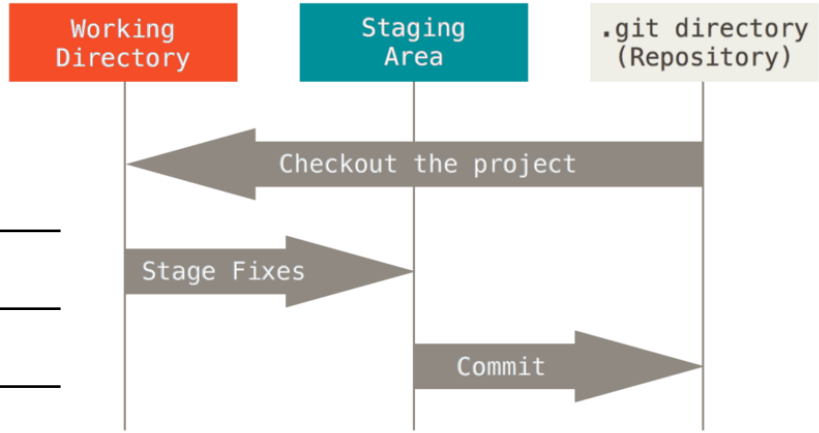
Git CLI vs Github Desktop

- Dekstop ok for everyday use
- CLI grants control
  - Runs anywhere
  - Advanced options

- Other options:
  - Source control in VScode
  - GitKraken

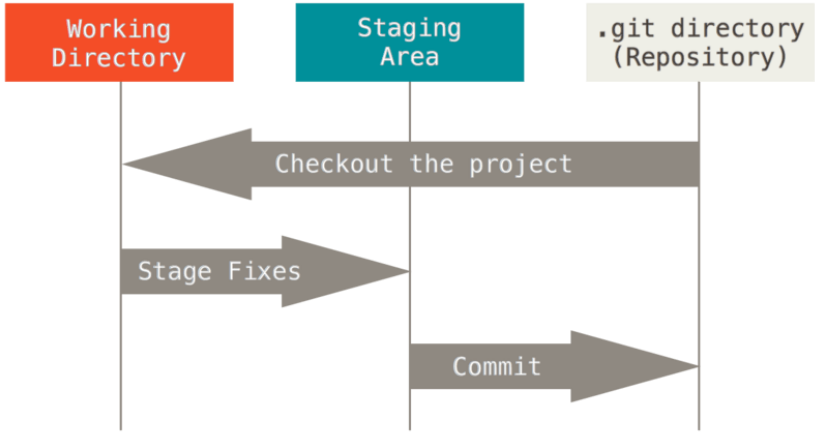
Common commands

git init	Initialize project
git clone	Clone existing repository
git add	Add files to staging area
git commit	Commit to repository
git push	Upload to remote
git pull	Pull from remote
git checkout	Switch between branches
git branch	List branches
git status	Show status of staging area



Initiating / Cloning repository

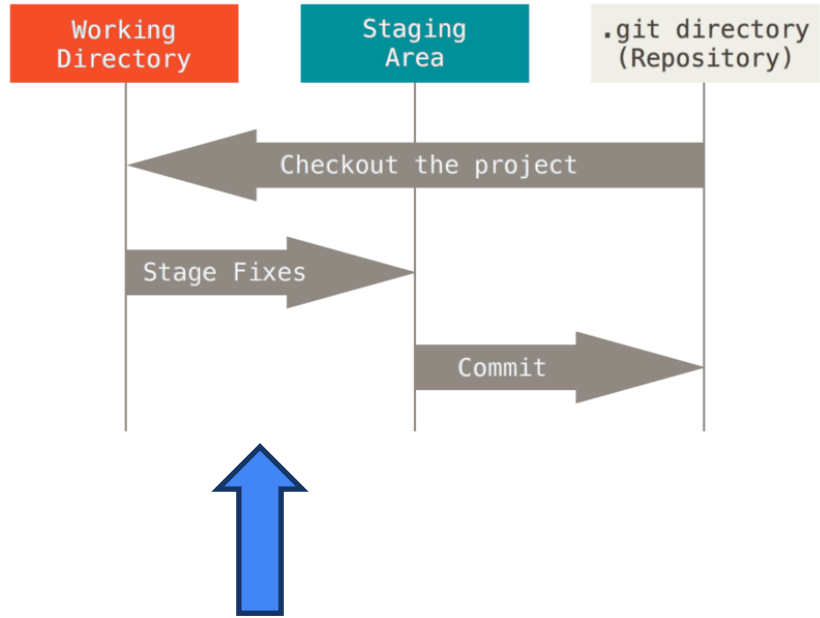
- Create a repository:
- Local: `git init {path}`
- Remote: `git clone {source} [path]`



Notation: [optional] {required}

Git add - Add files

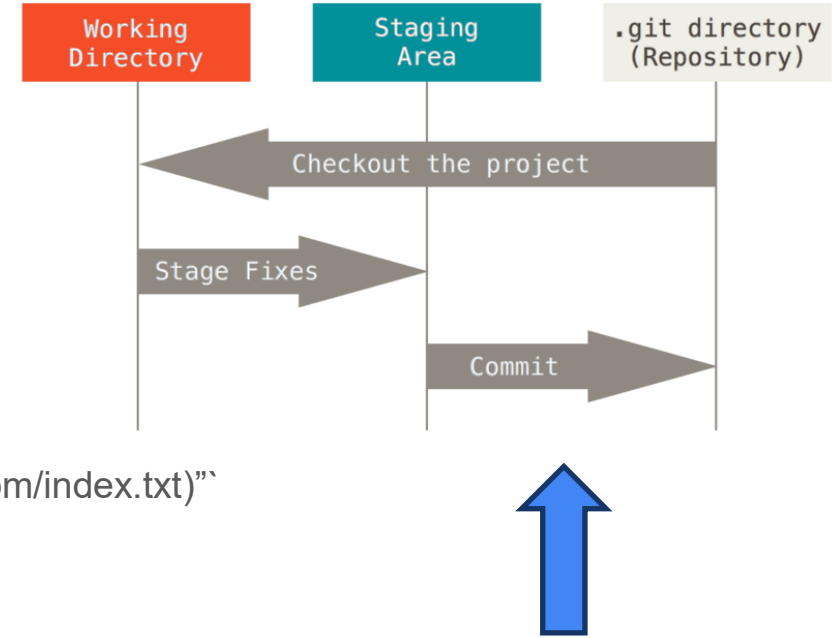
- `git add {path}`
- Don’t add everything!!
  - e.g. `git add * / .`
  - Only track source- and other necessary files.



Commit staging area

- Save staged files
- `git commit [-m “Commit message”]`
  - Use meaningful messages!
  - For the trolls:

`git commit –m “$(curl –s https://whatthecommit.com/index.txt)”`



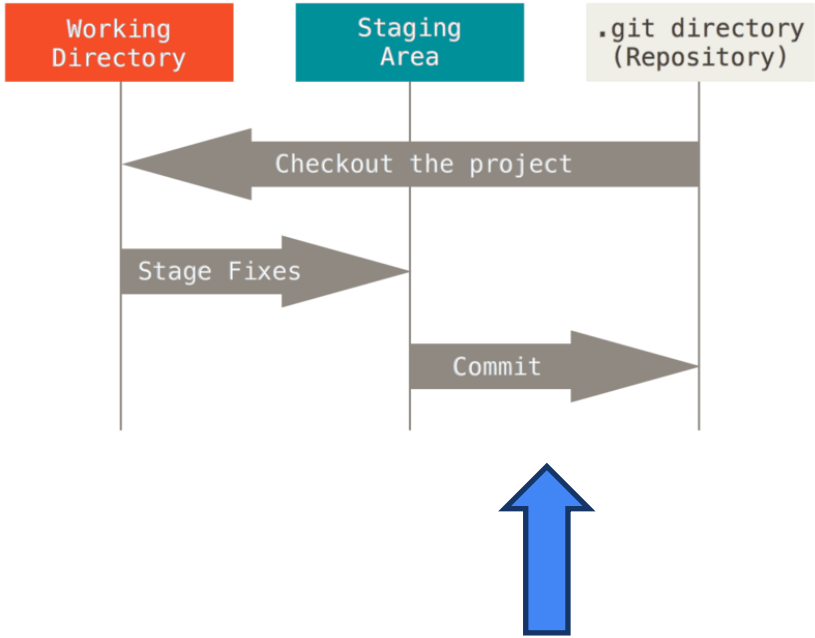

```
Author: Jonas Jakobsen
Date:   Sun Jan 28 20:45:48 2024 +0100

    A descriptive commit message
```

Jonasj2001 A descriptive commit message 9dab8eb · 3 minutes ago 1 Commits

pres.odp A descriptive commit message 3 minutes ago

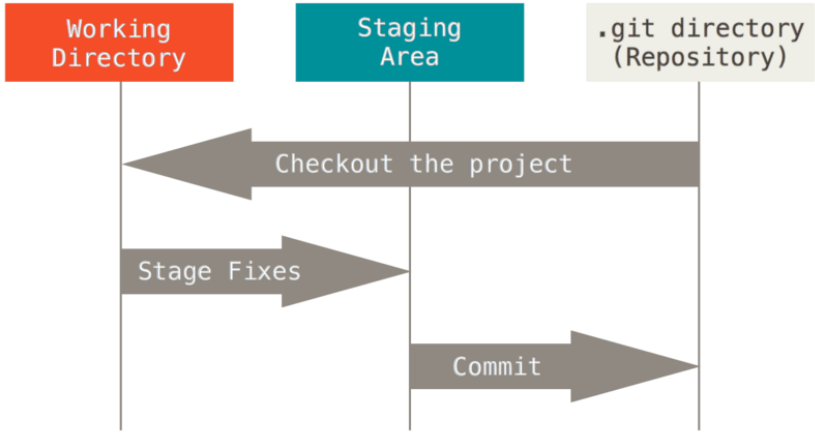
Amending commits

- Only use on the local database!
- `git commit –amend`
  - Overwrites old commit



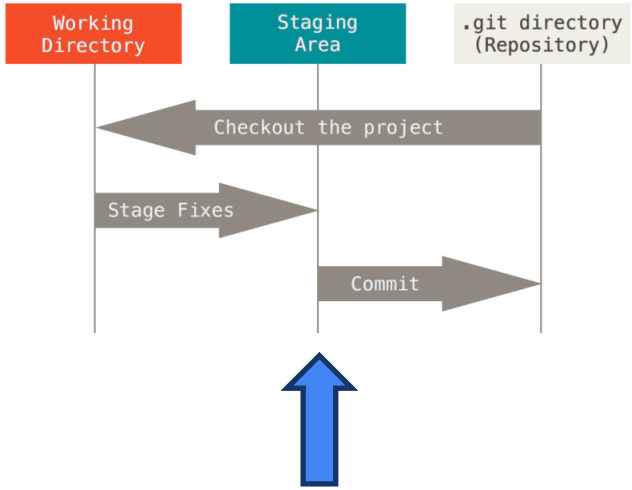
Interacting with remote

- `git pull`
  - Fetches and automatically applies incoming changes to the working directory.
- `git push`
  - Uploads your new .git directory to the remote.
- `git fetch`
  - Retrieves incoming changes without changing the working directory.



Status of staging area

- Keep track of files
- `git status [path]`




```
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   pres.odp

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .~lock.pres.odp#
        newfile

no changes added to commit (use "git add" and/or "git commit -a")
```

```
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   newfile
        modified:   pres.odp

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .~lock.pres.odp#
```

Exercise 1

Get started

AALBORG UNIVERSITY
DENMARK

AAU SATLAB

Exercise 1:

- Install Git (git-scm.com/downloads)
- Generate and add a SSH key to your Github account
- Create a new repository
- Clone the repository and add a file
- Commit and push to Github.

Download slides

Generate SSH-key:

`ssh-keygen –t ed25519 –C “your_email@example.com”`

Add SSH-key to Github:

https://github.com/settings/keys

Free Github Pro for students

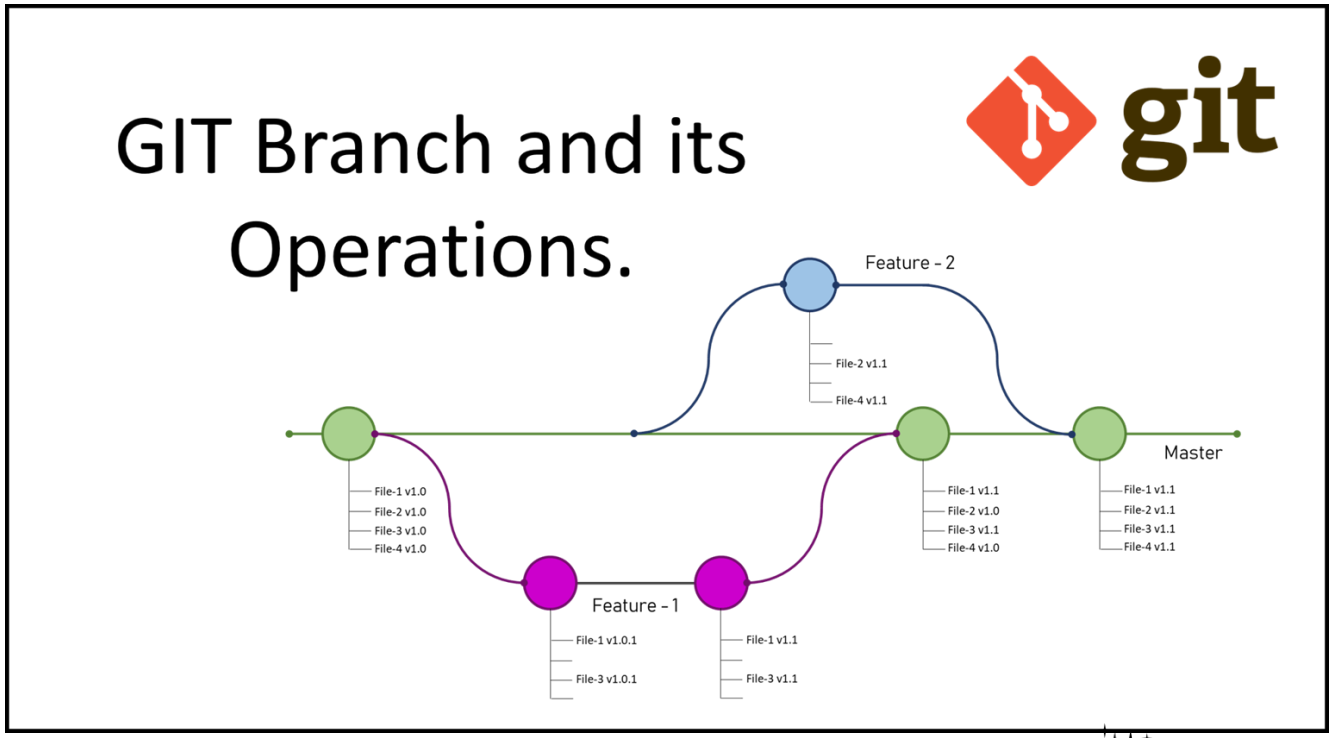
- https://github.com/settings/education/benefits
  - Unlocks more features w. private repositories
  - Copilot for free

Branching

AALBORG UNIVERSITY
DENMARK

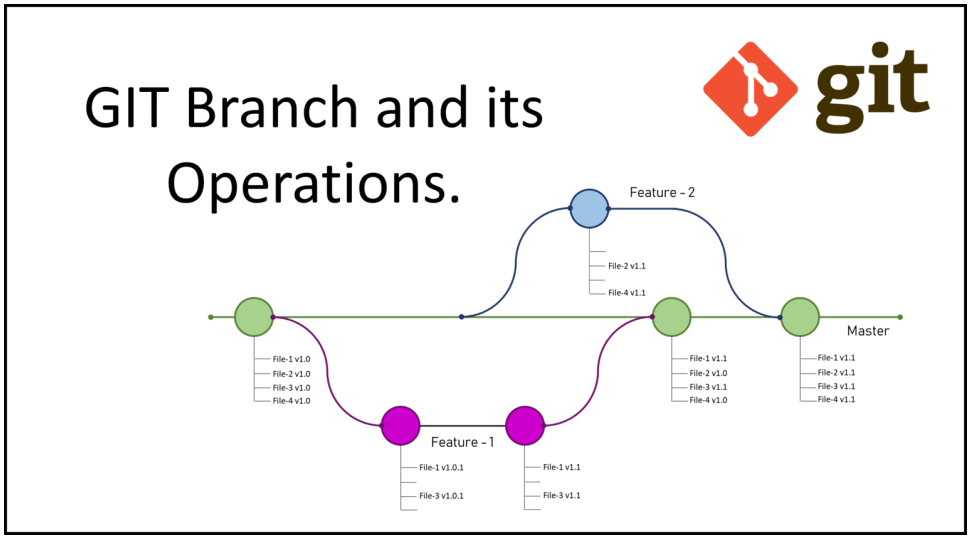
AAU SATLAB

Branching


GIT Branch and its Operations.
git
Feature - 2
File-2 v1.1
File-4 v1.1
Master
File-1 v1.0
File-2 v1.0
File-3 v1.0
File-4 v1.0
File-1 v1.1
File-2 v1.0
File-3 v1.1
File-4 v1.0
File-1 v1.1
File-2 v1.1
File-3 v1.1
File-4 v1.1
Feature - 1
File-1 v1.0.1
File-3 v1.0.1
File-1 v1.1
File-3 v1.1

Create and switch branch

- Create new branch:
  - `git checkout –b {branchname}`
- Switch branch
  - `git checkout {branchname}`



Push new branch to remote

- Error due to Distributed VCS
  - No upstream branch
- Set upstream branch
  - `git push –u origin HEAD`
  - -u origin = --set-upstream origin
  - HEAD (Reference to last commit)

```
fatal: The current branch newfeature has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin newfeature

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.
```

Merge branches

- Merge to current branch:
  - `git merge {branch} [-m message]`
- Remove branch:
  - Local: `git branch -d {local_branch}`
  - Remote: `git push origin -d {remote_branch}`

Pull Request (PR)

- Tool for collaboration
  - Useful for code review

Exercise 2

Branching

AALBORG UNIVERSITY
DENMARK

AAU SATLAB

Exercise 2:

- Create a new branch
- Checkout your new branch
- Add a new file
- Commit and Push to origin
- See your new branch online
- Merge your branch to main, or create and close a PR
- Delete the stale branch(-es)

Hungry?

- Register via. QR code or: forms.office.com/e/WtVF7VLvay
- If possible, use your student mail

Merge Conflicts

AALBORG UNIVERSITY
DENMARK

AAU SATLAB

How does a merge conflict look

Here are lines that are either unchanged from the common
ancestor, or cleanly resolved because only one side changed,
or cleanly resolved because both sides changed the same way.
<<<<<<< yours:sample.txt
Conflict resolution is hard;
let's go shopping.
=======
Git makes conflict resolution easy.
>>>>>>> theirs:sample.txt
And here is another line that is cleanly resolved or unmodified.

<<<<<<<	Your changes are located below
=======	Marks conflict
>>>>>>>	Remote changes located above

When does merge conflicts occur

- Git is good at merging files.
  - Conflicts happens if the same line is edited.

- Tools to help with source control:
  - VS Code
  - Git Kraken

Exercise 3

Merge conflicts

AALBORG UNIVERSITY
DENMARK

AAU SATLAB

Exercise 3

- Split into groups of two
- Edit the same file in two different places.
  - What happens when you push and pull?
- Repeat by editing the same line.
  - What happens?
- Resolve any potential merge conflict

Additional Features

AALBORG UNIVERSITY
DENMARK

AAU SATLAB

Stop git from tracking files

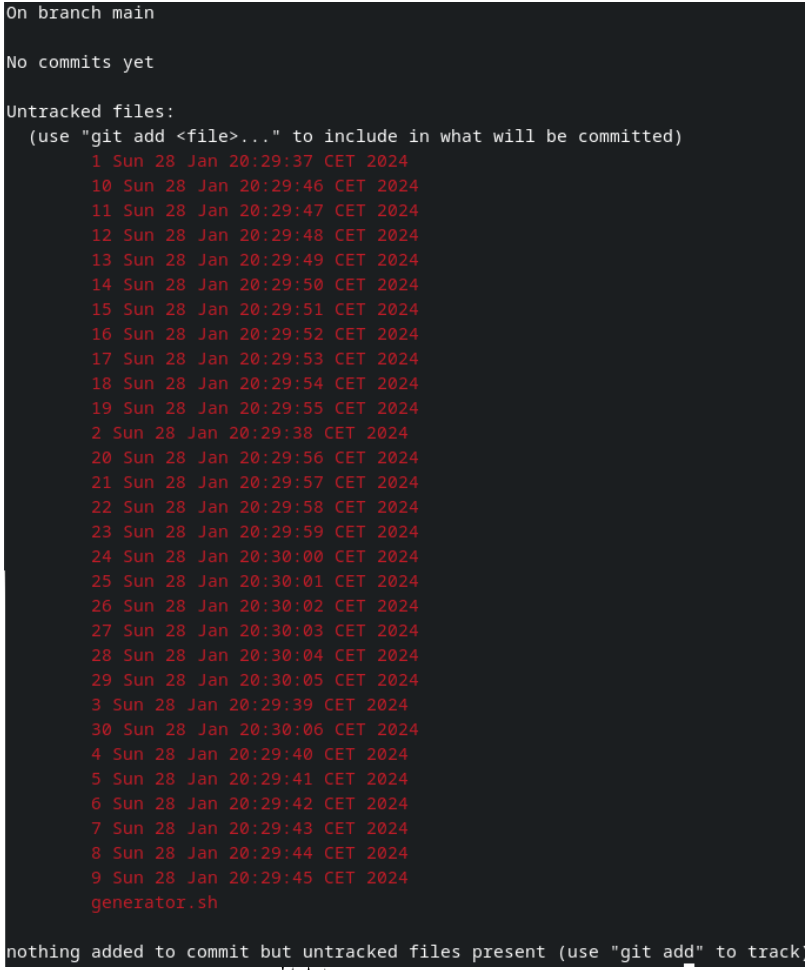
- Use `.gitignore` files.
- Tells git not to track specific files
  - E.g. `Target` folder in Rust projects

```
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        1 Sun 28 Jan 20:29:37 CET 2024
        10 Sun 28 Jan 20:29:46 CET 2024
        11 Sun 28 Jan 20:29:47 CET 2024
        12 Sun 28 Jan 20:29:48 CET 2024
        13 Sun 28 Jan 20:29:49 CET 2024
        14 Sun 28 Jan 20:29:50 CET 2024
        15 Sun 28 Jan 20:29:51 CET 2024
        16 Sun 28 Jan 20:29:52 CET 2024
        17 Sun 28 Jan 20:29:53 CET 2024
        18 Sun 28 Jan 20:29:54 CET 2024
        19 Sun 28 Jan 20:29:55 CET 2024
        2 Sun 28 Jan 20:29:38 CET 2024
        20 Sun 28 Jan 20:29:56 CET 2024
        21 Sun 28 Jan 20:29:57 CET 2024
        22 Sun 28 Jan 20:29:58 CET 2024
        23 Sun 28 Jan 20:29:59 CET 2024
        24 Sun 28 Jan 20:30:00 CET 2024
        25 Sun 28 Jan 20:30:01 CET 2024
        26 Sun 28 Jan 20:30:02 CET 2024
        27 Sun 28 Jan 20:30:03 CET 2024
        28 Sun 28 Jan 20:30:04 CET 2024
        29 Sun 28 Jan 20:30:05 CET 2024
        3 Sun 28 Jan 20:29:39 CET 2024
        30 Sun 28 Jan 20:30:06 CET 2024
        4 Sun 28 Jan 20:29:40 CET 2024
        5 Sun 28 Jan 20:29:41 CET 2024
        6 Sun 28 Jan 20:29:42 CET 2024
        7 Sun 28 Jan 20:29:43 CET 2024
        8 Sun 28 Jan 20:29:44 CET 2024
        9 Sun 28 Jan 20:29:45 CET 2024
        generator.sh

nothing added to commit but untracked files present (use "git add" to track
```

.gitignore

- Stop tracking of all files ending in 2024
  - Add: `*2024` to .gitgnore
- Transforms output to

```
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        generator.sh

nothing added to commit but untracked files present (use "git add" to track)
```



Git Worktrees

- Used to work with multiple copies of the same repository,
  - Uses only one database
  - No need to clone multiple times.
- `git worktree add [-b <new branch>] {path} [commit]`
  - Default to existing/new branch named after path name
  - Use [commit] to specific a specific branch (Only for existing branches)
- `git worktree remove {path}`
  - Remove worktree from list
- `git worktree list`
  - Show worktrees

Resetting / Restoring

- `git reset HEAD`
  - Unstage staged files to HEAD
- `git restore {file}
  - Remove modifications from file
- `git restore –staged {file}
  - Unstage file
- `git stash`
  - Temporarily save current progress
- `git stash pop`
  - Retrieve last stashed entry.

Next event: Docker!!!

- Next Thursday (20/11)
  - 16:15 in A4-106
- Sign up for food:
  - https://forms.office.com/e/QXZvBz3mC8
- Event page on AAU Satlab Facebook