# Git

# Oversigt

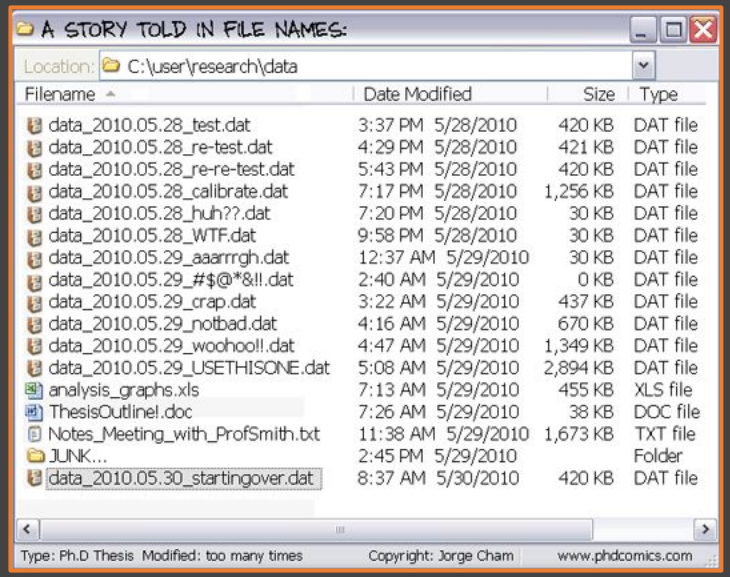
- Introduktion
- Hvad er Git?
- Hvorfor er det smart?
- Hvordan virker det?
- Git kommandoer
- Git workflow
- Eksempel
- Opgave


- Oversigt
- Introduktion
- Hvad er Git?
- Hvorfor er det smart?
- Hvordan virker det?
- Git kommandoer
- Git workflow
- Eksempel
- Opgave

# Introduktion

- Johan Vasegaard
- IKT studerende
- 6. Semester
- Praktik hos kamstrup, i IT-udviklingsafdeling

# Hvad er Git?



# Hvad er Git?

Hvad vil vi gerne have?

- En version af hver fil, som er den vi arbejder på lige nu.
- Historik så vi kan gå tilbage af gamle versioner af en fil.
- At kunne være flere udviklere på et projekt der sidder og arbejder på de samme filer, uden at ødelægge noget.

Kan vi få det med Git?

- Ja
- Ja
- Ja*

# Hvorfor er det smart?

Vi kan altid gå tilbage til en tidligere version af et projekt, eller en tidligere version af en specifik fil!

Vi arbejder altid på en enkelt version af hver fil, ikke noget med:
HelloWorld_1, HelloWorld_2, HelloWorld_2_1, etc...

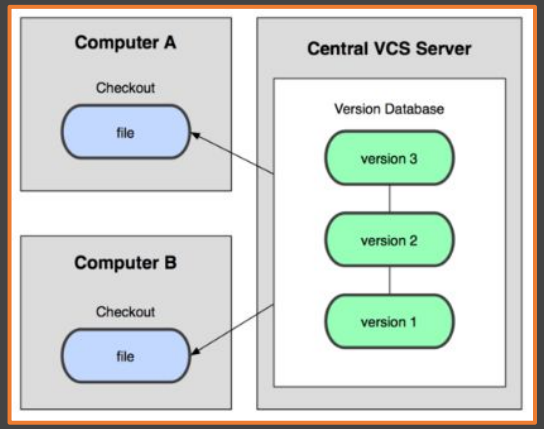
Siden at vores filer er synkroniseret med et fælles repo, og siden at Git holder øje med hvem der lave hver eneste lille ændring, kan flere udviklere side og arbejde på samme projekt.
(Dog skal de stadig følger reglerne…)

# Hvordan virker det?

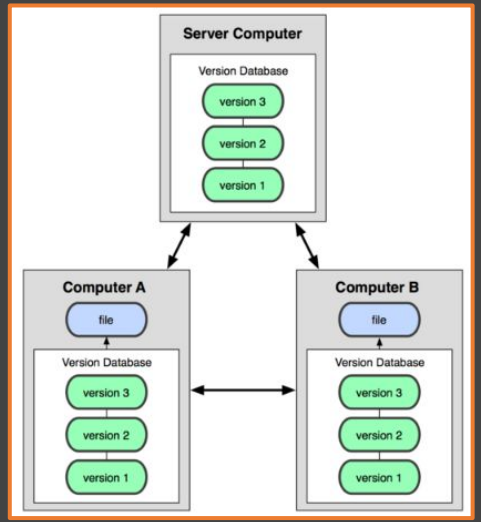
Git er et “Version Control System” (CVS)

Specifikt er det et “Distributed Version Control System” (DVCS)

**CVS**

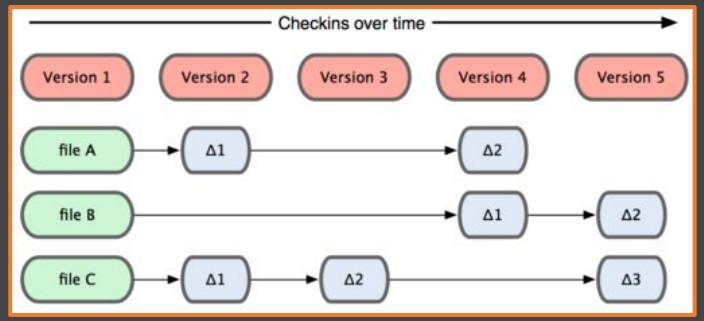



**DCVS**

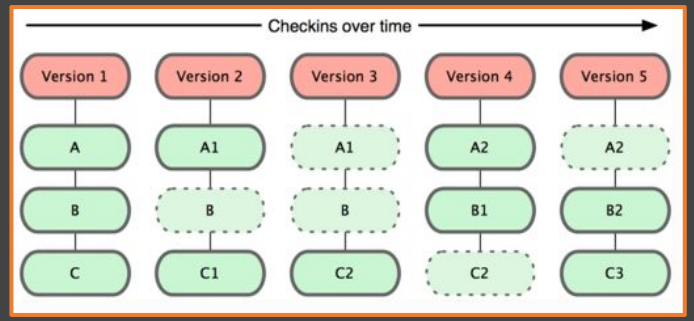


# Hvordan virker det?

Traditionel versionskontrol

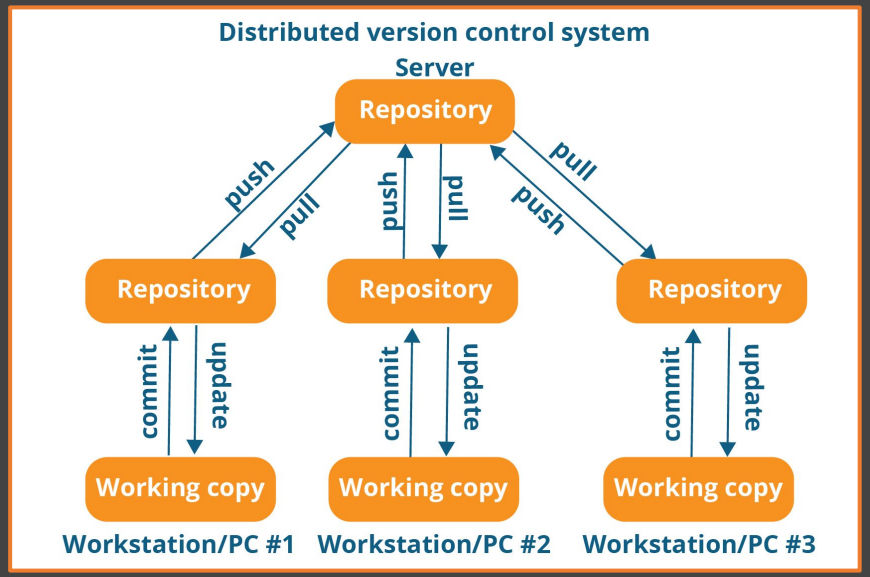



Git



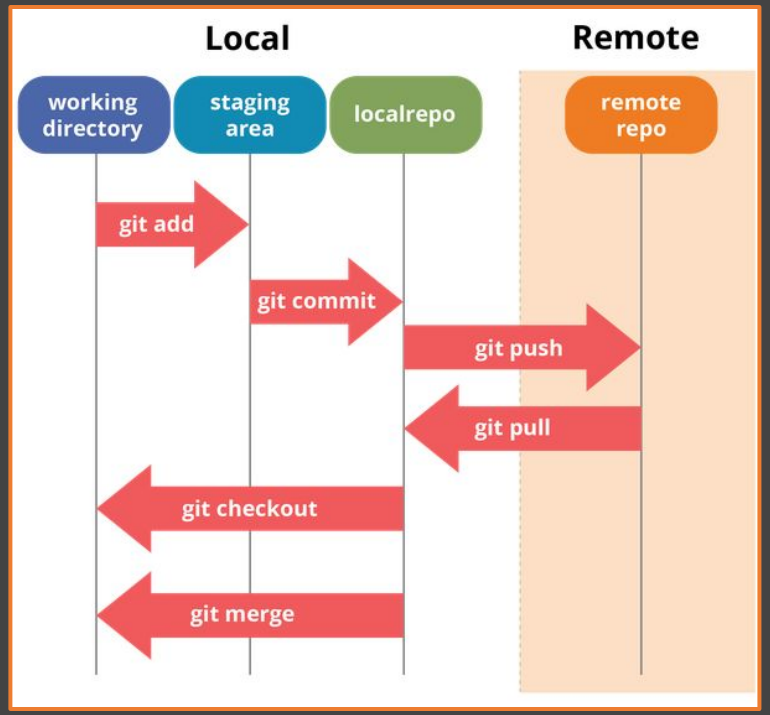

Git tager “snapshots” af de filer det ligger på en computer. Hver gang vi opdaterer vores repository sammenligner Git det nyt snapshot med det gamle.
Hvis det ikke er ændringen i en fil laver Git bare en reference til det tidligere snapshot.

# Hvordan virker det?

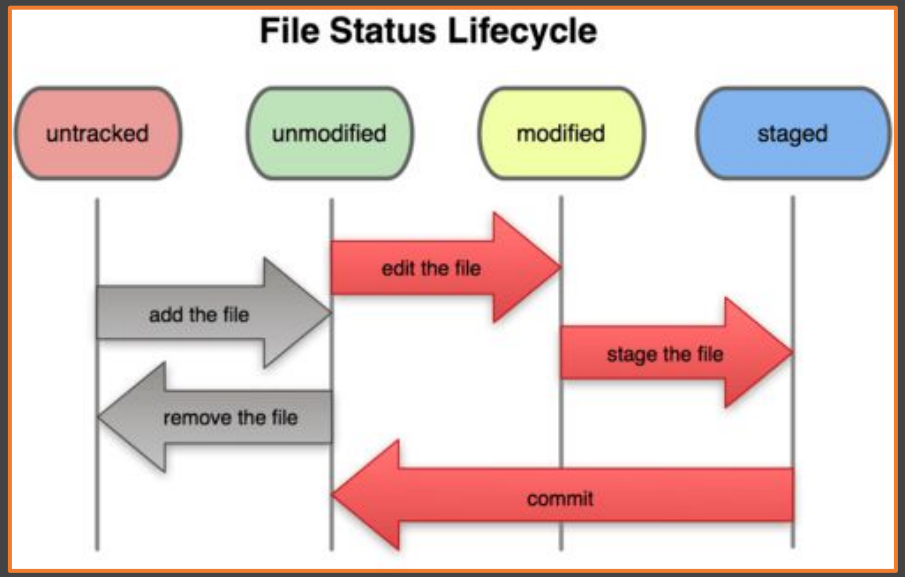



- Oversigt
- Introduktion
- Hvad er Git?
- Hvorfor er det smart?
- Hvordan virker det?
- Git kommandoer
- Git workflow
- Eksempel
- Opgave

# Hvordan virker det?



# Hvordan virker det?




- Oversigt
- Introduktion
- Hvad er Git?
- Hvorfor er det smart?
- Hvordan virker det?
- Git kommandoer
- Git workflow
- Eksempel
- Opgave

# Git kommandoer

| command | description |
|---|---|
| `git clone url [dir]` | copy a git repository so you can add to it |
| `git add files` | adds file contents to the staging area |
| `git commit` | records a snapshot of the staging area |
| `git status` | view the status of your files in the working directory and staging area |
| `git diff` | shows diff of what is staged and what is modified but unstaged |
| `git help [command]` | get help info about a particular command |
| `git pull` | fetch from a remote repo and try to merge into the current branch |
| `git push` | push your new branches and data to a remote repository |
| others: `init, reset, branch, checkout, merge, log, tag` | |

# Git workflow

Basic Git workflow:

1. Modify files in your working directory.
2. Stage files, adding snapshots of them to your staging area.
3. Do  a commit, which takes the files as they are in the staging area and stores that snapshot permanently to your Git directory.

Notes:

- If a particular version of a file is in the git directory, it's considered committed.
- If it's modified but has been added to the staging area, it is staged.
- If it was changed since it was checked out but has not been staged, it is modified.

# Git workflow

Good Git practice:

1. Add and Commit your changes to your local repo.
2. Pull from remote repo to get most recent changes (fix conflicts if necessary, add and commit them to your local repo).
3. Push your changes to the remote repo.

# Eksempel

- Oversigt
- Introduktion
- Hvad er Git?
- Hvorfor er det smart?
- Hvordan virker det?
- Git kommandoer
- Git workflow
- Eksempel
- Opgave

# Opgave

1. Sæt jeres Git info
   a. git config --global user.name “navn”
   b. git config --global user.email “din@email.com”
2. Clone jere gruppe repo fra redmine
3. Tilføj en .txt fil til jeres remote repo
4. All medlemmer tilføjer en linje til filen, en af gangen
   a. Husk at lave et pull når der er blevet lavet ændringer på remote repo
5. - kort opsummering -
6. To medlemmer skriver noget på samme linje i en fil, vi håndtere den merge-conflict der kommer.

# Bonus!

- Få hjælp i command prompt (command = config, add commit, etc..)
  - git help “command”
  - git “command” --help
  - man git “command”
- Gratis online bog: https://git-scm.com/book
- Git website: https://git-scm.com/
- Git tutorial 1: https://git-scm.com/docs/gittutorial
- Git tutorial 2 (simple): http://rogerdudler.github.io/git-guide/
- Git reference website: https://git-scm.com/docs
- Resolving merge-conflict: https://githowto.com/resolving_conflicts

# Næste gang?

- Mere om merging
- Branching og hvordan man bruger det
- Skifte mellem versioner af et projekt (gå til gamle commits)
- Andre Git-interfaces