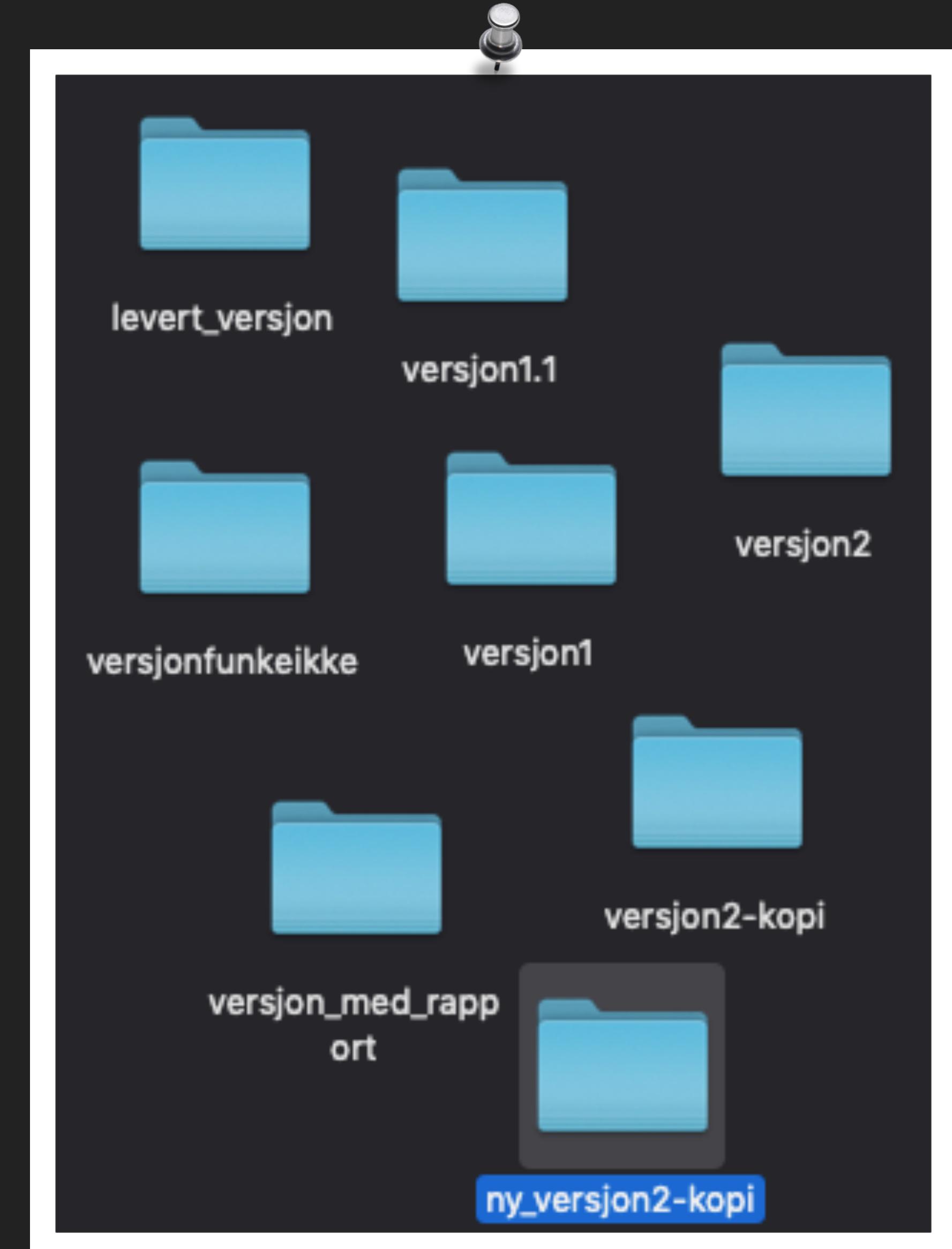


TROMSØSTUDENTENES DATAFORENING'S

KURS I VERSJONSKONTROLL

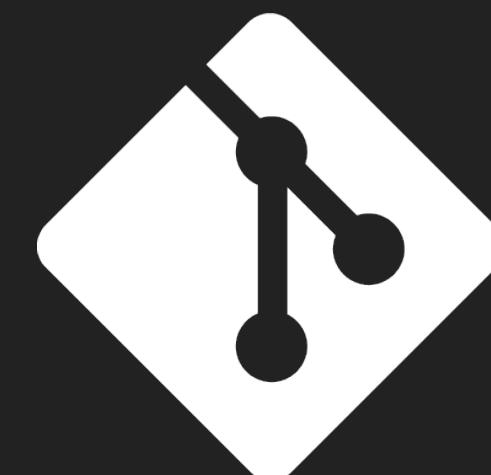
HVA ER VERSJONSKONTROLL?

- ▶ En historikk over hva som er gjort (og av hvem).
- ▶ Men hvorfor?
 - ▶ Mulighet for å «rulle tilbake».
 - ▶ Finne ut hvor det gikk galt
 - ▶ Lettere å gjøre endringer uten å nødvendigvis ødelegge alt som tidligere har eksistert

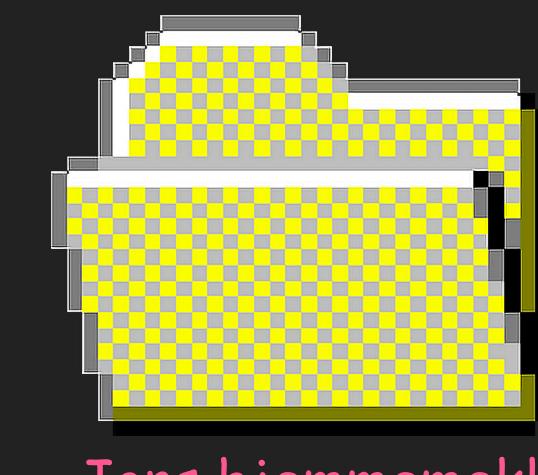
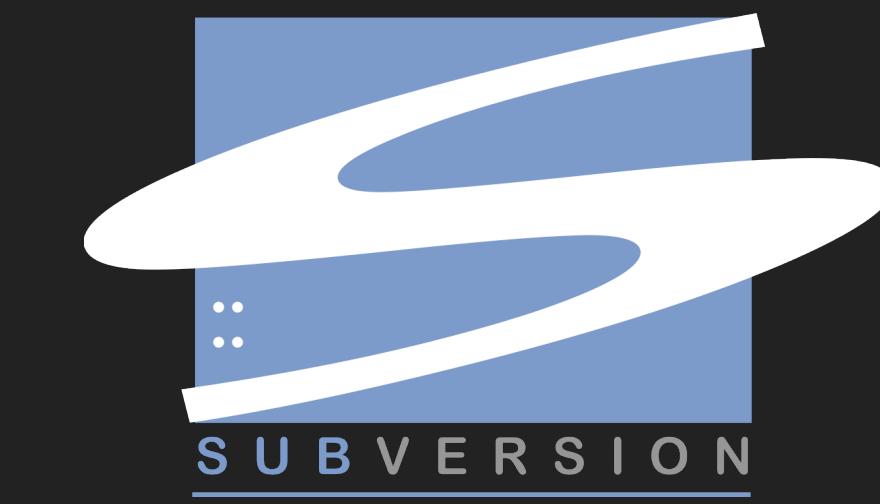


HVORFOR VERSJONSKONTROLL FOR KODE?

- ▶ CTRL+Z fungerer bare så langt tilbake i tid.
- ▶ Stadig nye versjoner, med forbedringer* og feilrettinger.
 - ▶ Feilrettinger spesifikt til gamle versjoner
- ▶ Viktig å ha kontroll på hver minste endring i koden mellom versjoner.
- ▶ Enklere å samarbeide med prosjekter.
- ▶ Enklere å dokumentere koden



git



* bugs



git

THE STUPID CONTENT TRACKER

LITT OM GIT

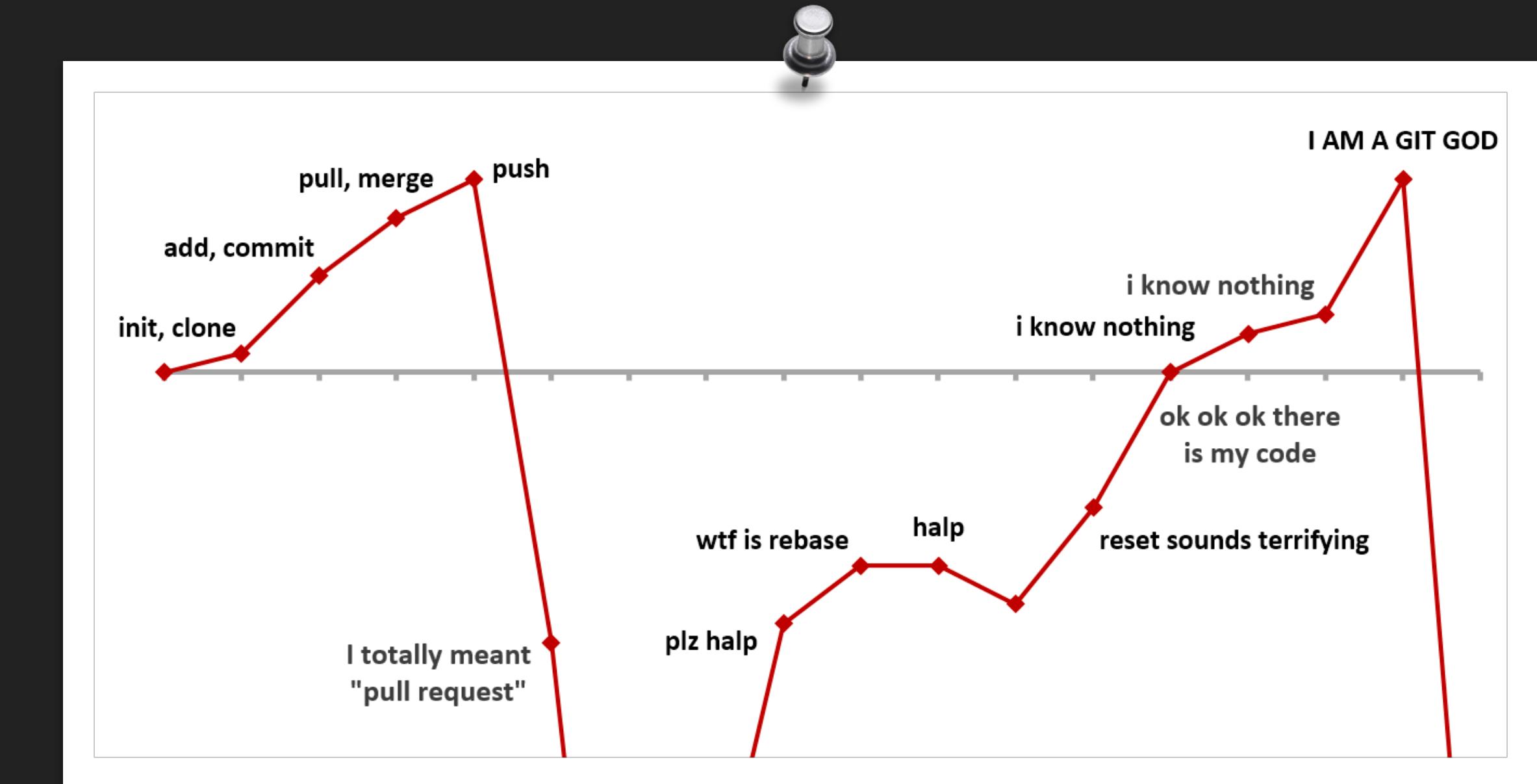
- ▶ En **protokoll** for versjonering av filer
- ▶ «Mappe» basert
- ▶ Holder styr på:
 - ▶ Versjon av filene
 - ▶ Hvem og hva for endringer
 - ▶ Enkelt* å dele prosjekter/kode
 - ▶ Enkelt å beskrive hva som er gjort når og hvor



* Det blir fort litt rot!

LITT OM GIT 2

- ▶ git er bransjestandard!
- ▶ Gigantisk open source community som bruker git
- ▶ Flere måter å bruke git på!
- ▶ Enkelt og fantastisk verktøy når man er over «kneika»
- ▶ Øvelse gjør mester, og her trengs det øvelse!



* Det blir fort litt rot!

HVORDAN FUNKER GIT?*

- ▶ Lagrer en «endringer» av filene dine i fine små «bokser»
- ▶ Boksene stables i hyller i rekkefølge
- ▶ Du kan hente og dele bokser ved hjelp av pekere

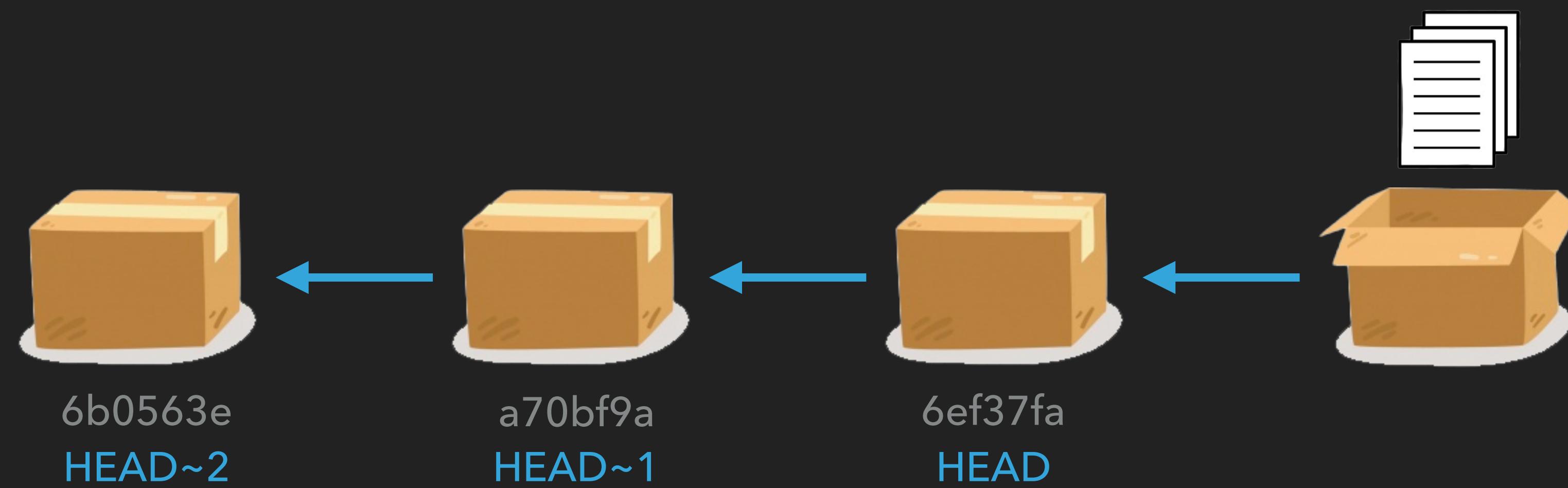
(I starten vil dette gjøres så enkelt som mulig)



*MEGET FORENKLET

HVA MED PEKERNE?

- ▶ Du jobber alltid «på» siste lukkede eske (`HEAD`)
- ▶ Du putter alltid* endringer i en «åpen» eske som følger (Å putte i en eske heter å «tracke»)
- ▶ Pekerne bruker en id (`commit hash`) for å vite hvem de peker på



* som regel

\$ git status

- ▶ Viser deg:
 - ▶ Hvilken branch vi er på, og om den er bak remote branch*
 - ▶ Hvilke filer som er endret
 - ▶ Hvilken av filene som er «tracked».
- ▶ Brukes ofte!
Hjelper deg å holde følge med hva som skjer.



KURS I VERSJONSKONTROLL

```
$ git add <PATH>
```

- ▶ Putter endrede filer i esken.
- ▶ Hvis du fortsetter å endre filene må de nye endringene legges til på nytt
- ▶ PATH kan være filenavn eller mapper.

Det finnes andre snarveier for å legge flere ting samtidig.



KURS I VERSJONSKONTROLL

\$ git commit

- ▶ 1. Teiper igjen esken, skriver på signaturen og en melding på esken.

Denne meldingen bør forklare hva esken inneholder

1

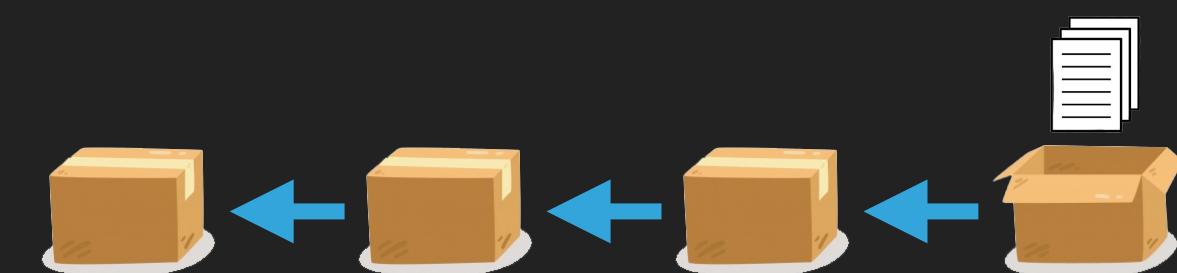


- ▶ Alle filene du har kjørt git add på vil være i denne esken

- ▶ 2. Denne **commiten** blir nå den fremste i rekken, og den neste commiten vil peke på denne

- ▶ I det du kjører kommandoen vil ett nytt vindu åpne seg hvor du får skrive meldingen.
Her kan man lure seg rundt med å bruke -m flagget
git commit -m 'Her er en fin melding'

2



KURS I VERSJONSKONTROLL

\$ git log

Vanlig

- ▶ Viser deg en liste av tidligere commits, hvem som har signert, og hva meldingene er.
- ▶ Populær å lage seg en pyntet versjon for lettere lesbarhet

Pyntet

```
* 78e059c - Working version. Should now do all the magic (6 months ago) <jonjohansen>
* ca6d2e0 - Shuffle some blocks around (7 months ago) <jonjohansen>
* 749207b - Add snippets for github interaction. Going places! :) (7 months ago) <jonjohansen>
* 495c3e0 - Add option to pass in path to structure file (7 months ago) <jonjohansen>
* 2e9e538 - Move methods around to clear up code a bit (7 months ago) <jonjohansen>
* 0cd81c3 - Requests will be used for HTTP communication. Package management added (7 months ago) <jonjohansen>
* 6a0b687 - Add preliminary self-delete function in script (7 months ago) <jonjohansen>
* 61ca85d - Initial project set up (7 months ago) <jonjohansen>
(END)
```

```
commit 78e059c90d217e5e45030c915c2ff5bfcca43c89
Author: jonjohansen <jonjohansen2k@gmail.com>
Date: Mon Apr 8 19:20:21 2019 +0200

Working version. Should now do all the magic

commit ca6d2e09a267a3aaa4e62263209621549552e1ab
Author: jonjohansen <jonjohansen2k@gmail.com>
Date: Wed Apr 3 02:47:08 2019 +0200

    Shuffle some blocks around

commit 749207b6f11d58df29b635b87ec5f3f1466772dc
Author: jonjohansen <jonjohansen2k@gmail.com>
Date: Wed Apr 3 02:39:46 2019 +0200

    Add snippets for github interaction. Going places! :)

commit 495c3e02cc0b0e368719d1f8340882baf2b32f85
Author: jonjohansen <jonjohansen2k@gmail.com>
Date: Mon Apr 1 04:51:35 2019 +0200

    Add option to pass in path to structure file

commit 2e9e538fd41c0164a94519fc2e4dc0b920615d90
Author: jonjohansen <jonjohansen2k@gmail.com>
Date: Mon Apr 1 04:21:22 2019 +0200

    Move methods around to clear up code a bit

commit 0cd81c30c96b6eb7f7109e48e6473eabafc8f763
Author: jonjohansen <jonjohansen2k@gmail.com>
Date: Mon Apr 1 04:14:19 2019 +0200

    Requests will be used for HTTP communication. Package management added

commit 6a0b687d183ffbbba63fb35782c33f98e5c6f58d4
Author: jonjohansen <jonjohansen2k@gmail.com>
Date: Mon Apr 1 02:19:33 2019 +0200

    Add preliminary self-delete function in script

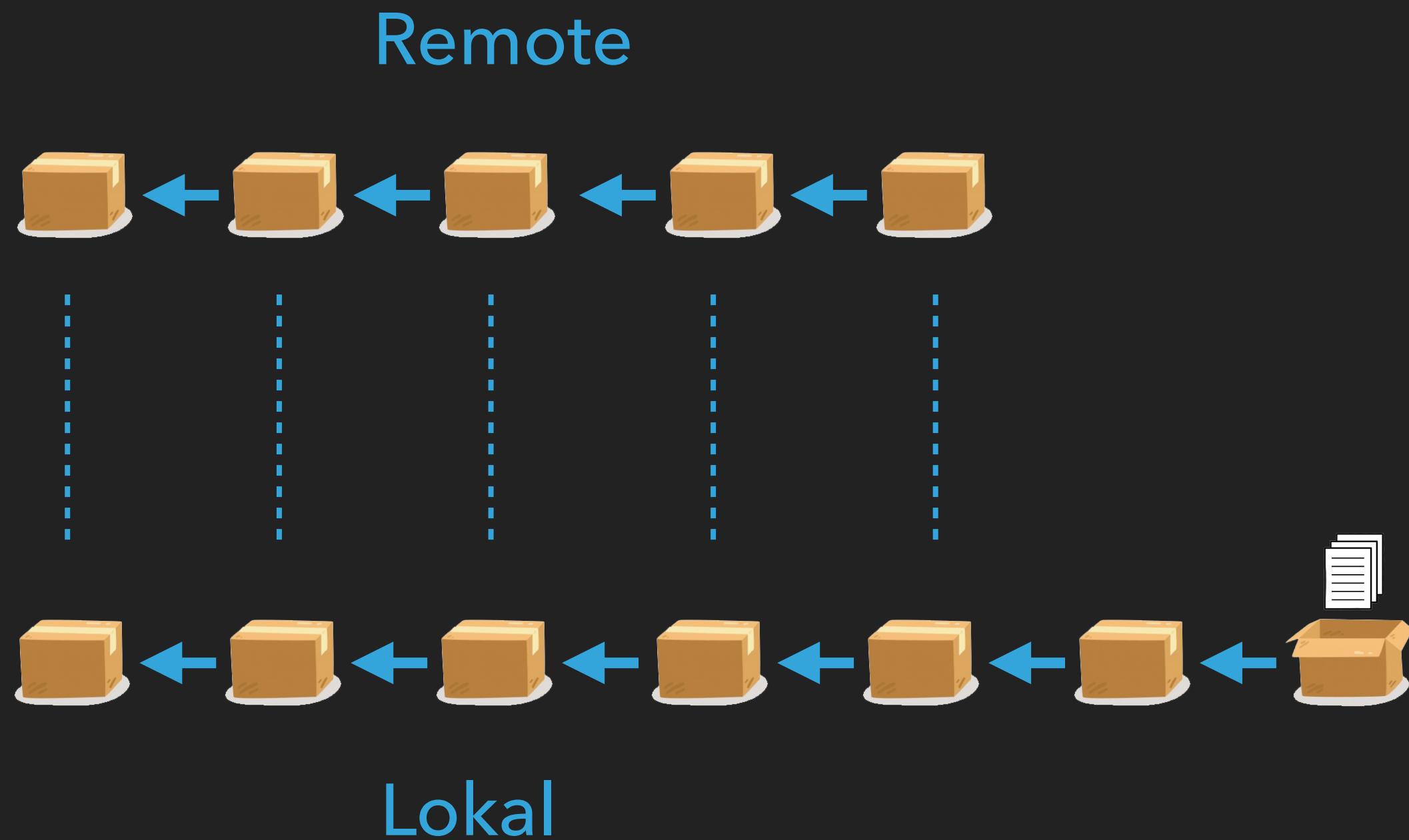
commit 61ca85d06e09216e9010008728bfd9ad62ba13c5
Author: jonjohansen <jonjohansen2k@gmail.com>
Date: Mon Apr 1 01:52:35 2019 +0200

    Initial project set up

    - Reads a json object and creates folders accordingly
    - Parses arguments for sending in username and token
        - If no arguments, it checks config and offers the user to use these or supply own
        - Throw custom errors if none are submitted
    - Emojis
    - Errors
    - Colored print!
(END)
```

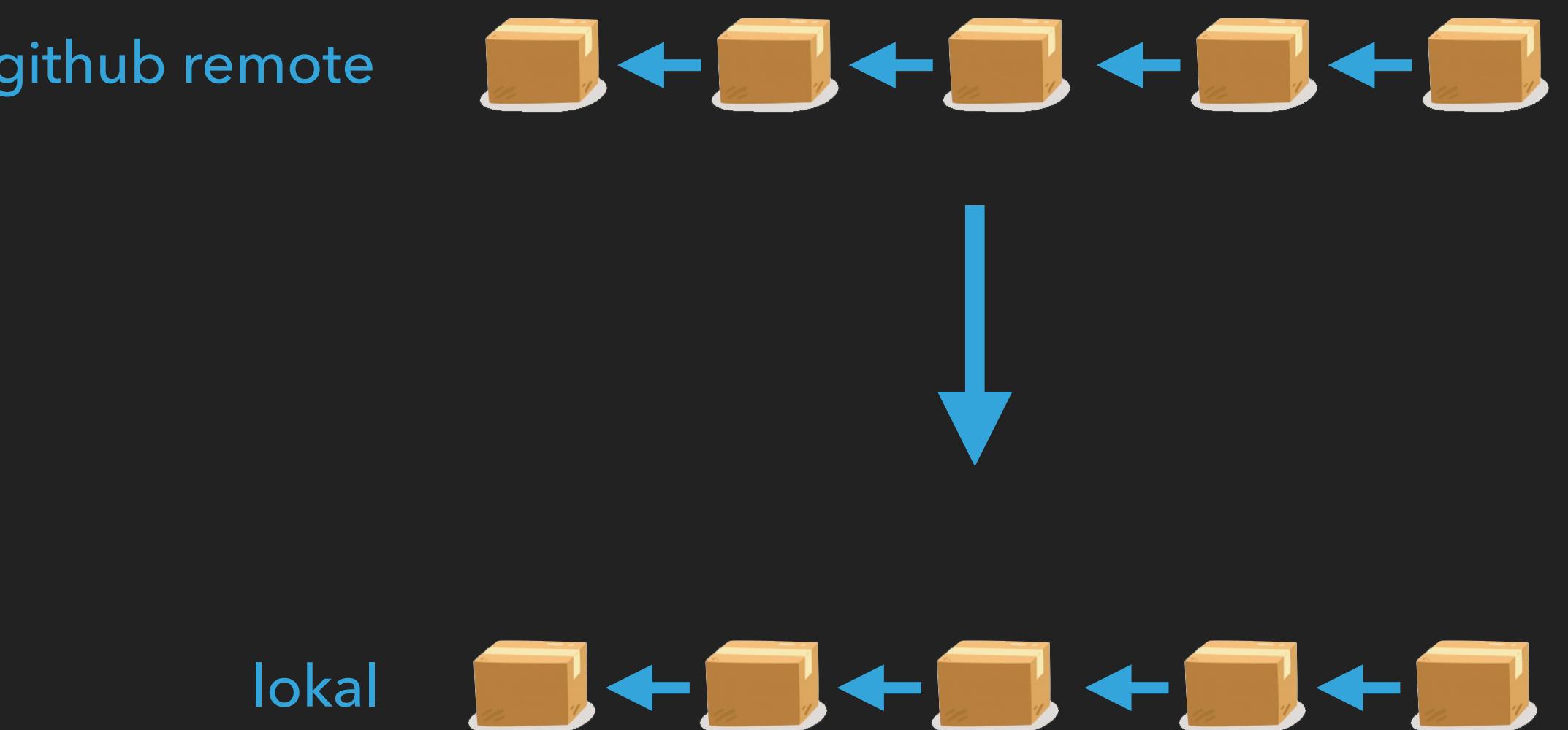
REMOTES

- ▶ En git-historikk som ligger på en «annen plass»
 - ▶ Kopi av ditt repository
- ▶ Typisk jobbe opp mot en git-server, feks github.
- ▶ Man kan ha flere remotes
 - ▶ Den «vanlige» heter origin



```
$ git clone <URL>
```

- ▶ Hente ned et eksisterende prosjekt fra en **remote**
- ▶ Brukes relativt sjeldent
- ▶ «Frisk start»

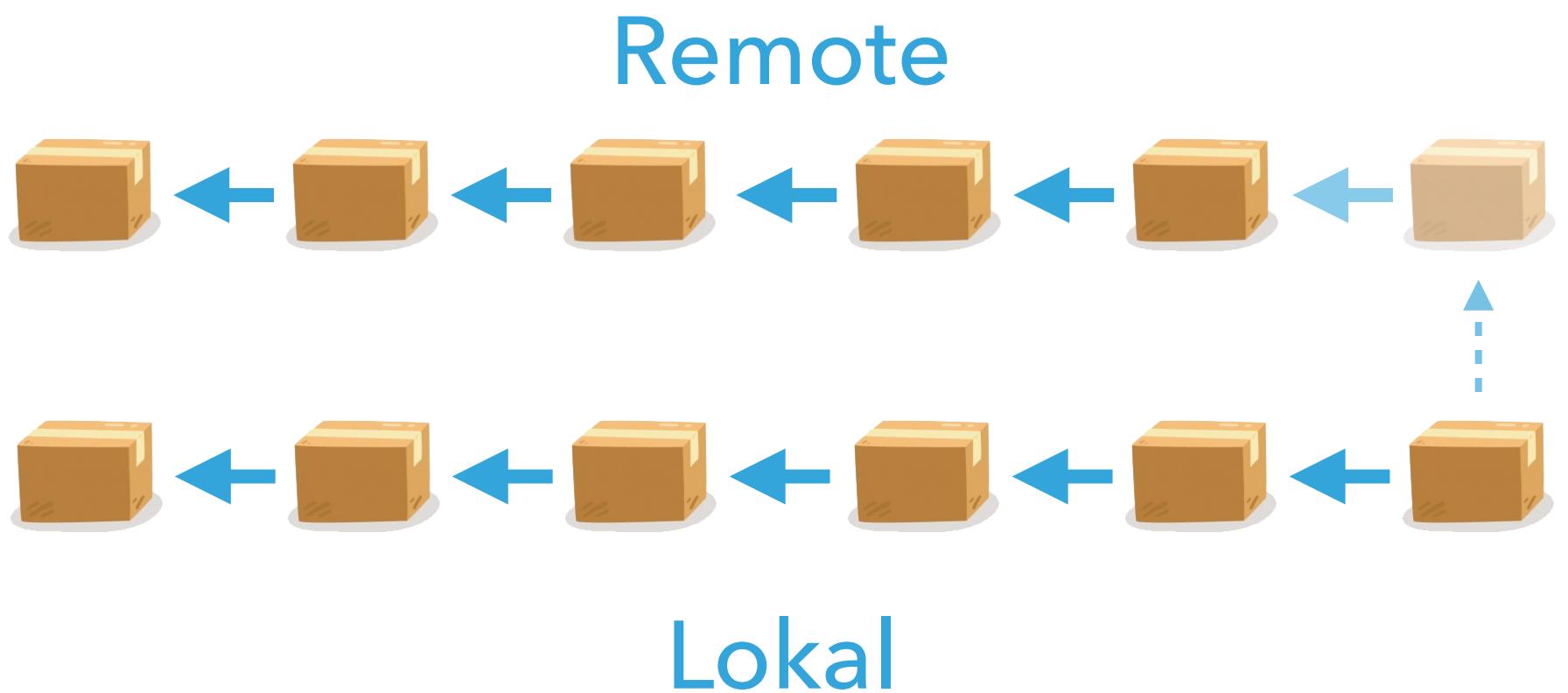


KURS I VERSJONSKONTROLL

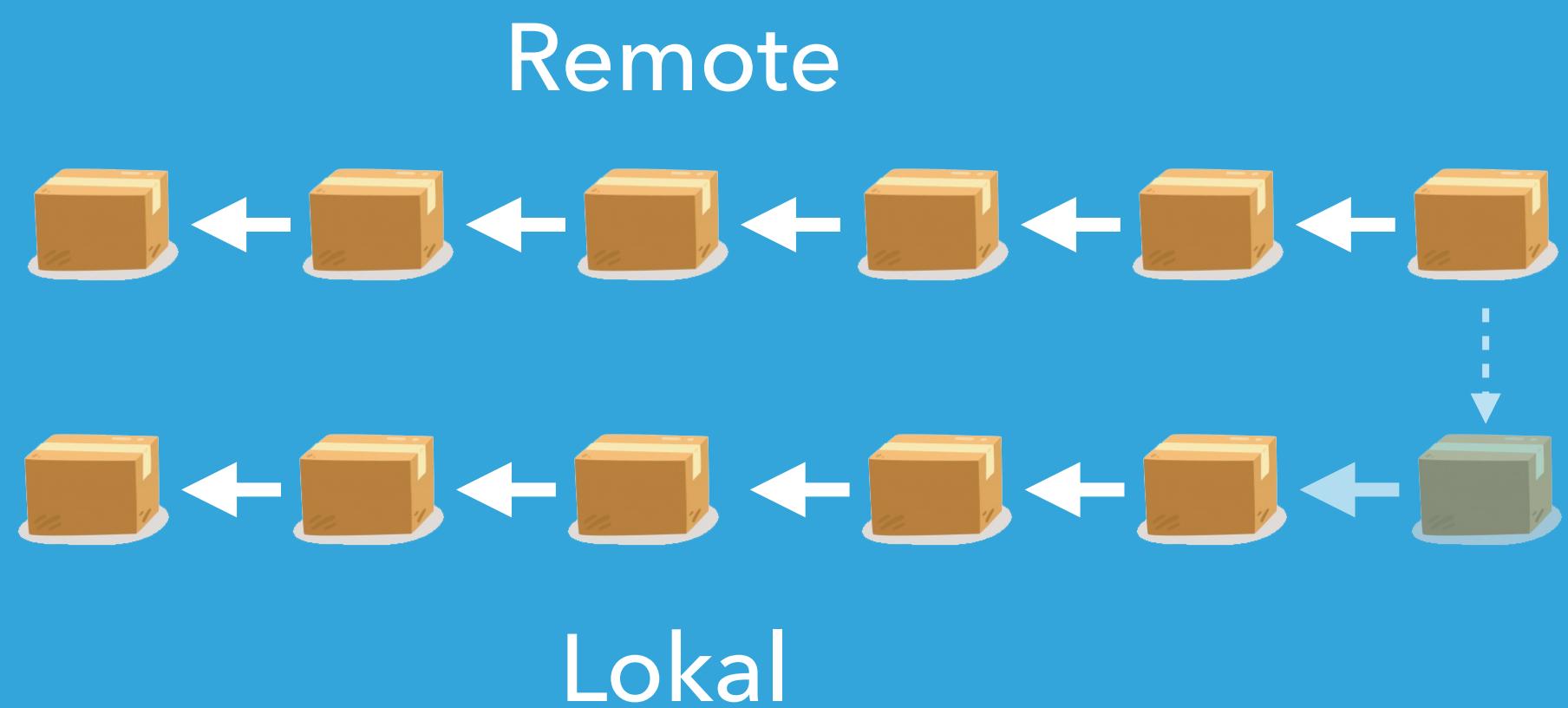
\$ git pull og \$ git push

- ▶ Henter og pusher flere commits i slengen
- ▶ git push
 - ▶ Dytte commits fra lokalt repository opp mot en remote
- ▶ git pull
 - ▶ Hente ned commits fra remote repository

Push



Pull

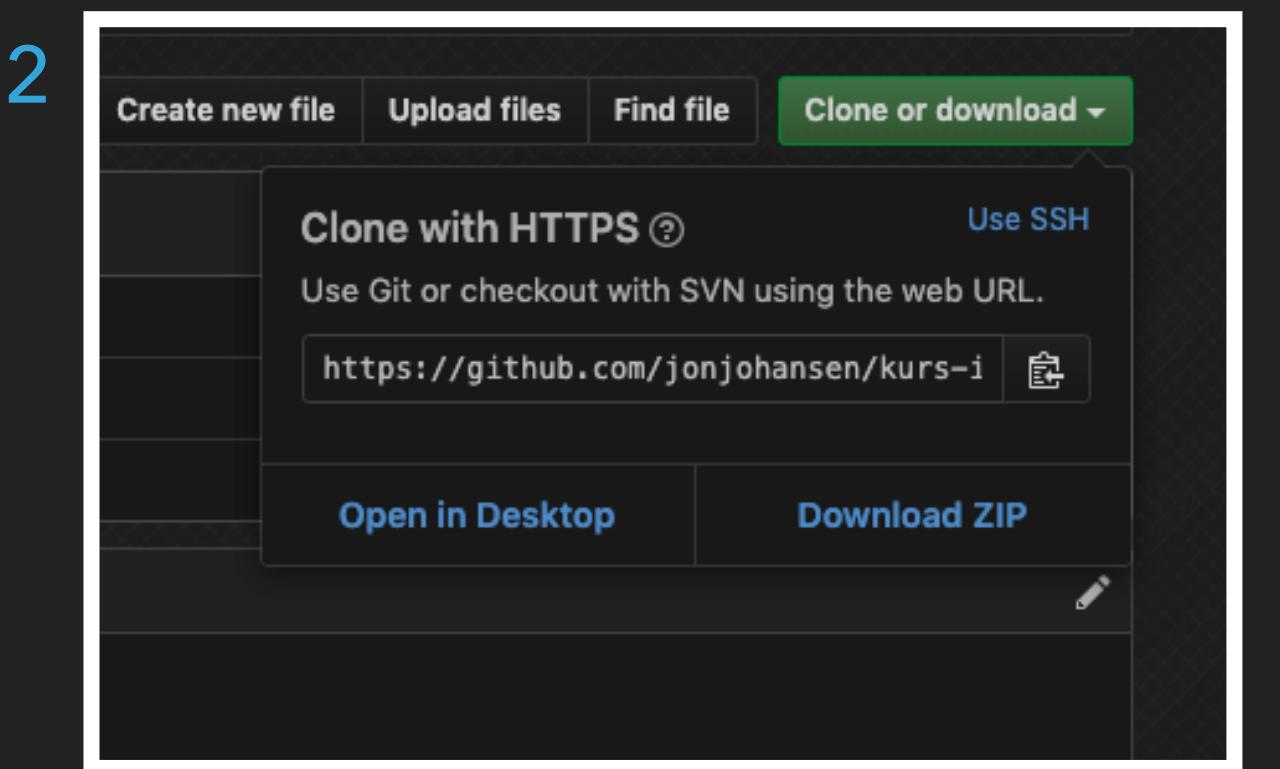
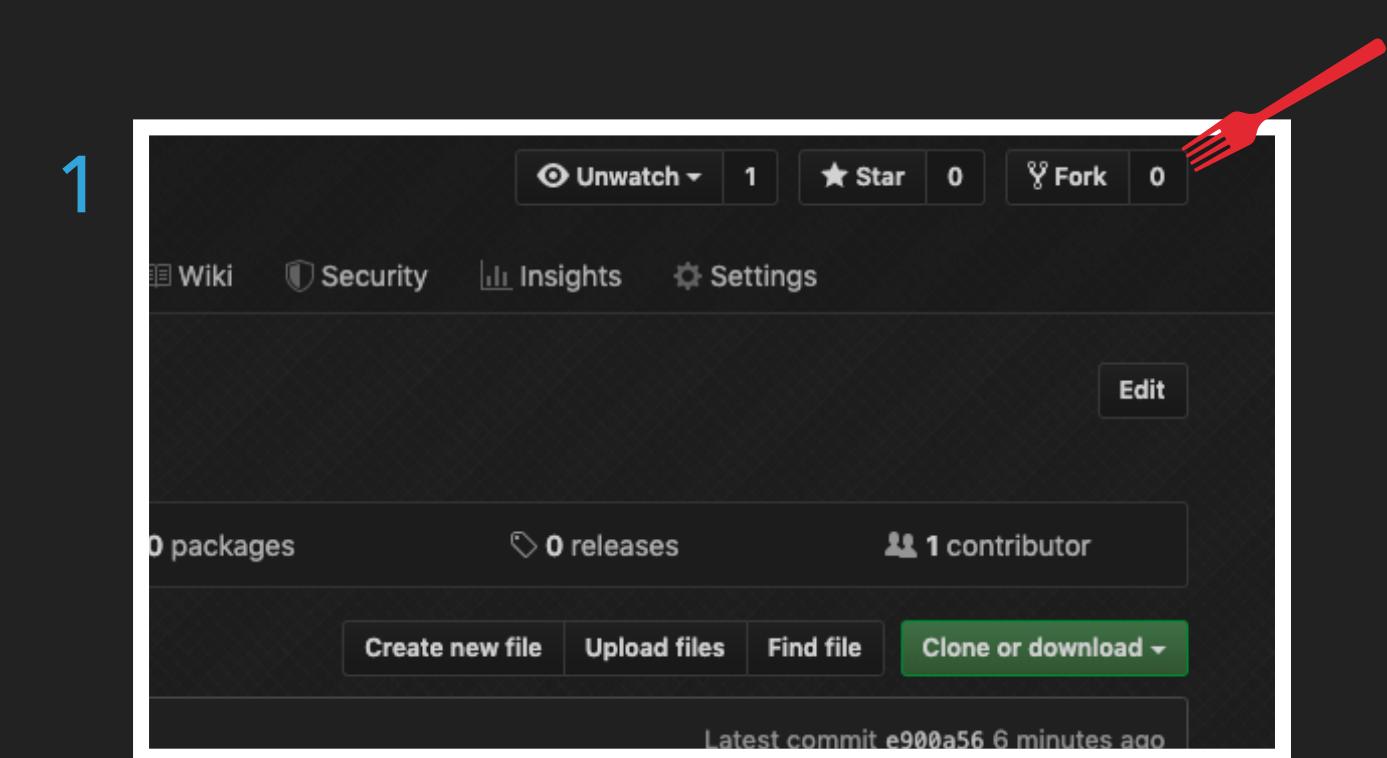




LA OSS PRØVE LITT

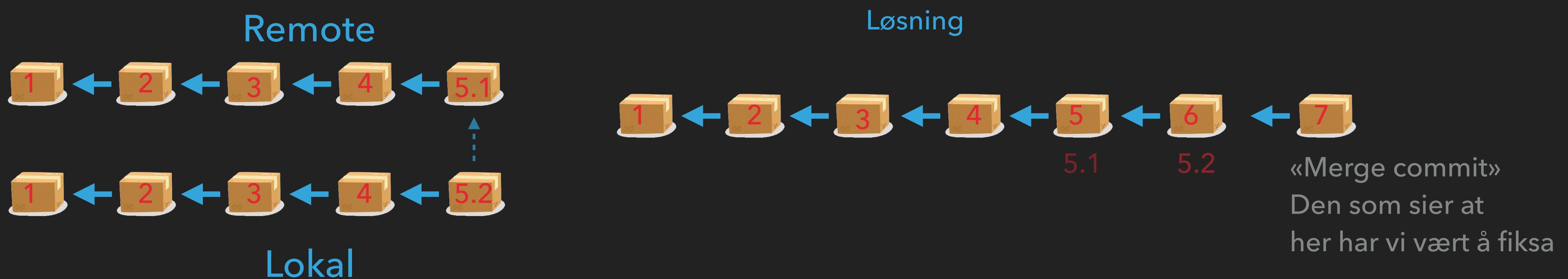
KURS I VERSJONSKONTROLL

- ▶ Logg inn på www.github.com
- ▶ Gå til github.com/jonjohansen/kurs-i-versjonskontroll
- ▶ Trykk på fork knappen. 1
Velg brukernavnet ditt.
- ▶ Prøv å «git clone» din nye «fork». 2
- ▶ Gå inn i mappen som git clone lager (kurs-i-versjonskontroll)
- ▶ Skriv git status



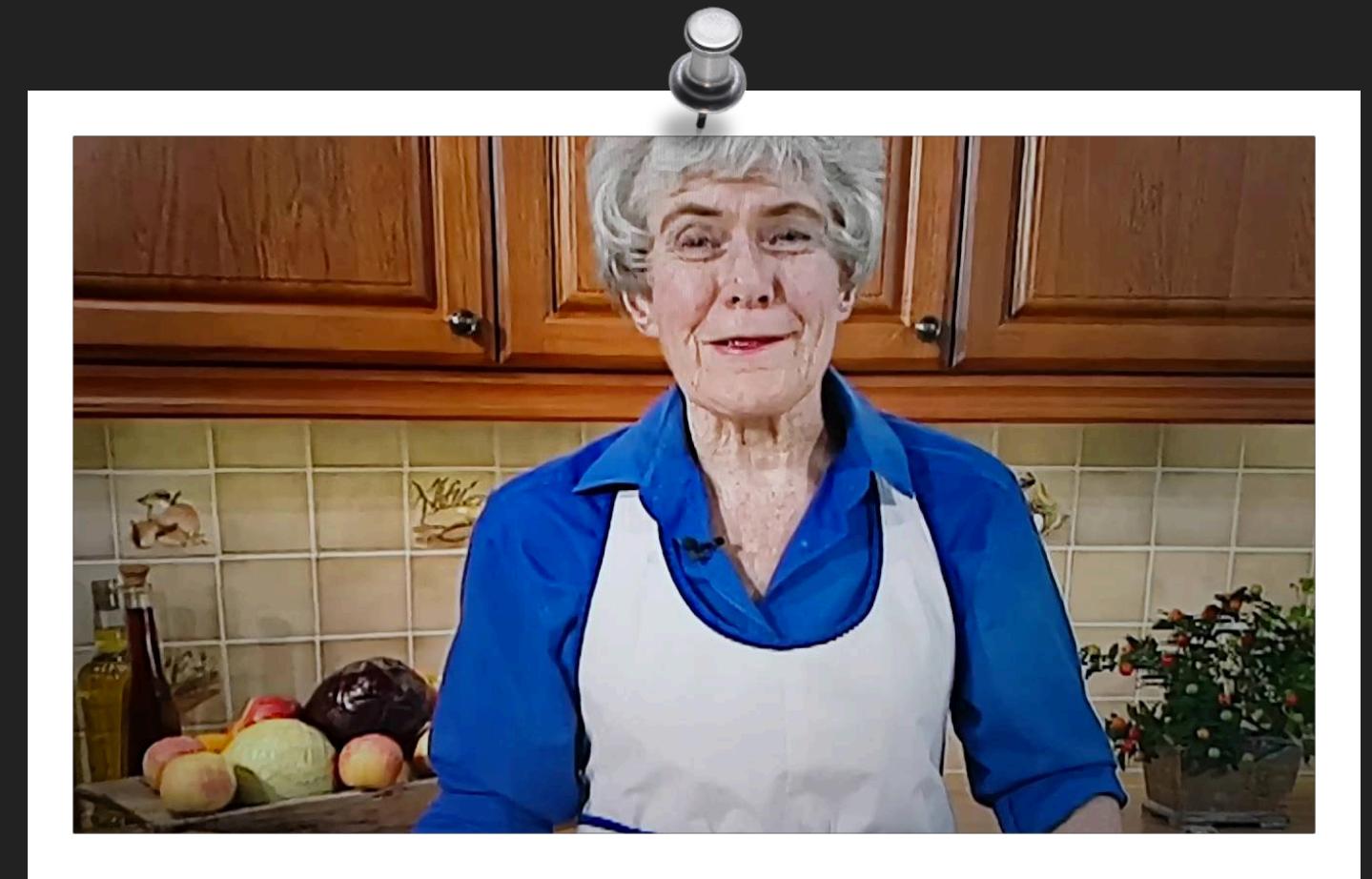
- ▶ Åpne «endre-meg.txt i
- ▶ Endre linje nr. 3
- ▶ Test git status nå
- ▶ git add endre-meg.txt
- ▶ Kjør git status!!!
- ▶ git commit med en fin melding (vi får lov å bruke git commit -m «Fin melding»)
- ▶ git status igjen!
- ▶ Prøv å pushe den opp til Github! (git push)

- ▶ Typisk problem:
Vi har startet å jobbe uten å ta inn endringer fra remote.
- ▶ Dette skjer når:
 - ▶ Noen har pusha noe på remote mens vi har jobbet (sammarbeid)
 - ▶ Du har pusha fra lab-maskin, og glemt å pulle fra hjemme-pcen



KURS I VERSJONSKONTROLL

- ▶ La oss prøve det!
- ▶ Sjekk git log
- ▶ Så jukser vi litt, og skriver
git pull <https://github.com/jonjohansen/kurs-i-versjonskontroll.git>





CONFLICT

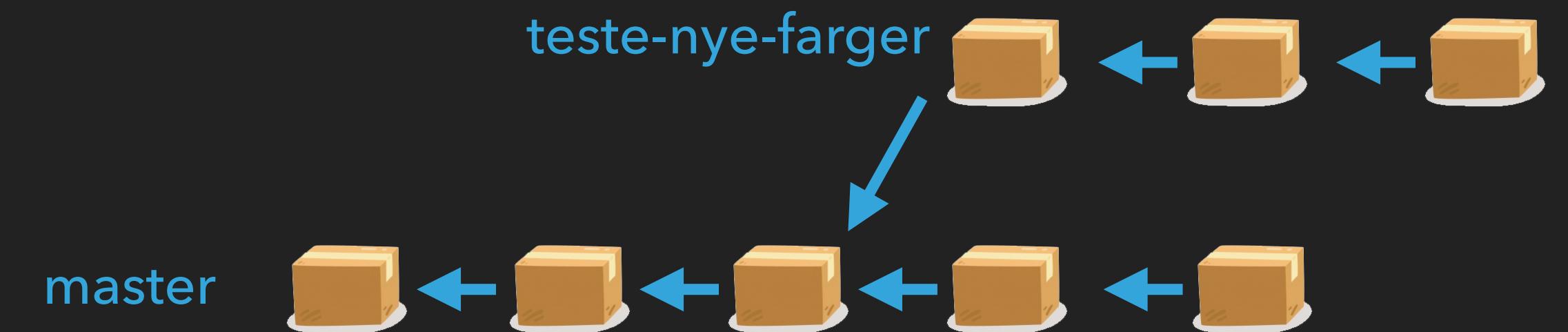
- ▶ Du får beskjed om at nå har du fått en **merge conflict!**
- ▶ Git status - Sjekk hvor **conflicten** er
(Legg merke til at det står at du har en **merge conflict** i statusvinduet)
- ▶ Åpne filen(e) som har **conflicten** (endre-meg.txt)
- ▶ Fiks **conflicten**
- ▶ Skriv git status igjen
- ▶ Legg til løsningen din (**git add endre-meg.txt**)
- ▶ Commit løsningen din
- ▶ Push løsningen din
- ▶ 👍 Sjekk github/git log å se hvordan det ser ut

MEN HVA MED BRANCHES?

Fredrik

BRANCHES!

- ▶ Flere esker peker mot samme
- ▶ Hovedbranchen heter som regel «master»
- ▶ Man må «merge» hvis man skal få endringene fra en branch til en annen
- ▶ Mer om dette i oppfølgerkurset neset semester (eller så kan du undersøke på nettet)
<https://learngitbranching.js.org>)



Kommandoer

- ▶ Ny branch: `git branch <NAME>`
- ▶ Skifte branch: `git checkout <NAME>`
- ▶ Begge: `git checkout -b <NAME>`
- ▶ Merge: `git merge <NAME>`

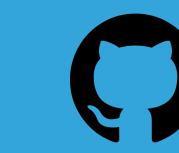
AVSLUTNINGSVIS

- ▶ Øvelse gjør mester
- ▶ Det kommer til å skje feil, løs de! Lær å løse de!
- ▶ Git kan brukes enkelt, og det kan brukes avansert
- ▶ Hjelp hverandre, spør hverandre!
- ▶ Google!!
- ▶ Ta ting steg for steg, bli komfortabel med verktøyet
- ▶ Mange fine ressurser for å lære git
<https://td-org-uit-no.github.io/git-cheatsheet/>





JON H L JOHANSEN



[GITHUB.COM/JONJOHANSEN](https://github.com/jonjohansen)



[JON#9452](#)