CHALMERS

UNIVERSITY OF TECHNOLOGY

Git Tutorial

Filip Buric

January 2018

sys²bio
www.sysbio.se

Overview

- version control
- basic git   (command line)
- exercises!

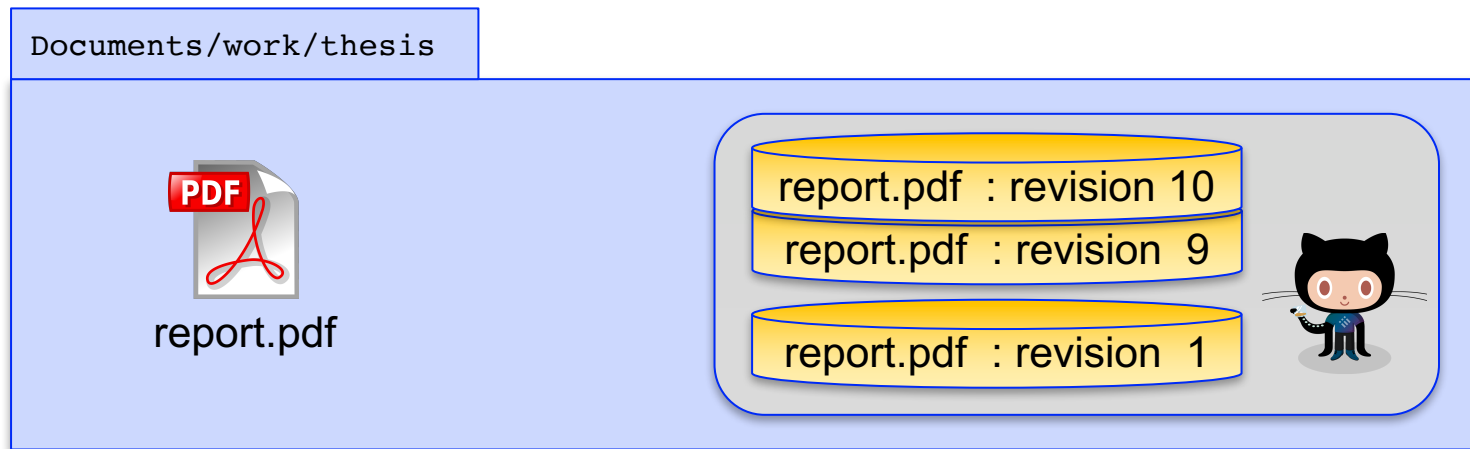
- Ask whenever confused

Version Control

- **Issue**: Files with long, complicated history.
  Want to keep different versions:

  > `Report_v3_comments_2018_01_05.docx`
  > `experiment_pipeline_10_2017_11_05.sh`

- **Compound issue:** Other people work on them too

- Programs like **git** (version control systems) keep track of changes made by different people
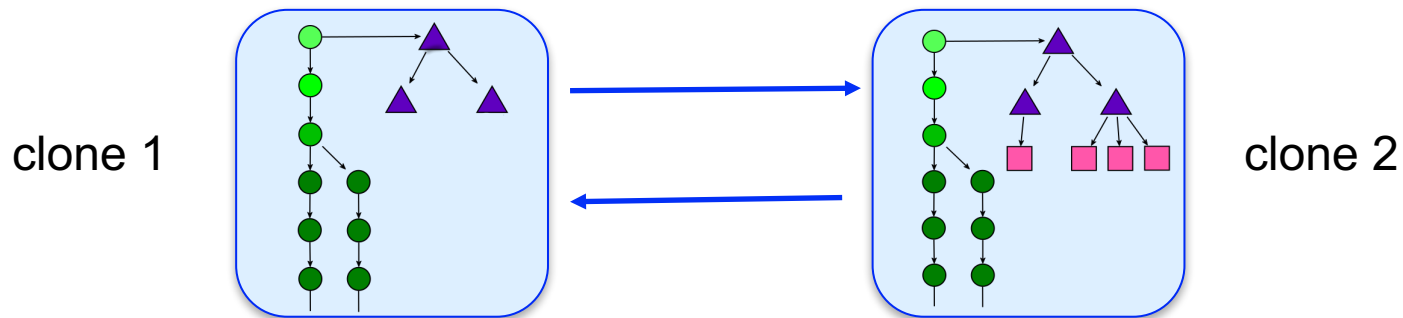
Git Concepts

- a git project is called a **repository** or **repo** = directory with history
- a repo contains a collection of snapshots (called **revisions**) of the directory:

Documents/work/thesis

report.pdf

report.pdf : revision 10
report.pdf : revision 9
report.pdf : revision 1

Git Concepts

- revisions are connected in ***branches***, reflecting file evolution

clone 1 clone 2

- repos are *decentralized*
  - Each ***clone*** contains everything (all revisions + history)
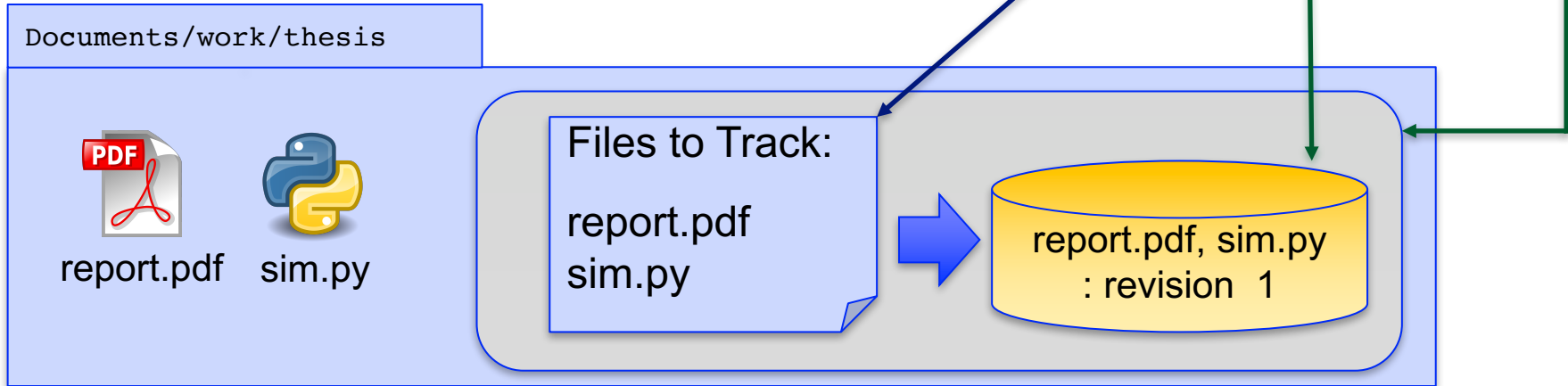 - Changes can be passed between clones

Creating a Repo and Recording Changes

0) Initializing the repo inside your project directory   `git init`

1) Instruct git to start tracking files   `git add`

2) Commit list of files-to-track into a revision   `git commit`
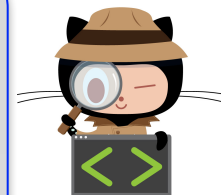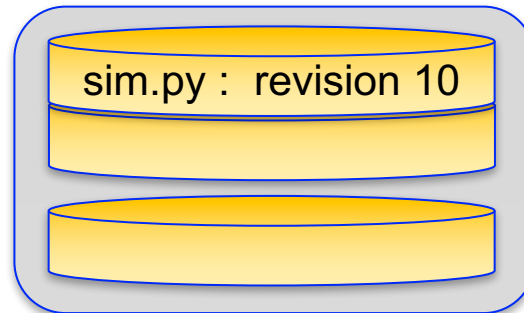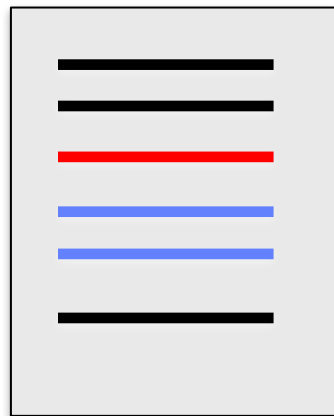
Exercise 1

- **10** minutes

- Go to   `goo.gl/QVrbJp`

- Notes are good-to-know info only

Making and Committing Changes

- Git reports what changed since latest revision: **git status**
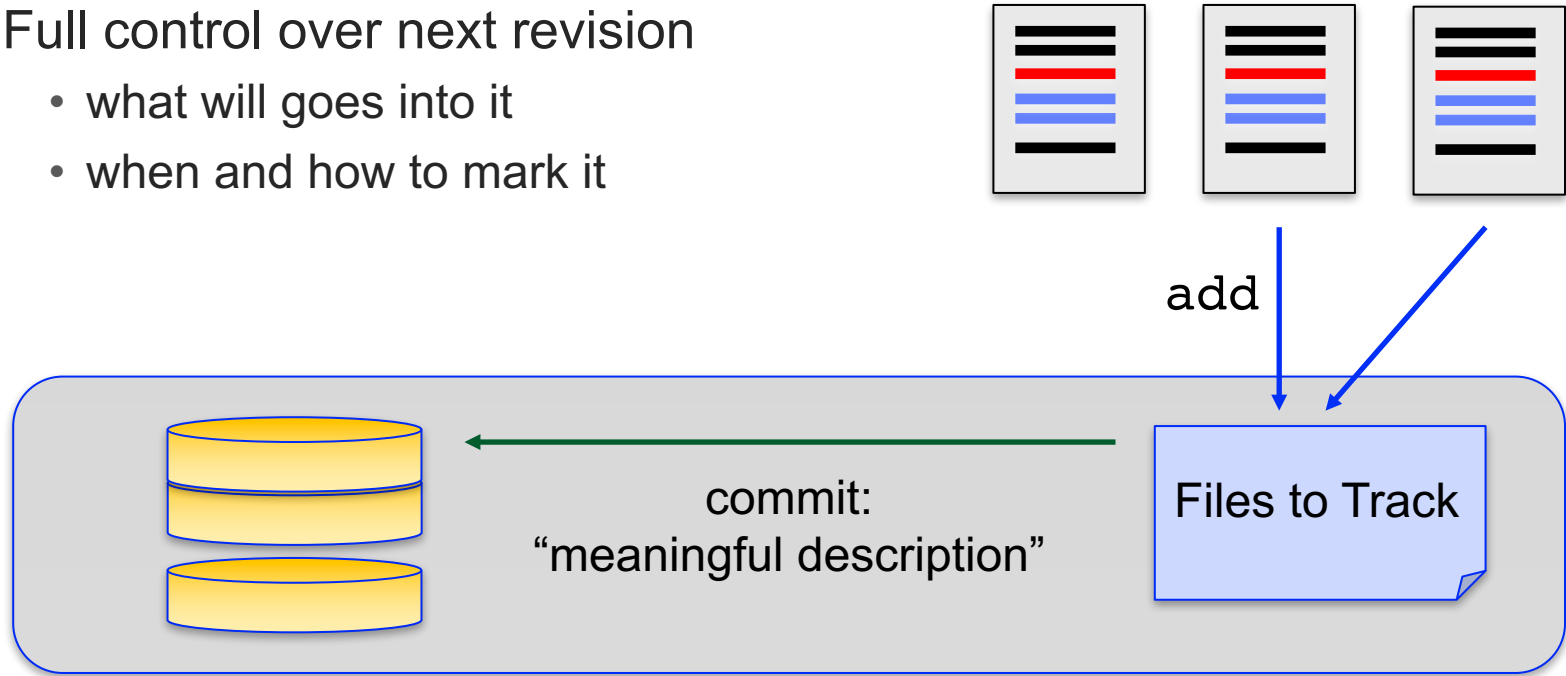- Differences can be inspected: **git diff**
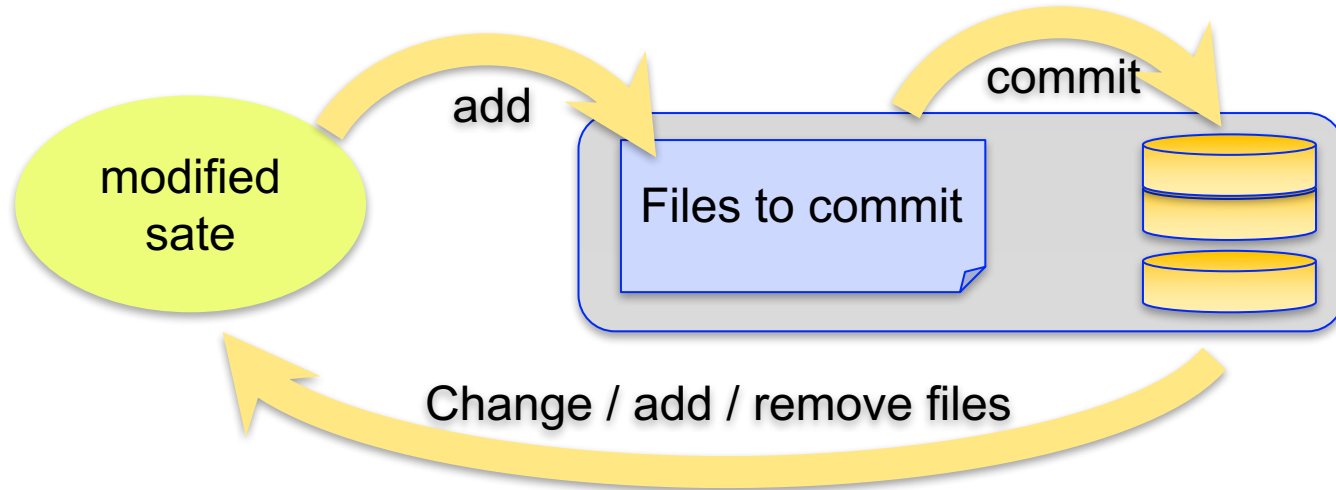
(current file)

sim.py

sim.py : revision 10

Making and Committing Changes

- Full control over next revision
  - what will goes into it
  - when and how to mark it

add

commit:
"meaningful description"

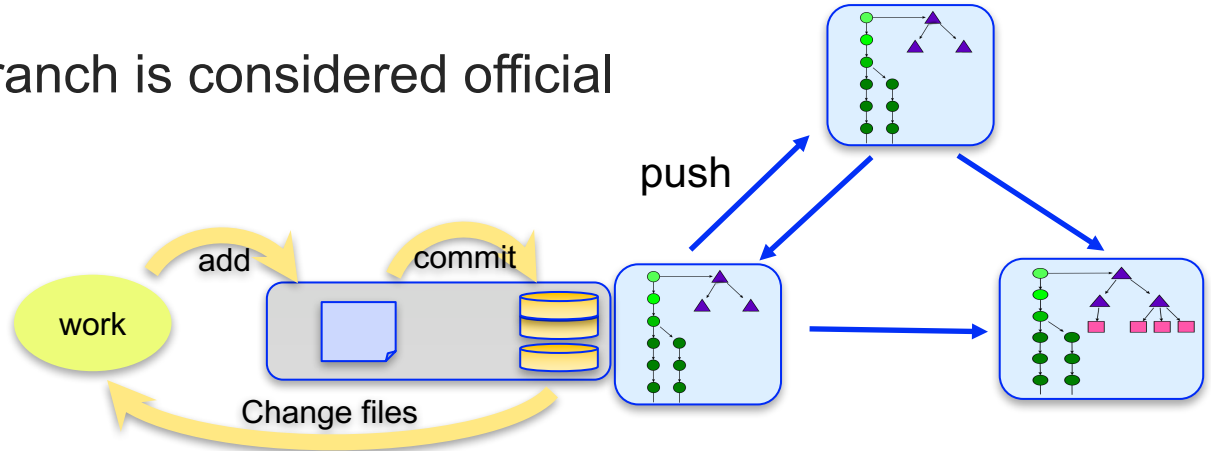Files to Track

Typical Work Loop

Exercise 2

- **15** minutes
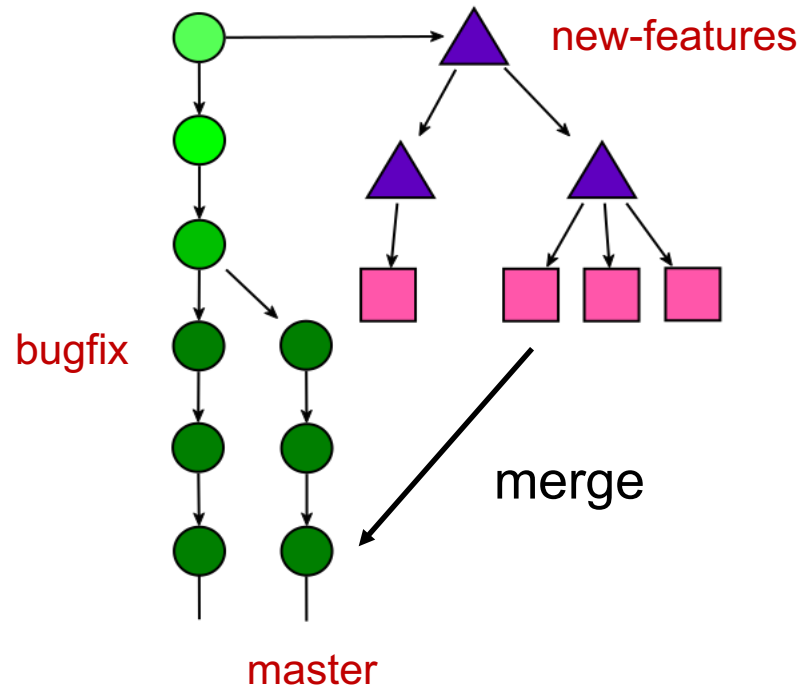
Collaborating

- Convention:

  one repo and one branch is considered official

- Collaborators:
  - clone from this repo
  - work
  - **push** their contributions to it

Collaborating

- Work usually done on branches:

    - maintain separation of interest
        (e.g. "development" vs "bug fixing")

    - isolate changes
        (e.g. "experimental" branch)

new-features

bugfix

merge

master

Exercise 3

- **10** minutes

Wrap-up

- Do use git to track your work – even if working alone

- Don't be afraid to break things!   Almost always possible to recover.

- Complex tool but daily routine involves only a handful of commands

Thank you!