Git crash course

Thomas Breuer

Lehrstuhl für Algebra und Zahlentheorie, RWTH Aachen University, Germany

Summer School, September 06, 2021

Aim of this talk

- Version control systems: Why do we use them?
- How can we use the version control system `git` locally?
- How do we use `git` with the GitHub services,
  for the development of OSCAR
  (and for this summer school)?
- This is all very basic (but still looks complicated).

Version control systems – why?

- (for software development, writing papers, . . .)
- record the history of changes over time
- for each revision (timestamp):
  Who changed **what**, **when**, and **why**?
- when working alone:
  no danger to lose parts of the work (due to the backups),
  undo changes
- when working with others:
  “merge” contributions from different sides into one source
- “branch”: maintain different versions of one software
- if public: admit contributions from outside

Git

. . . is a version control system,

. . . is free,

. . . does not need a central server (is “decentralized”),
users’ repositories have the full history information,
the different repositories are equally good,

. . . has GUIs, but here we show just the command line version,

. . . calls the revisions/snapshots **commits**.

A live session with git

(commands in order of appearance)

`git init`,
`git status`,
`git add <<files>>`,
`git commit`,
`git log`,
`git diff`,
`git diff --cached`,
`git diff <<old>> <<new>>`,
`git branch`,
`git checkout <<branch>>`,
`git merge <<branch>>`,
`git stash`,
`git stash pop`.

GitHub: Use remote repositories

Up to now, we have used git on the local computer only.

For collaborative work,
we use also public remote repositories,
hosted by GitHub.com (`https://github.com`).
(There are alternatives; we use GitHub.)

The services are free and open.

The code is in git repositories.

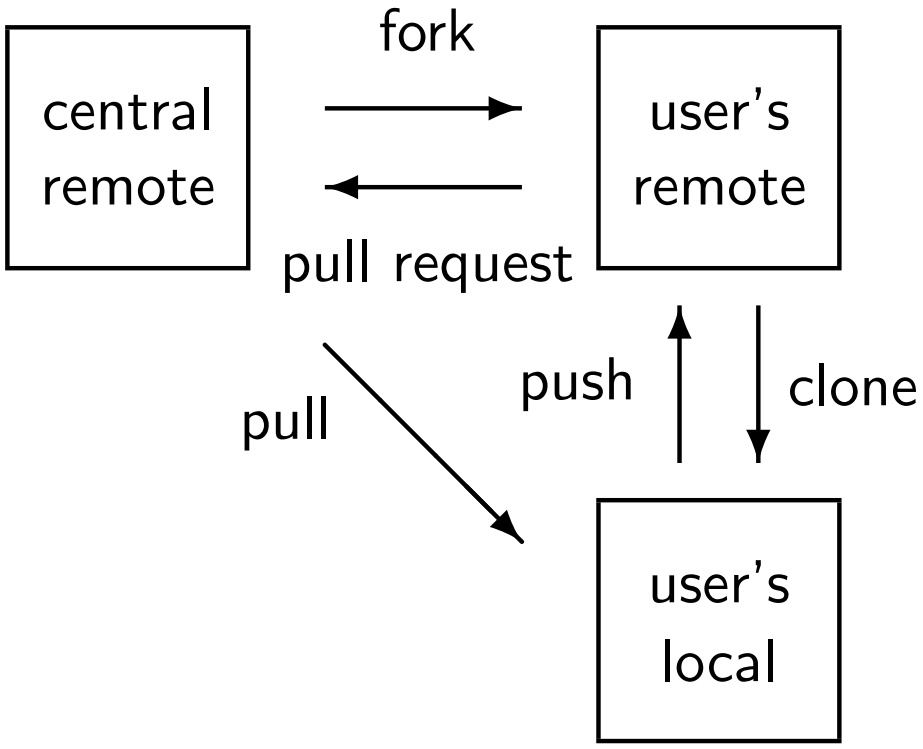
We use branching and merging (in principle ...).

We interact with GitHub via the command line and via a web interface.

Proposed GitHub workflow

Do not directly merge your changes into the (central) remote repository.

Create a so-called **pull request** from your branch,
which then can be discussed, improved, discussed, . . . ,
and eventually merged.



Proposed GitHub workflow: Setup

- Make sure that you have a GitHub account, and that local user name and e-mail address fit to the values of the account.
  Check with `git config -l`,
  set with `git config user.name "..."` and
  `git config user.email "..."`.
- Create/take a central remote repository. (Here:
  `https://github.com/oscar-system/Summerschool21Exercises.jl`)
- Create your own remote copy via `fork`.
  (Click the button in the web page of the repository.)
- Create your own local copy via `clone`.
  (Copy the URL under `Code/Clone` in the web page,
  then execute `git clone` with this URL on your computer.)
- Notify the central remote repository. On your computer, call:
  `git remote add upstream <<url-central>>`.
  (Check with `git remote -v`.)
  Now `git pull -r upstream main` should work.

Proposed Github workflow: Repeat

- In your local repository,
  - create a new branch (`git checkout -b <<name>>`),
  - edit some files,
  - stage the changed files (`git add`),
  - commit the changes (`git commit`),
  - incorporate changes in the remote `main` branch
    since you started editing (`git pull -r upstream main`;
    `-r` means "rebase": rewrite the history such that
    the remote changes come first and the local changes come on top),
  - commit the updated version,
  - and then push the new version to `origin` (`git push`;
    git will propose additional parameters).
- In the web page of the central remote repository,
  GitHub will propose to create a pull request.
  Check that these are the changes you want to propose;
  if not then go to the previous step.
  Edit the description if necessary.
  Finally, create the pull request.

Proposed workflow: Repeat

- Wait for comments (reviews).
  It may be necessary to get approvals from others
  before the pull request can be merged.
- After merging, update `main`
  (`git checkout main; git pull -r upstream main`,
  and delete the local branch (`git branch -D <<name>>`).

Some links

git documentation

- `man git`, `man giteveryday`, `man gittutorial`
- Reference Manual: `https://git-scm.com/doc`
- Cheat sheet: `https://education.github.com/git-cheat-sheet-education.pdf`

GitHub documentation

- Documentation: `https://docs.github.com`
- Tutorial: `https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners`

Markdown syntax

- Documentation: `https://guides.github.com/features/mastering-markdown/`
- Cheat sheet: `https://guides.github.com/pdfs/markdown-cheatsheet-online.pdf`