GitHub workflow

Paula Preuß, Pierre-Francois Duc

20.05.2025

REINER LEMOINE INSTITUT

Why should you care?

REINER LEMOINE INSTITUT

1. Help make changes in the code more transparent (important for open source!)
2. Reviewing helps to avoid bugs
3. Link issues to resolution

Agenda

REINER LEMOINE INSTITUT

1. Branch out
2. Committing changes
3. Creating and reviewing PRs
4. Merge conflicts
5. Issue linking
6. Additional ressources

REINER LEMOINE INSTITUT

0. The very basics

What you will need

REINER LEMOINE INSTITUT

- Install git for Windows https://git-scm.com/download/
- Install PyCharm (or another IDE) https://www.jetbrains.com/pycharm/download/
- Create a GitHub account

Cloning a remote repository

REINER LEMOINE INSTITUT

- open a terminal with **`git`** (you can use **`git bash`** if on windows)
- **`cd`** to navigate to directory where the repository should be
- copy github HTTPS address
- `git clone <address>`
- open the project in your IDE
- follow any additional steps, like setting up your virtual environment

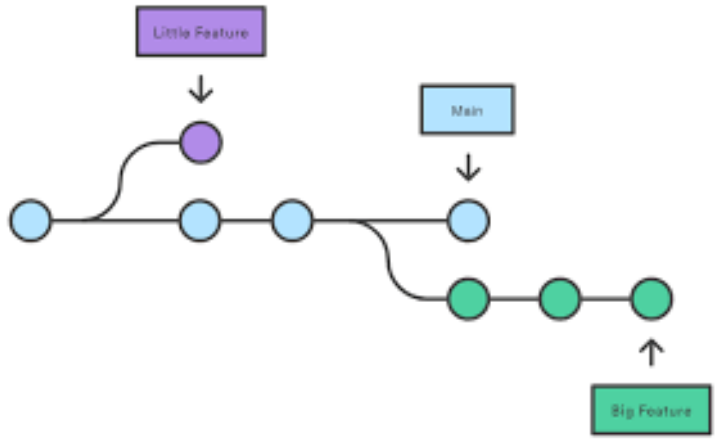
REINER
LEMOINE
INSTITUT

1. Branches

Working on a repository

REINER LEMOINE INSTITUT

- never work directly on the main branch -> fork off a branch and **merge** back in
- `git checkout main`
- `git pull`
- `git checkout -b <name>` -> new branch
- there are some conventions like `feature/<name>` or `fix/<name>`



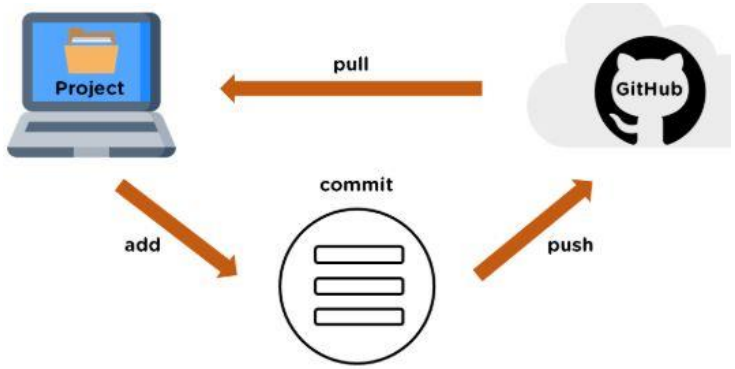
REINER LEMOINE INSTITUT

2. Committing changes

Making commits

REINER LEMOINE INSTITUT

- commits reflect a **set of changes** to one or more files
- go together with a **commit message**
- better to err on the small side -> **commit often**
- they are local changes until they are pushed
- once they are pushed,
  - other people can pull them
  - you have a backup of your work



REINER LEMOINE INSTITUT

3. Creating and reviewing PRs

Preparing for review: Rebasing

REINER LEMOINE INSTITUT

- you'd like to leave a neat PR for your colleague to review
- no pressure to have a super neat history from the beginning
- `git log` – check commits made to the branch
- `git rebase -i main` – move branch on top of other branch and edit commits in branch

! `git push -f` when rewriting commit history

Pro tip: `git config --global rebase.autosquash true` to automatically sort commits for interactive rebase

Preparing for review: Resetting

REINER LEMOINE INSTITUT

- motivation: you'd like to leave a neat PR for your colleague to review
- no pressure to have a super neat history from the beginning
- `git log` – check commits made to the branch
- `git reset --soft <commit>` – undo changes until specified commit but keep changes made to files
- make new, meaningful commits

- `git stash` / `git stash pop` – put away local changes temporarily / get them back

Merging changes into the code

REINER LEMOINE INSTITUT

- create pull request (PR)
- can be reviewed commit by commit or at once
- merge into main branch (usually)

→ pull main branch so that new branches reflect the changes

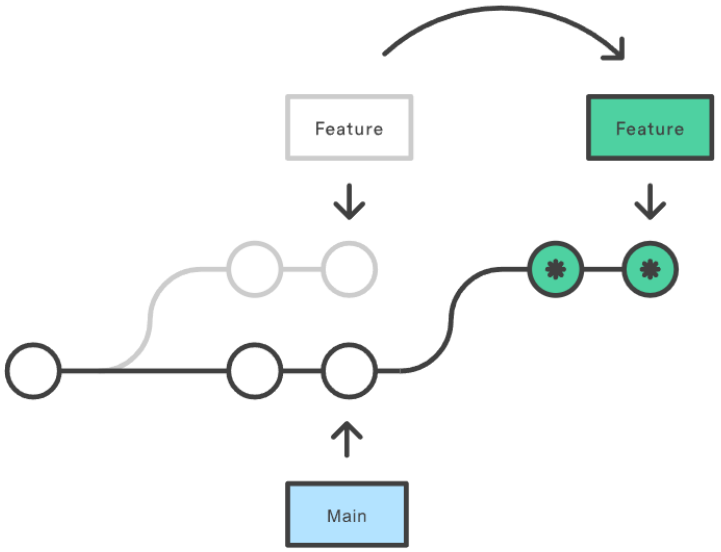
REINER LEMOINE INSTITUT

4. Merge conflicts

Dealing with merge conflicts

REINER LEMOINE INSTITUT

- open a PR to merge your changes
- if you have modified a file that has also been modified on main, you will run into merge conflicts
- `git rebase (-i) main` -> move branch on top of current state of main
- the rebase will stop on any merge conflicts and wait for them to be resolved
- `git rebase --continue`



REINER LEMOINE INSTITUT

5. Issue linking

Issues

REINER LEMOINE INSTITUT

- todos can be managed over git issues
- can be linked to pull requests to track development
- can be automatically closed using magic words like *fix*, *solve*, *close*
- templates can be used to create issues

REINER LEMOINE INSTITUT

6. Templates, GitHub actions, pre-commit and more

RLI Super-repo

REINER LEMOINE INSTITUT

- Git - Super-Repo Documentation
- templates for issues, PRs
- pre-commit setup
- GitHub actions
- etc...

Licence

Please quote as:
"GitHub workflow basics (2025-05-20)"

Paula Preuß

E-Mail: paula.preuss@rl-institut.de

Web: http://www.rl-institut.de