HPC Café – Git in Practice

Collaborations, Workflow, Continuous Integration

THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

xkcd.com/1597/

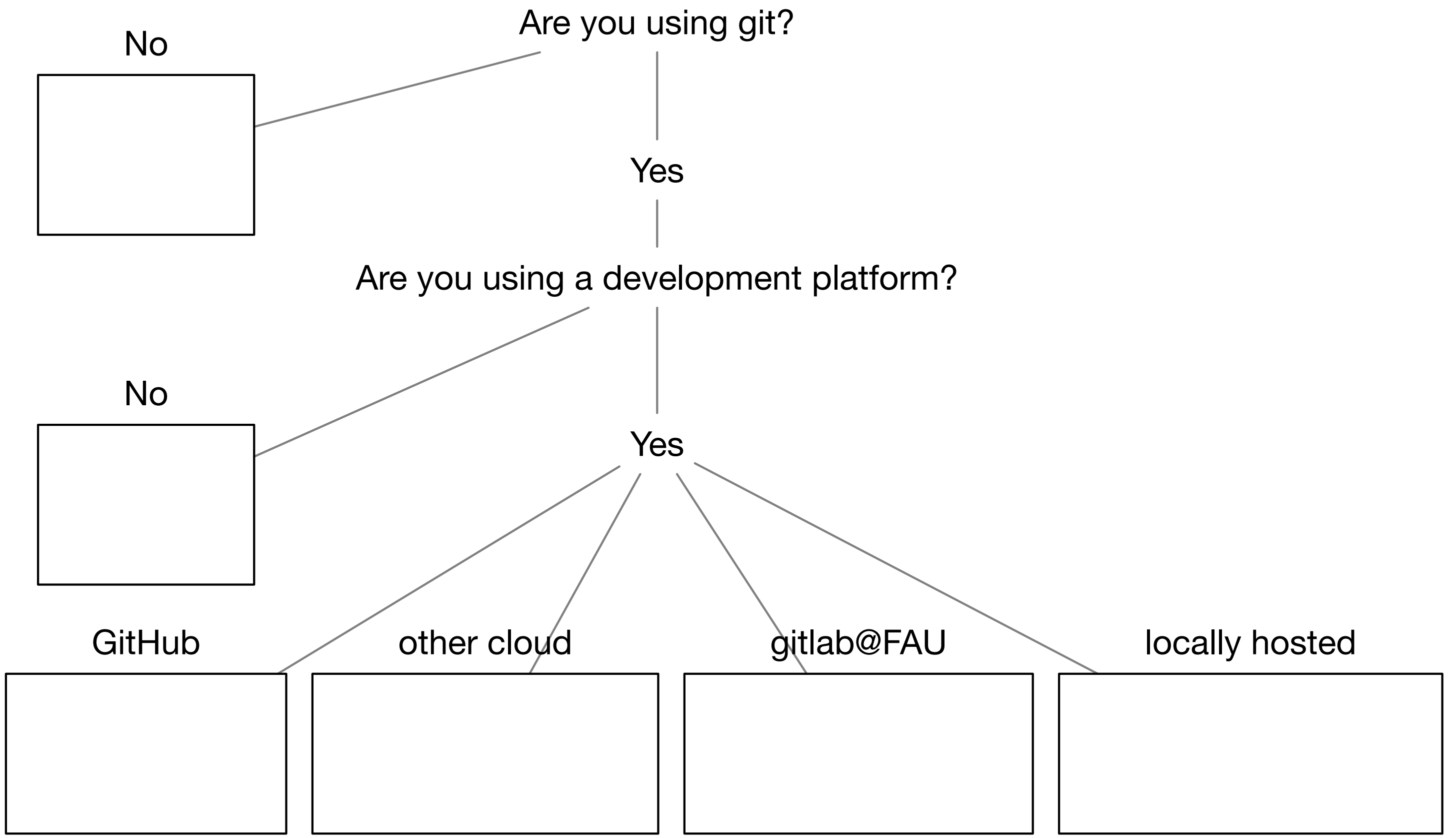
Julian Hammer, Thomas Gruber, 2021-03-09

Goal

- Explain and show best practices for (collaborative) development
- Motivate use of git and development platforms
- Motivate to go open source and contribute yourself
- Highlight solutions to everyday problems

- Practical/”how to” oriented
- Feel free to ask background **questions**, at **any time**

Survey


Are you using git?
No
Yes
Are you using a development platform?
No
Yes
GitHub
other cloud
gitlab@FAU
locally hosted

Practicing and Understanding Git

ohmygit.org

- Game to learn git
- Teaches fundamentals concepts as well as commands
- Recommendation also to intermediate git users

Overview

Git Collaboration Platforms – Why?

- Hosts your repository and metadata
  Backups* and sharing made easy
- Web front-end
  Easy access to git and non-git users
- User management
  Permissions, groups, collaborators
- Issue tracker
- Documentation tools
  e.g., a wiki
- Integration & deployment tools

All of this is just one click away…

* do NOT rely on this!

Git Collaboration Platforms – Which to use?

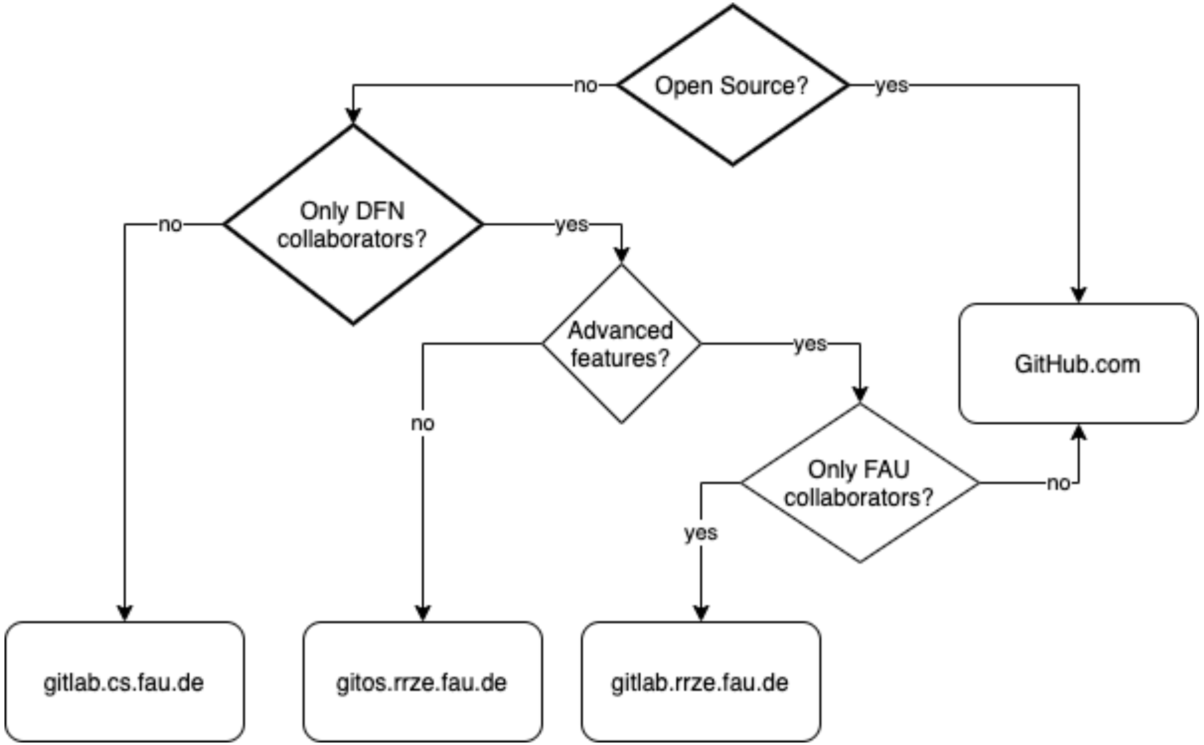
GitHub.com

- de-facto standard for open source projects
- private repositories have limited features
- owned by Microsoft

GitLabs hosted at FAU:

- gitlab.cs.fau.de
  - collaboration with anyone
- gitos.rrze.fau.de
  - collaborate with DFN users
- gitlab.rrze.fau.de (enterprise features)
  - collaborate with FAU users
  - account request via idm.fau.de

Git Collaboration Platforms – Which to use?



Platform and Repository Access

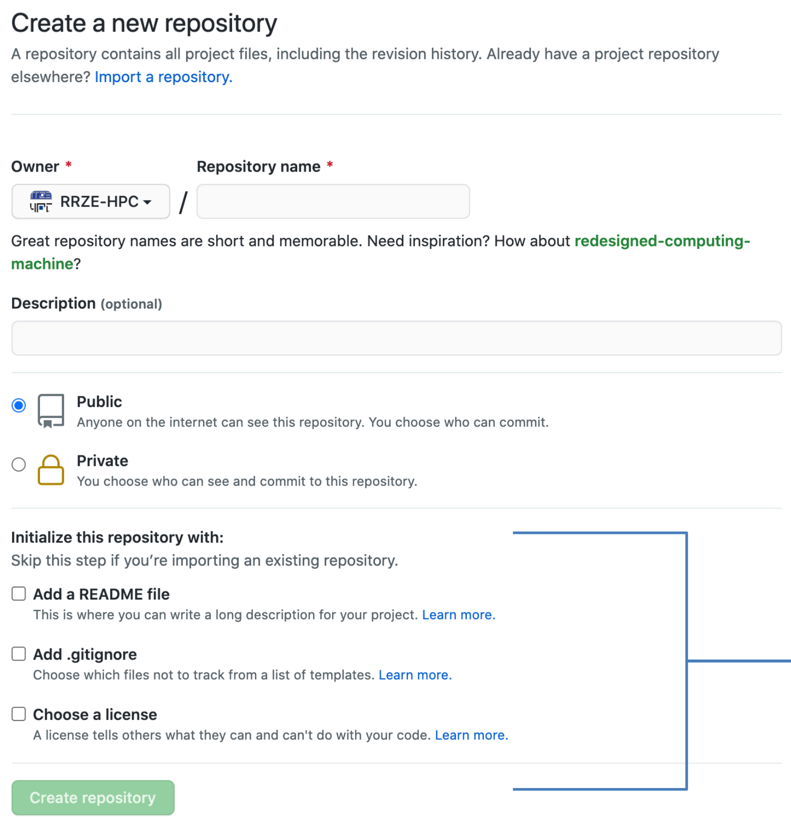
- To web platforms: login via website
  consider two-factor-authentication (2FA)

- Repository access: SSH key-pair
  - Generate SSH key file
    `ssh-keygen -t ed25519`
  - Upload public key to platform
    e.g., ~/.ssh/id_ed25519.pub
    User Profile → Settings → SSH Keys → Add key

See HPC Café – Secure System Access (2020-06-09) for details:
https://hpc.fau.de/files/2020/06/2020-06-09-hpc-cafe-security.pdf
https://www.video.uni-erlangen.de/clip/id/17820

New Project

github.com/new



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * Repository name *

RRZE-HPC /

Great repository names are short and memorable. Need inspiration? How about redesigned-computing-machine?

Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

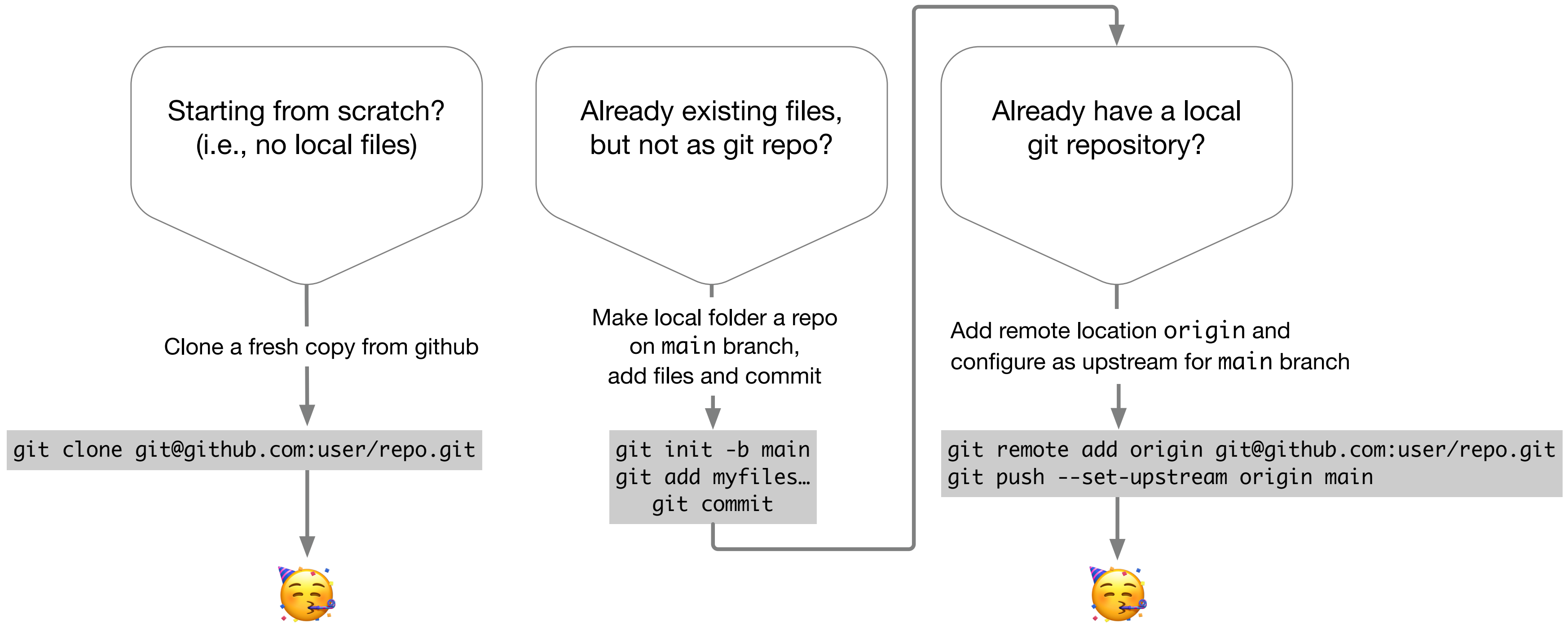
☐ Add a README file
This is where you can write a long description for your project. Learn more.

☐ Add .gitignore
Choose which files not to track from a list of templates. Learn more.

☐ Choose a license
A license tells others what they can and can't do with your code. Learn more.

we'll get to it

Setup Local Repository


Starting from scratch? (i.e., no local files)
Clone a fresh copy from github
Already existing files, but not as git repo?
Make local folder a repo on main branch, add files and commit
Already have a local git repository?
Add remote location origin and configure as upstream for main branch

```
git clone git@github.com:user/repo.git
```

```
git init -b main
git add myfiles…
git commit
```

```
git remote add origin git@github.com:user/repo.git
git push --set-upstream origin main
```

Initialize Repository

Useful files to have:

- `README.md`
  Say what this it’s about, how to use and interact with it.

- `LICENSE.md`
  Tell others how to treat your code.
  https://choosealicense.com/

- `.gitignore`
  Ignoring unnecessary files, makes life simpler.
  https://gitignore.io

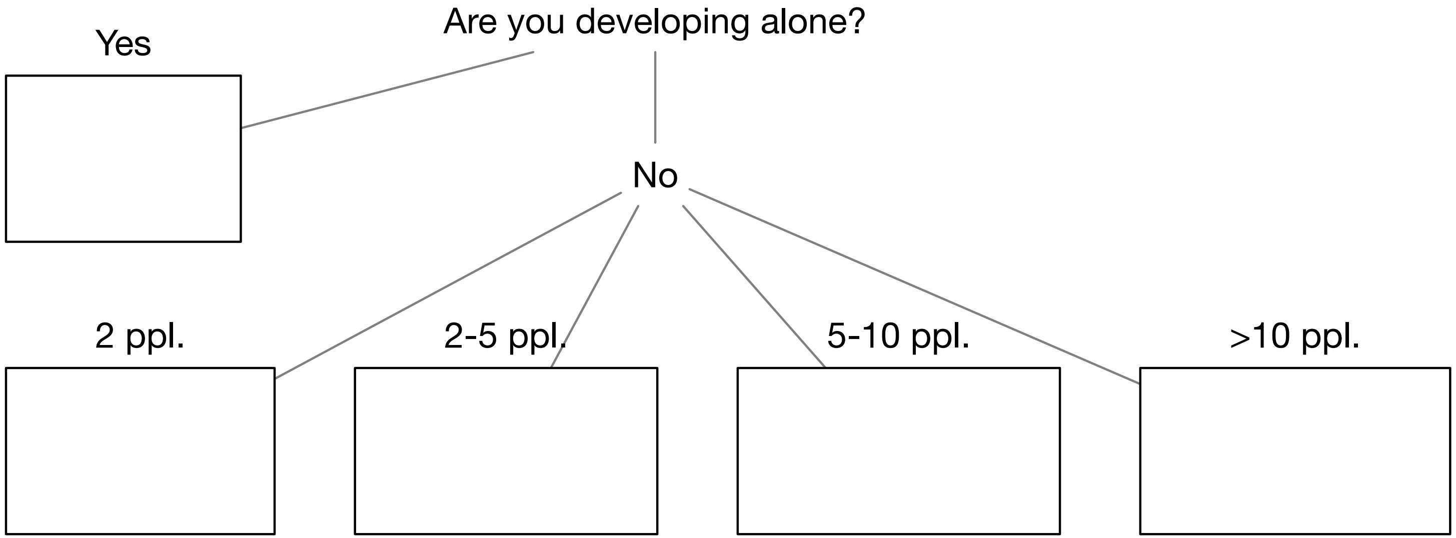
! Don‘t forget to `git add`, `commit` and `push`.

Working Together

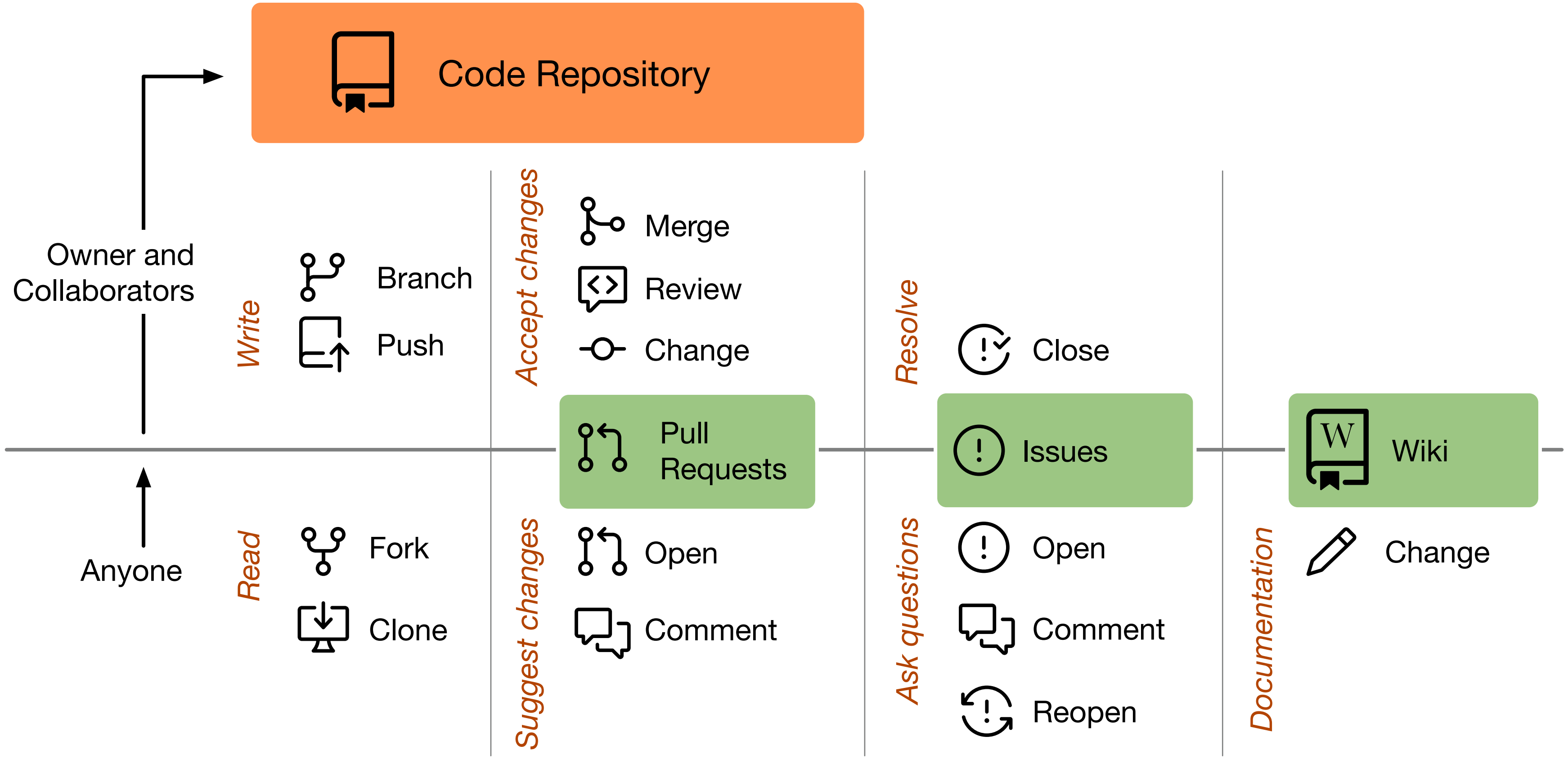
I am a lazy person, which is why I like open source,
for other people to do work for me.
[Linus Torvalds]

I wish…
[me]

Survey



Working Together: Interactions



Organizations and Collaborators

Organizations

- Shared ownership
- Default permissions for org repos
- Teams
- Public Face

➢ Use for research groups and institutions

github.com/organizations/new

Collaborators

- Individual permissions

➢ Use for external collaborators

Working Together: Issues

Ticketing system

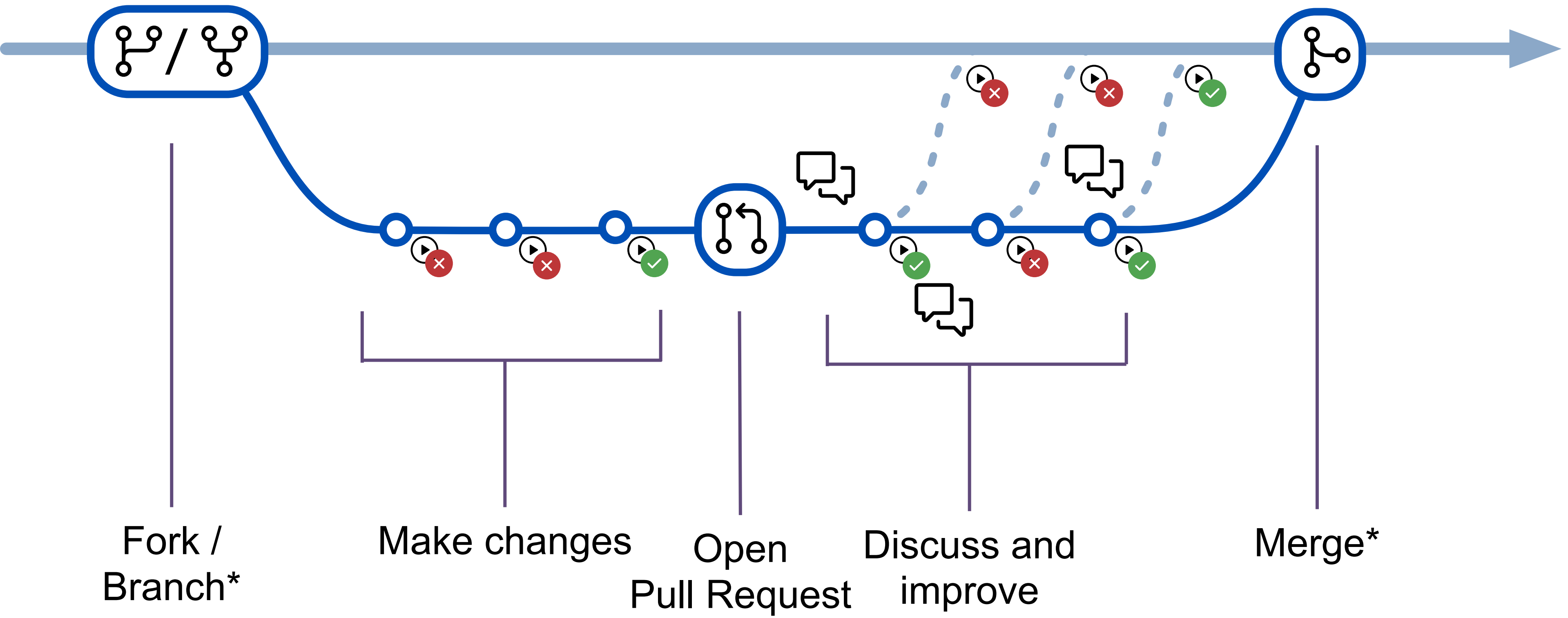
Usually first contact between developers and users.

➔ Used for support, questions, bugs, feature requests, discussions…

Useful:

- References to commits (`commithash`) and other issues/PRs (`#number`)
- Mentions (`@username`)
- Check lists (`- [ ] item`)
- Tags (e.g., bug, feature request, support)
- Markdown, templates…

Working Together: Workflow



* Only with write permissions

Continuous Integration

Test *continuously*

- Build: Does it compile?
- Unit Tests:
  Produces correct results?
- Coverage:
  Are more tests needed?
- Lint: Is code “well written”?
- Matrix builds: Compatibility

(Continuous Deployment)

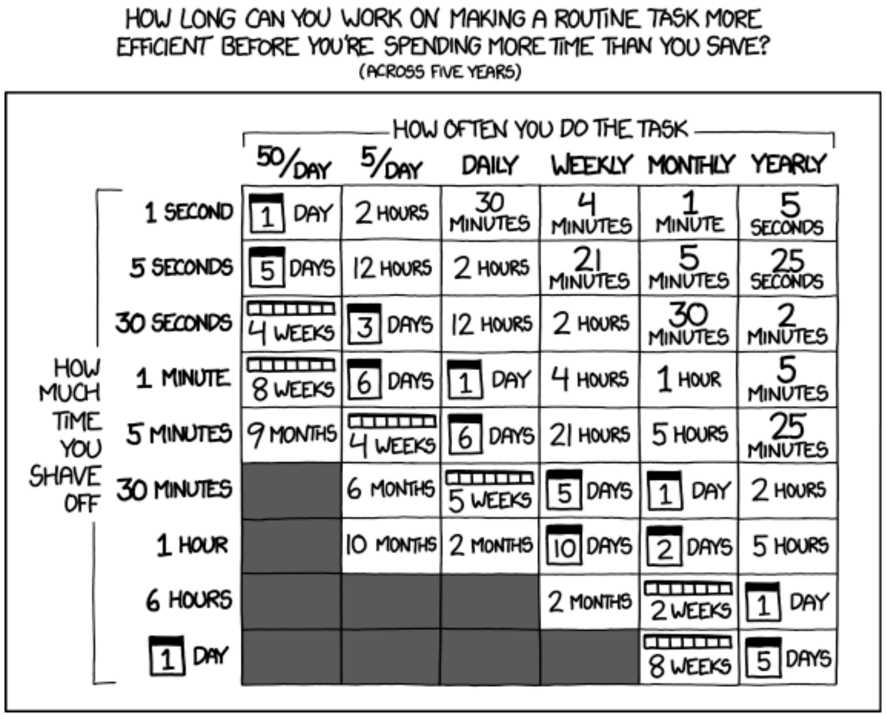
- Upload releases
- Deploy to production

Usually ON EVERY PUSH

You ask WHY???

- It’s “free”
- Find bugs earlier
- Encourages test-driven development (write test before code and fore every bug found)
- Find regressions (reintroduction of already fixed bugs)
- Helps contributors get engaged
- Standardized environment

Test? Why Bother?



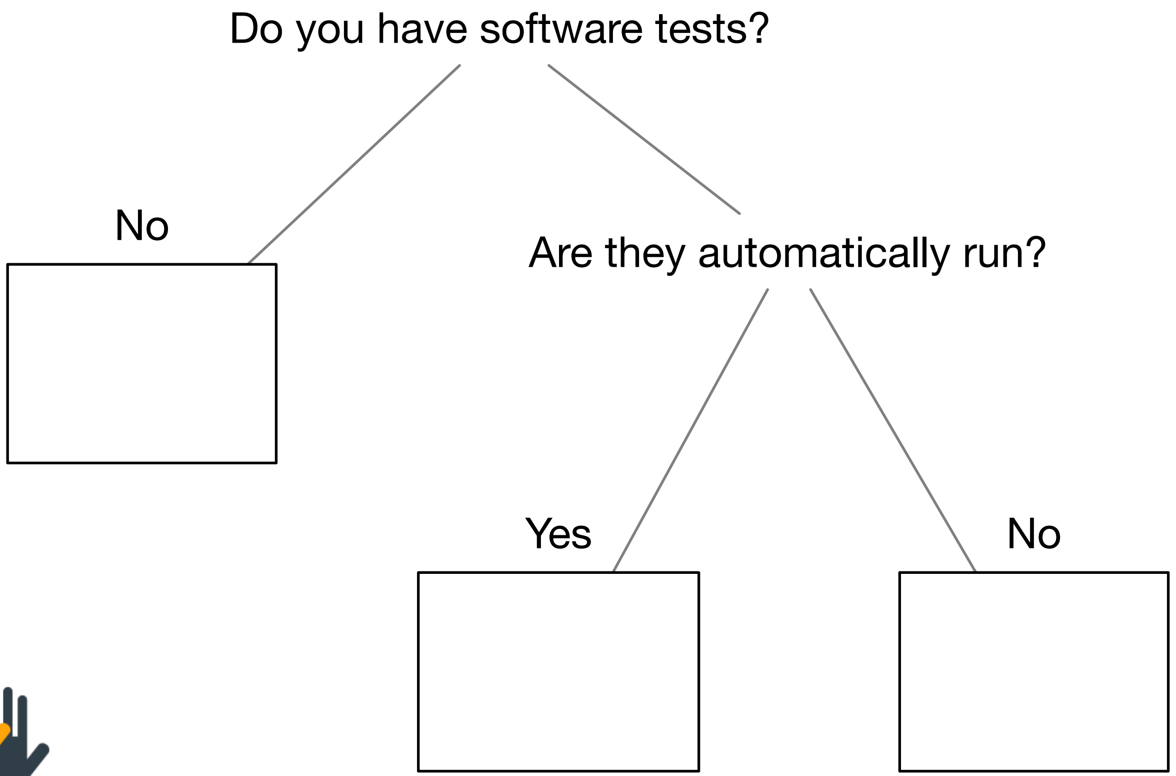
HOW LONG CAN YOU WORK ON MAKING A ROUTINE TASK MORE EFFICIENT BEFORE YOU'RE SPENDING MORE TIME THAN YOU SAVE?
(ACROSS FIVE YEARS)

	HOW OFTEN YOU DO THE TASK					
HOW MUCH TIME YOU SHAVE OFF	50/DAY	5/DAY	DAILY	WEEKLY	MONTHLY	YEARLY
1 SECOND	1 DAY	2 HOURS	30 MINUTES	4 MINUTES	1 MINUTE	5 SECONDS
5 SECONDS	5 DAYS	12 HOURS	2 HOURS	21 MINUTES	5 MINUTES	25 SECONDS
30 SECONDS	4 WEEKS	3 DAYS	12 HOURS	2 HOURS	30 MINUTES	2 MINUTES
1 MINUTE	8 WEEKS	6 DAYS	1 DAY	4 HOURS	1 HOUR	5 MINUTES
5 MINUTES	9 MONTHS	4 WEEKS	6 DAYS	21 HOURS	5 HOURS	25 MINUTES
30 MINUTES	▨	6 MONTHS	5 WEEKS	5 DAYS	1 DAY	2 HOURS
1 HOUR	▨	10 MONTHS	2 MONTHS	10 DAYS	2 DAYS	5 HOURS
6 HOURS	▨	▨	▨	2 MONTHS	2 WEEKS	1 DAY
1 DAY	▨	▨	▨	▨	8 WEEKS	5 DAYS

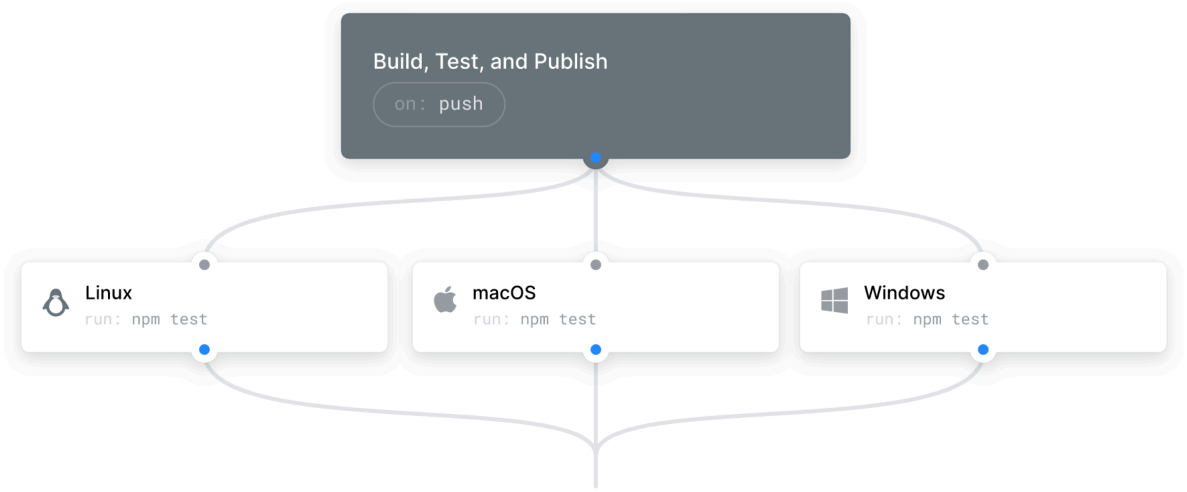
- More than one developer?
  Write tests!
- Your code is complex?
  Write more tests!
- Plan on developing long term?
  Write even more tests!
- Want to enjoy your life?
  Trust me: write tests!

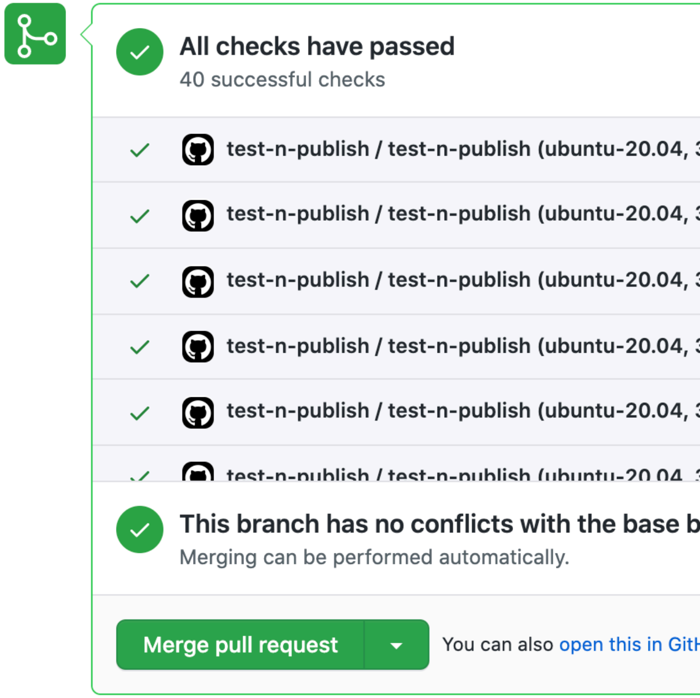
- Setting up a test environment is *always* worth the effort!

Survey


Do you have software tests?
No
Are they automatically run?
Yes
No

Continuous Integration with Actions




All checks have passed
40 successful checks

✓ test-n-publish / test-n-publish (ubuntu-20.04, 3
✓ test-n-publish / test-n-publish (ubuntu-20.04, 3
✓ test-n-publish / test-n-publish (ubuntu-20.04, 3
✓ test-n-publish / test-n-publish (ubuntu-20.04, 3
✓ test-n-publish / test-n-publish (ubuntu-20.04, 3
✓ test-n-publish / test-n-publish (ubuntu-20.04, 3

This branch has no conflicts with the base b
Merging can be performed automatically.

Merge pull request

You can also open this in GitH

Tags and Releases: Versioning Your Software

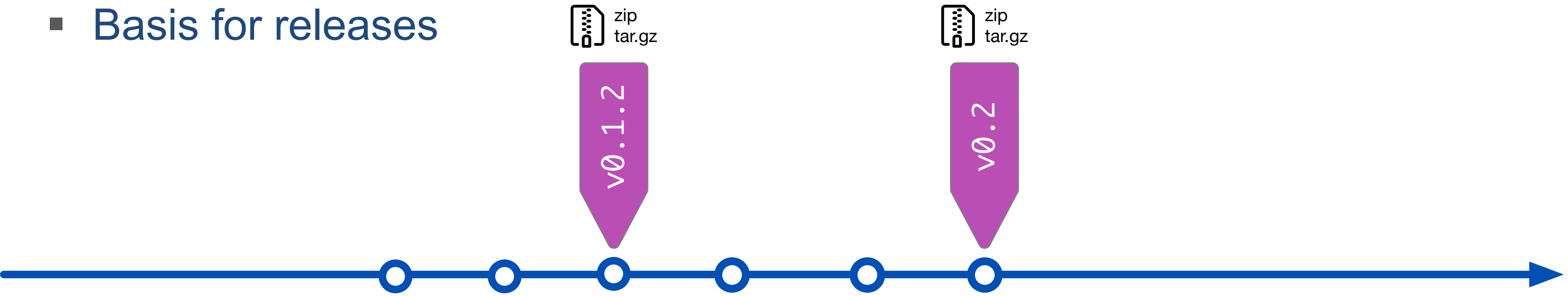
Tags

```
git tag v0.1.2
git push --tags
```

- Marks a defined state (commit)
- Easy to find and go back to
  `git checkout v0.1.2`
- Basis for releases

Releases

- Code + what ever you want
- e.g., generated or compiled files



Stashing Changes

- When working together, editing the same file is common
- Local changes need to revert back to HEAD without data loss

```
git stash push (<file(s)>)
```

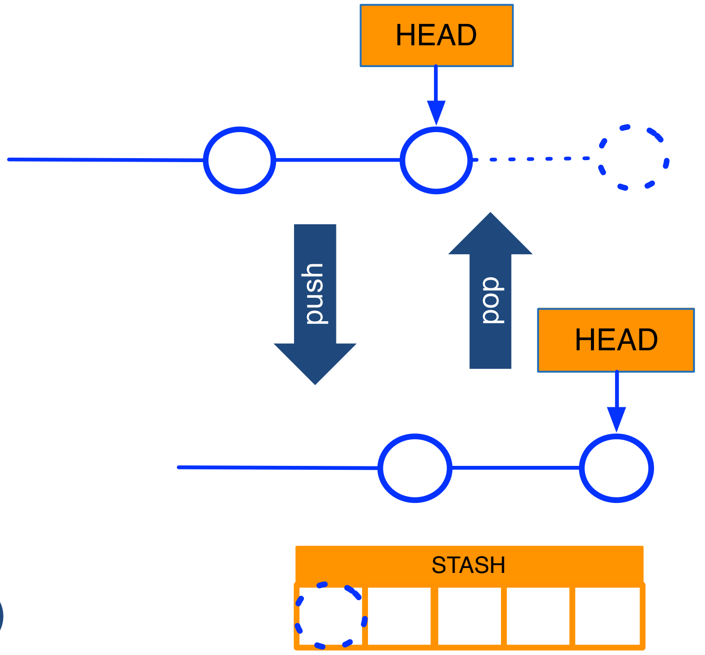
Put files into temporary stash area
Restore HEAD

```
git stash list/show
```

List/show all pushes

```
git stash pop
```

Get changes back (and merge if needed)



Additional Features

- GH Forks
  - For collaboration on project, fork repository to your account
  - Create and work on branch in forked repository
  - Create PR by “compare across forks” to original project
- GH Wiki & Pages
  - Place for documentation, show cases, project webpage, …
  - In free plan: GH pages only for public repos
- Cite code with DOI by Zenodo
  - Financed by the EU
  - Cloud storage for papers, reports, video, software, data, …
  - Integrates in GH to create DOI with every release

Further information

- https://docs.gitlab.com
- https://github.com/git-guides
- https://guides.github.com
- https://git-scm.com

- Continuous Integration:
  - https://docs.github.com/en/actions
  - https://docs.gitlab.com/ee/ci/

ohmygit.org