



OST

Eastern Switzerland
University of Applied Sciences

Distributed Systems (DSy)

Monorepos vs. Polyrepos (Multirepo)

Thomas Bocek

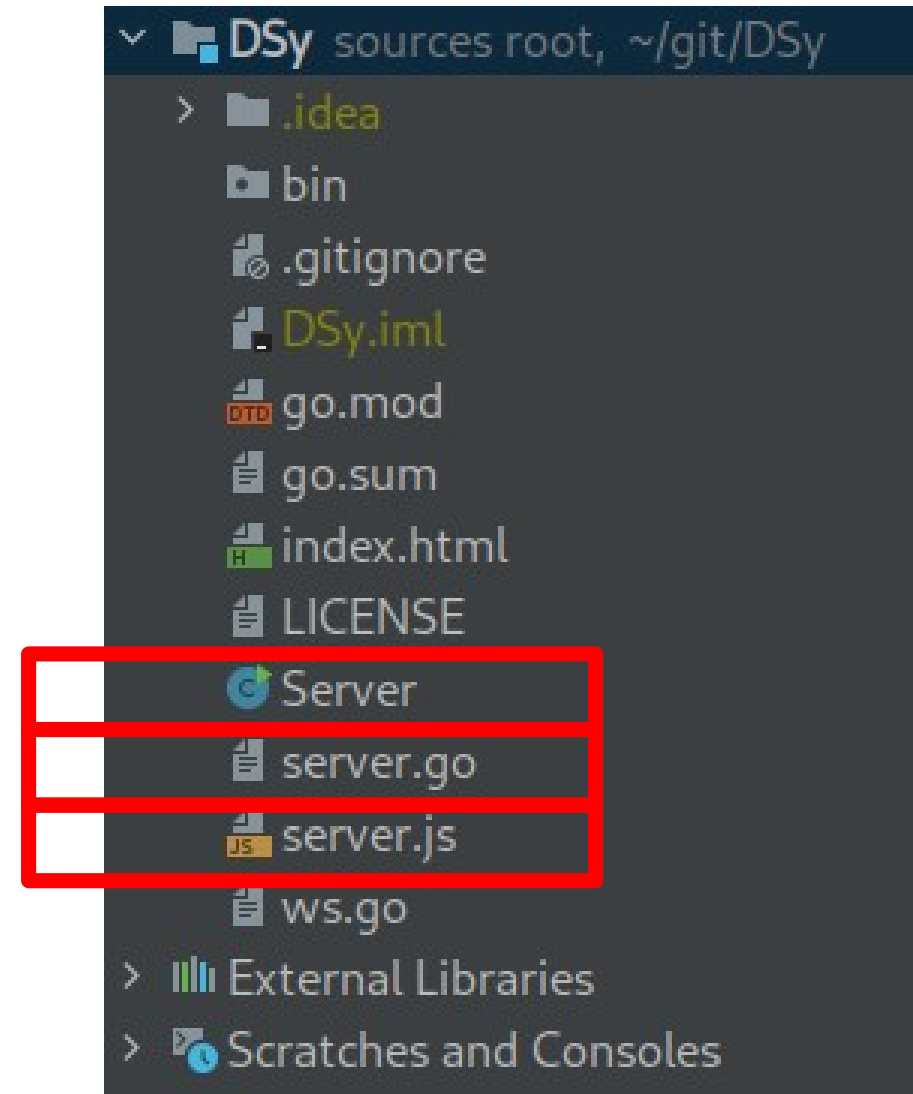
17.03.2022

Learning Goals

- Lecture 4 (Protocols, part 2)
 - How can new protocols improve latency?
 - What is an amplification attack?
- Lecture 4 (Repositories)
 - What is a monorepo, what is a polyrepo?
 - When to use which type?

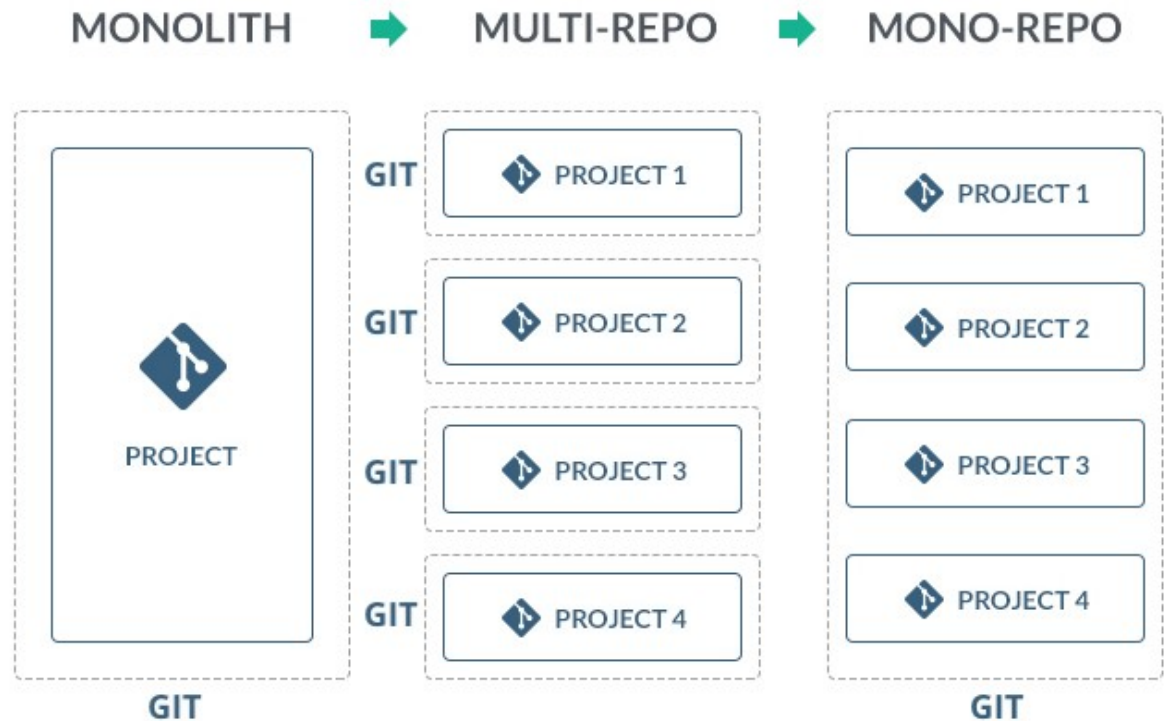
Project Setup

- Project setup the wrong way:
 - <https://github.com/tbocek/DSy>
 - Everything (Java, Golang, Javascript) in one directory
- Split up
 - Backend, frontend, service1, etc. in separate repo
 - ~1 Technology per split
 - Most likely you won't have a frontend mix of frontend technologies e.g., Angular with Vue
 - Sometimes you do :)
 - Sometimes you have a script directory, with different languages (bash, javascript)
 - Sometimes you don't :)



Monorepo

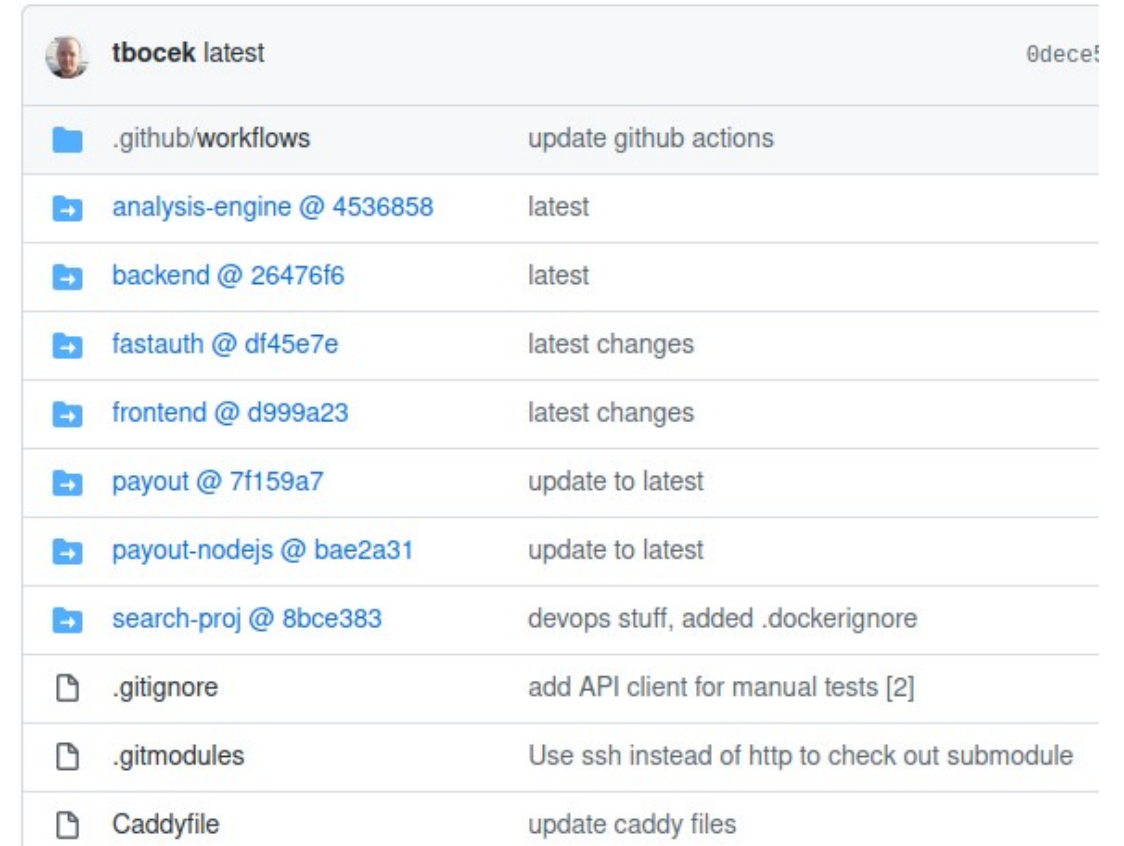
- One repository for all projects
 - 1 directory with frontend, 1 directory with backend, etc.
 - Tools e.g., **lerna** - update dependencies, **hoisting**
- Other names: onerepo or unirepo
- Examples
 - **Simform**
 - Started with monorepo, switched to mulirepo, now with hybr approach “you can’t blindly follow any approach”
 - Google, Facebook, Twitter
 - Use monorepos (others **do not**)
 - **Flatfeestack**
 - Uses hybrid approach, thinking about switching back to polyrepo (reduce complexity)



<https://codefresh.io/continuous-integration/using-codefresh-with-mono-repos/>

Polyrepo

- Multiple repositories for a project
 - Frontend in a different repository than the backend
 - Example: <https://github.com/flatfeestack>
 - Wip, not ready to make it public...
 - Frontend: Svelte, npm
 - Backend: Golang
 - Other names: manyrepo or multirepo
- Sync via git **submodules** or via bash script
 - **Submodules**: can also be used as dependency management



File	Commit Message
.github/workflows	update github actions
analysis-engine @ 4536858	latest
backend @ 26476f6	latest
fastauth @ df45e7e	latest changes
frontend @ d999a23	latest changes
payout @ 7f159a7	update to latest
payout-nodejs @ bae2a31	update to latest
search-proj @ 8bce383	devops stuff, added .dockerignore
.gitignore	add API client for manual tests [2]
.gitmodules	Use ssh instead of http to check out submodule
Caddyfile	update caddy files

Pro/Cons - Opinion

- **Monorepo**
 - Tight coupling of projects
 - Everyone sees all code / commits
 - Encourages code sharing within organization
 - Scaling: large repos, specialized tooling
- **Polyrepo**
 - Loose coupling of projects
 - Fine grained access control
 - Encourages code sharing across organizations
 - Scaling: many projects, special coordination
- Opinion: **Accenture** - “From my experience, for a smaller team, starting with mono-repo is always safe and easy to start. Large and distributed teams would benefit more from poly-repo”
- My opinion: for small teams and project, use polyrepo. (I worked with small teams with mono and polyrepo, I have worked in big projects with polyrepos, but never in a big project with monorepos)
- Other opinion (sales pitch): <https://monorepo.tools>