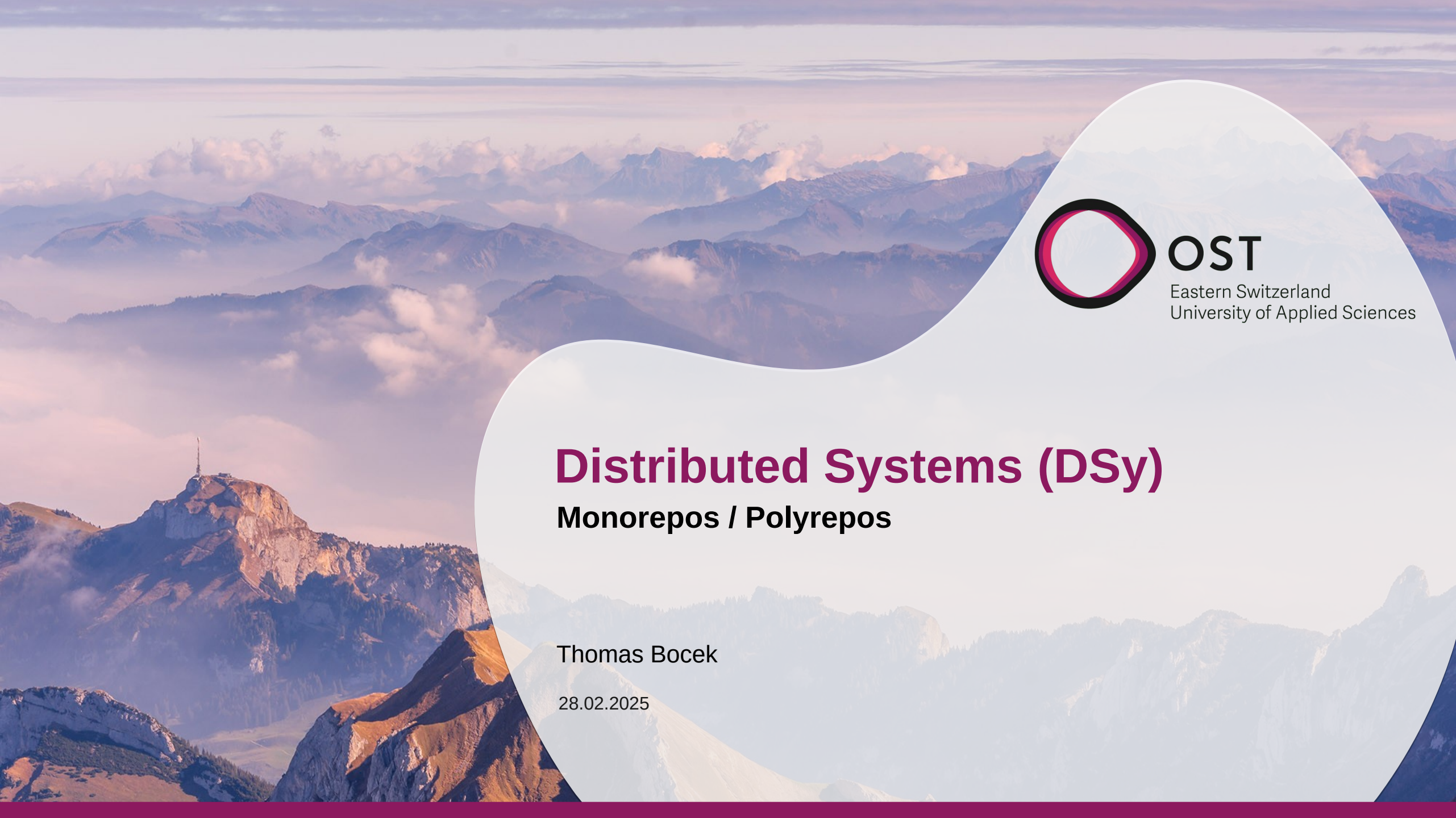


OST
Eastern Switzerland
University of Applied Sciences

Distributed Systems (DSy)

Monorepos / Polyrepos

Thomas Bocek

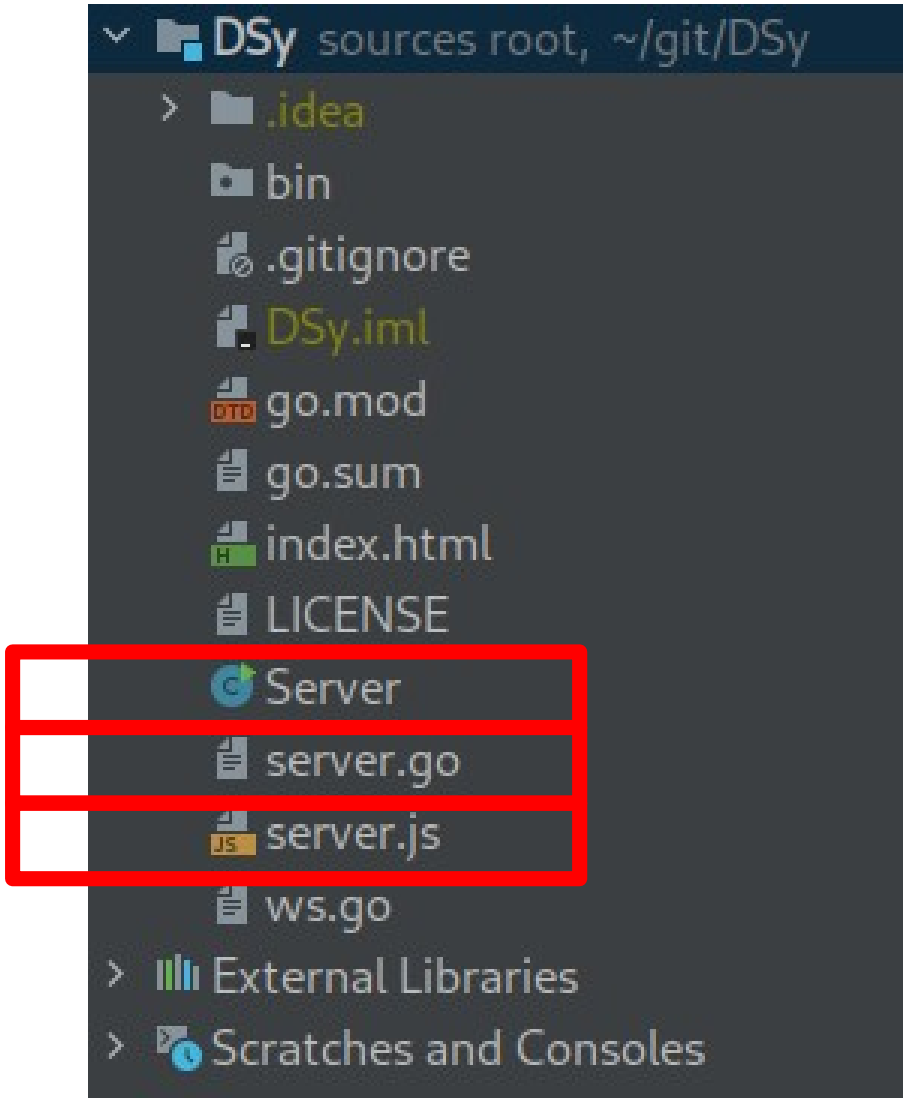
28.02.2025

Learning Goals

- Lecture 3 (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) flat in one directory – do not do this
- But, keep it simple at the start, follow best practices [example]
  - Java: src/main/java – default in maven/gradle
  - Golang: everything in root directory [example]
  - Javascript/Typescript
    - Depends on type: backend / frontend / plugin
    - src, public, tests [1, 2, 3, 4]



OST

Project Setup

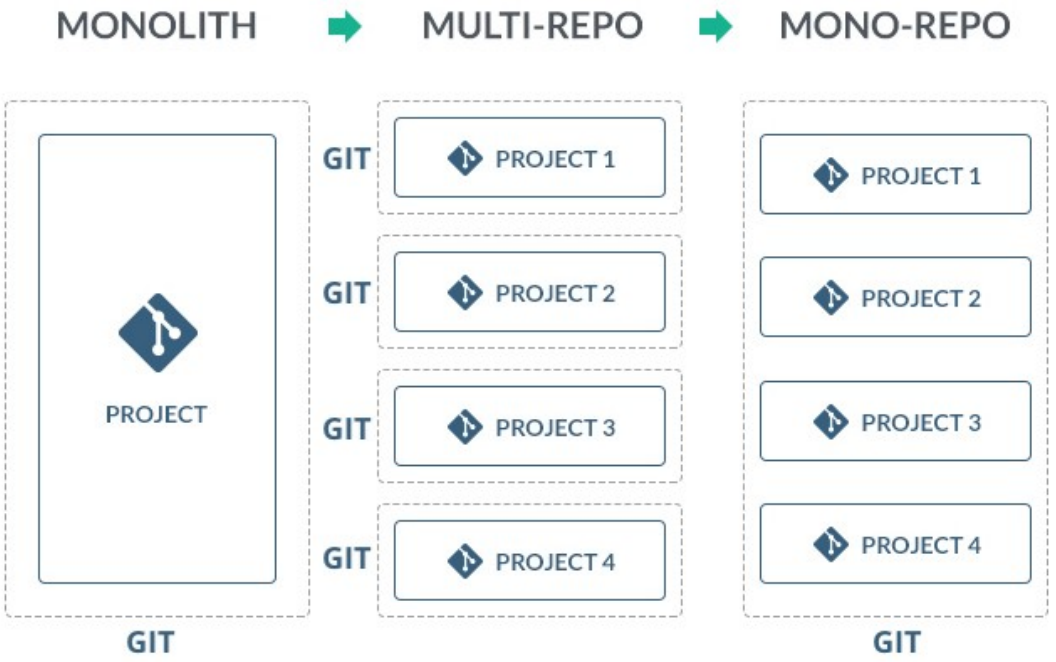
- General rules

  1) Be consistent
  2) Other projects always prefer other structures
  3) A perfect structure does not exist

- Split up
  - Backend, frontend in separate repositories
  - ~1 technology per split for simple projects
    - Most likely you won't have a frontend mix of frontend technologies e.g., Angular with Vue
    - Sometimes you have a script directory, with different languages (bash, javascript)
  - More complex setups?
    - Multiple backends, multiple packages

Monorepo

- One repository for all projects (aka onerepo or unirepo)
  - 1 sub-directory with frontend, 1 sub-directory with backend, etc.
  - Tools e.g., turborepo - update dependencies, hoisting
  - Tech independent: Bazel, Buck2
- Examples
  - Simform: started with monorepo, switched to mulirepo, now with hybrid approach “you can’t blindly follow any approach”
  - Google, Facebook, Twitter: use monorepos (others do not)
  - Flatfeestack: used hybrid approach (scripts, submodules), now monorepo (e.g.)

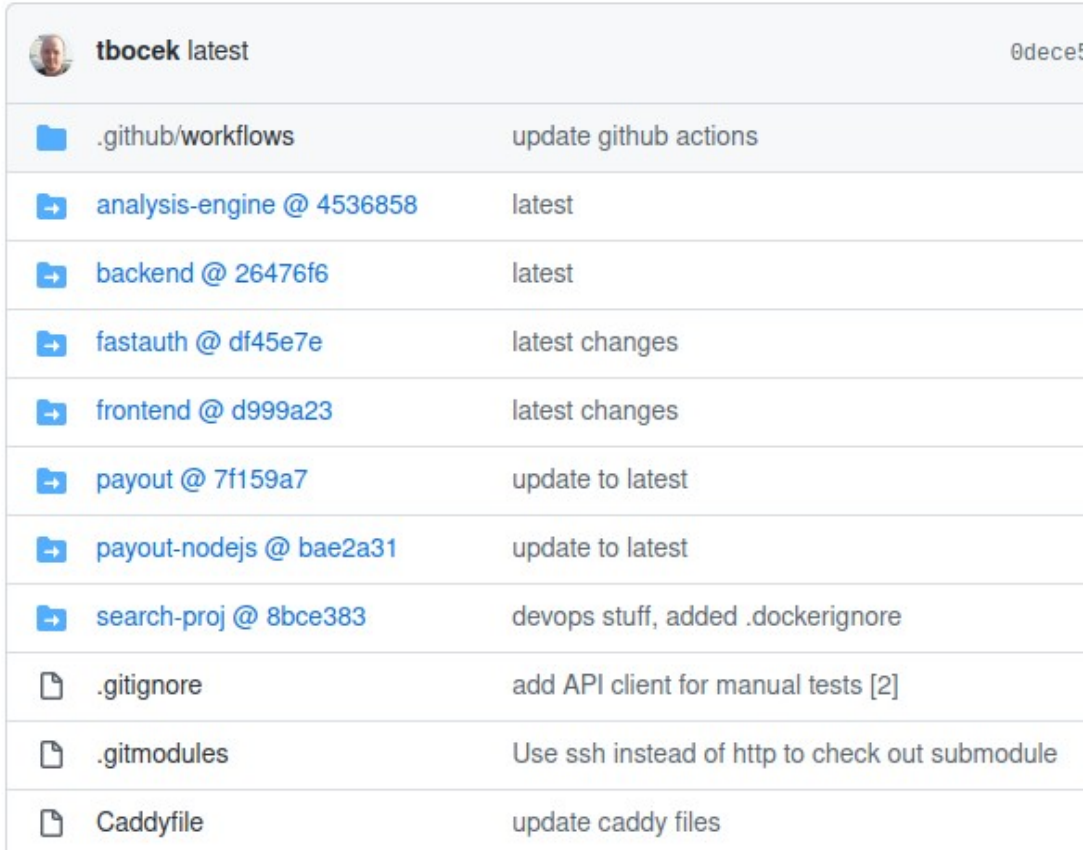


https://codefresh.io/continuous-integration/using-codefresh-with-mono-repos/

OST

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
- Sync with repo

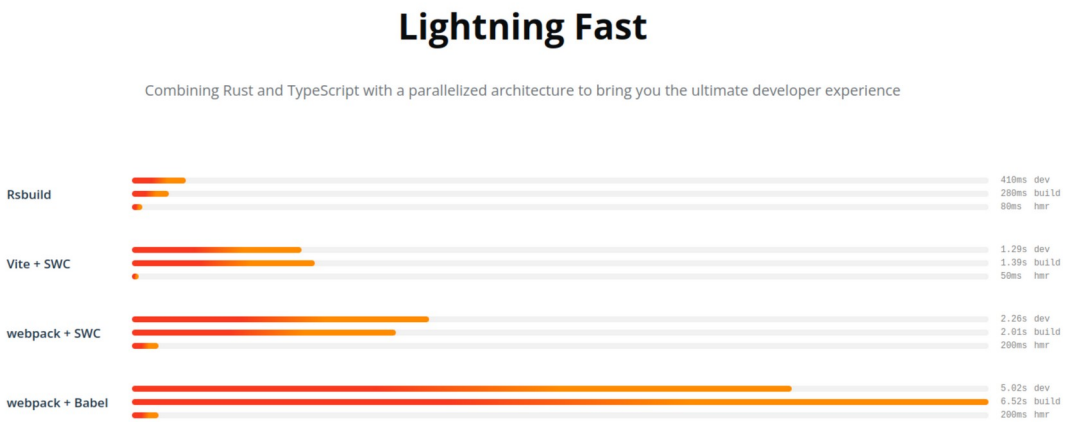


Pro/Cons - Opinion

- **Monorepo**
  - Tight coupling of projects
    - E.g., generating openapi.yml from backend, generate types for frontend → simply copy
  - Everyone sees all code / commits
  - Encourages code sharing within organization
  - Scaling: large repos, specialized tooling
- **Polyrepo**
  - Loose coupling of projects
    - If you want to generate openapi.yml, you need access from the backend repository to the frontend (e.g., curl+token)
  - Fine grained access control
  - Encourages code sharing across organizations
  - Scaling: many projects, special coordination

Tools

- Overview: https://monorepo.tools
- Pnpm workspace
- Turbo / Lerna/Nx – JS/TS, Gradle monorepo support, otherwise you need to do it manually, or write a script / make / just
  - Caching!
  - E.g., Bazel: (big projects) → define input/output → if no change in input, do not run command
  - But e.g., Golang, Rust have extensive caching mechanisms
  - Caching with docker in upcoming lecture
- Project dependencies: generating types
- Many tools competing in the same space
- Complexity!
  - Is build speed slow? Rsbuild ~ 211ms



OST

Examples

- Turborepo
  - npx create-turbo@latest my-turborepo
  - npx turbo build
  - npx turbo dev / npx turbo dev --filter=web
  - Add package with separate package.json...
- pnpm workspace
  - Another solution for packages

- Buck2: tried to run a simple tutorial
  - Error after error… complexity!

```
Time elapsed: 0.2s
BUILD FAILED
Error evaluating build file: `root//:BUCK`

Caused by:
    0: Traceback (most recent call last):
         * BUCK:9, in <module>
             cxx_binary(

       error: Error coercing attribute `srcs` of `root//:main`
         --> BUCK:9:1
          |
        9 | / cxx_binary(
       10 | |     name = "main",
       11 | |     srcs = ["main.cpp"],
       12 | |     link_style = "static",
       13 | | )
          | |_^
          |
    1: Error coercing attribute `srcs` of type `attrs.list(attr
```


OST

Pro/Cons - Opinion

- 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: tightly-coupled projects: monorepo, loosely-coupled: polyrepo
- Challenge task: I would chose a monorepo

- Upcoming lectures
  - Dependency hell
  - Docker / docker compose
  - Supply chain attacks!