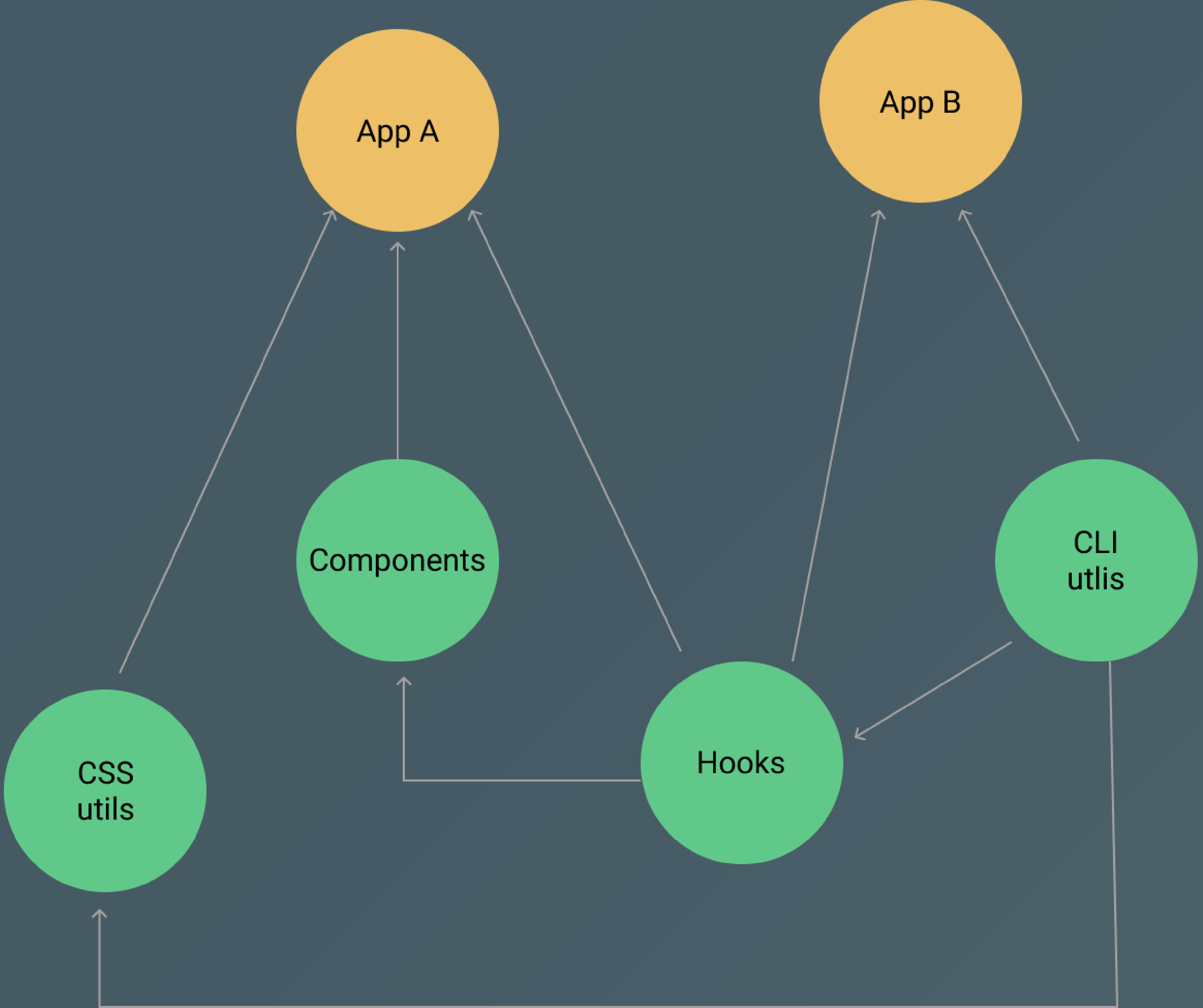
Mono-repo Starter

https://github.com/cef62/monorepo-template

What is a mono-repo?

What problem space does it try to solve?


App A
App B
Components
CLI
utlis
CSS
utils
Hooks

Mono-repo tools

- yarn workspaces
- lerna
- pnpm
- turborepo
- nx
- rush

This repository is a quick way of bootstrapping a new mono-repo for a Web Application.

Tech stack

- pnpm
- typescript
- esbuild
- vite JS
- prettier
- eslint
- husky
- lint-staged
- chokidar

Used to run the examples:

- React
- MUI

Mono-repo Goals

The repository focuses on creating one or more applications sharing and consuming multiple utilities.
You have

- one application: `playground`
- two components collections- `cool-components` and `other-components` -
- and a tool: `esbuild-plugins`

Coding style and Formatting conventions

- An editor config file defines basic rules for the coding style.
- `Prettier` and `eslint` automatically format the code for you.

Linting and Static type-checking

- `eslint` ensures standard best practices on how to write Javascript code.
- `Typescript` offers static type-checking and a better DX.

Automation and git hooks

`husky` and `lint-staged` allows for automation driven by `git hooks`.

A `pre-commit` hook will format your code before committing it. This automation will guarantee consistency on your codebase no matter who's writing the code.

You can easily add other automation to validate your code before pushing it to Github or refreshing the local artifacts after fetching new code.

PNPM

Out of many tools helping you build a mono-repo, `pnpm` is my choice of preference.

It's not full-featured as `nx` or `rush` and not quite famous as `yarn` or `lerna`, but it's by far the more effective tool for my needs.

PNPM Configuration

`pnpm` offers outstanding defaults out-of-the-box.

A few tweaks I usually apply are:

- `exact dependencies`, no versioning ranges allowed
- `strict peer dependencies`, ensuring everything run as expected
- shell emulator, for better cross-platform scripting

PNPM Workspace dependencies

One of the things I like the most about `pnpm` is that workspace dependencies are 1st class citizens.

Esbuild and Chokidar

Those two tools enable you to develop and consume multiple libraries from one application with zero downtime.

- `esbuild` compiles a medium-size library in a tenth of seconds or less
- `chockidar` is a smart watcher over your code

Vite JS

`vite` has been a game-changer for me. `webpack` is excellent, and I still use it in many projects, but the speed and simplicity of `vite` are unparalleled.

NodeJS exports

`Node` had exports map for a long time.
Nowadays, most transpiler and bundler tools support the feature.

The most significant issue for me is that `typescript` doesn't support the features natively yet.

Luckily, there's an easy workaround :)

@cef62 - Twitter

@cef62 - Github