什么是 monorepo?

Monorepo 是管理项目代码的一个方式，指在一个项目仓库 (repo) 中管理多个模块/包 (package)，不同于常见的每个模块建一个 repo

代码仓库管理有哪些常见方案方案，有什么优劣

git-subModule，难用，有认知成本。

MultrRepo 中，一个项目一个仓库，配置，组件，工具类，通过版本管理工具作相应的权限管理。

缺点因为物理隔离，物理隔离，修改某个问题或者添加需求需要都改一遍。

Monorepo 可以在一个仓库里维护多个 package，可统一构建，跨 package 调试、依赖管理、版本发布都十分方便，搭配工具还能统一生成 CHANGELOG。

缺点是因为在同一个仓库里，没法做到完美的权限控制，比如 A 部门要跟 B 部门合作，但是不需要 B 部门看到自己的代码，这也是目前版本管理工具都具有的问题

Monorepo 怎么实现

- yarn workspace

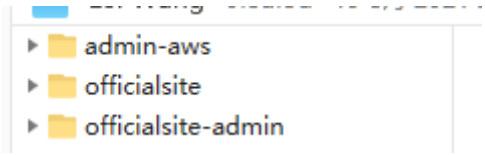
yarn workspace 是 yarn 实现的，常用于项目代码的管理

- Lerna

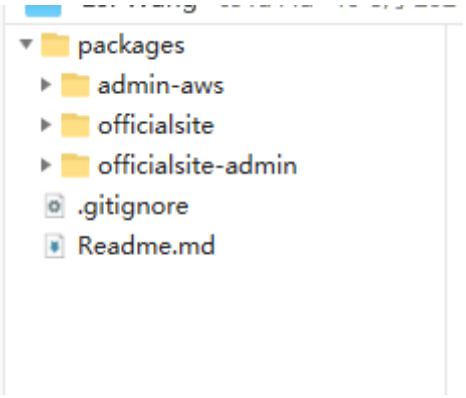
Lerna 另一个管理工具，vue,babel,react 等都在用，常见于需要发布 npm 的包的管理中。

有没有实例

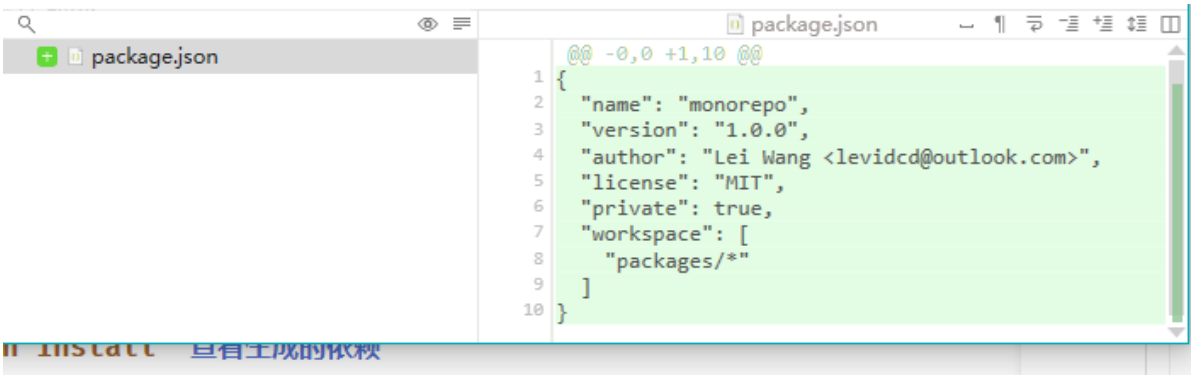
已有项目管理后台，官网，官网后台，一个项目一个仓库



调整目录结构，将相互关联的项目放置在同一个目录，推荐命名为 packages；

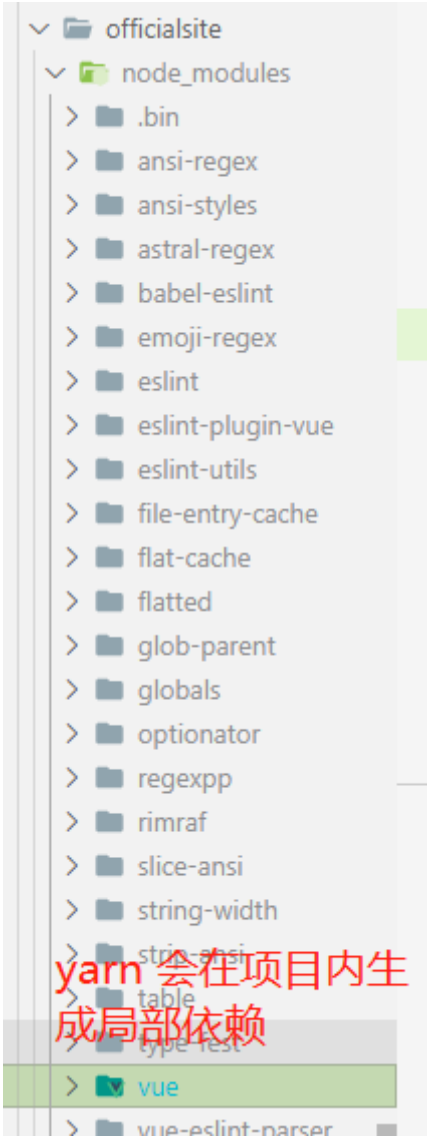


package.json 添加 `"workspaces": [ "packages/*"]`, 说明 packages 下为各个项目



`yarn install` 查看生成的依赖

> admin-aws 与 official-admin 依赖 vue@2.6,
>
> official 依赖 vue@3



这样基础的 monorepo 项目结构就完成了

怎么运行

- `yarn add eslint -D -W` W 代表要在 monorepo 中添加全局依赖
- `yarn workspace admin-aws add lodash` 代表要在指定 packages 中添加依赖
- `yarn workspace <workspace_name> <command>` 在指定的 package 里运行指定的命令

例如 `yarn workspace admin-aws run serve` 开启 admin-aws 的 devserver

- `yarn workspaces run <command>` 在所有 package 中运行指定的命令,若某个 package 中没有对应的命令则会报错

  例如 `yarn workspaces build` 构建所有包

- `yarn install` 在项目去构建依赖

怎么统一配置

Eslint

```
// package-A .eslintrc.js
module.exports = {
  extends: "../../.eslintrc.js",
};
```

Babel

```
// package-A babel.config.js
module.exports = {
  extends: "../../babel.config.js",
  presets: ["@vue/cli-plugin-babel/preset"],
};
```

如何复用

建立一个 package，在其他的仓库中引用它

- 我新建一个 package,物理路径为 package/shared, package name 为 `@kitc/shared`

- 接下来在其他 package 中可以引用这个包，重新 `yarn install`

```
"name": "admin-aws",
"version": "0.1.0",
"private": true,
▷ 调试
"scripts": {
  "serve": "scripty",
  "build": "scripty",
  "lint": "scripty"
},
"scripty": {
  "path": "../../scripts/packages"
},
"dependencies": {
  "core-js": "^3.6.5",
  "vue": "^2.6.11",
  "@kitc/shared": "^1.0.0"
},
"devDependencies": {
```

- 在项目中去使用它

```
  <div id="app">
    <img alt="Vue logo" src="./assets/logo.png" />
    <com404></com404>
    <HelloWorld msg="Welcome to Your Vue.js App" />
  </div>
</template>

<script>
import HelloWorld from "./components/HelloWorld.vue";
import { com404 } from "@kitc/shared";
console.log(com404);
export default {
  name: "App",
  components: {
    HelloWorld,
    com404,
  },
};
</script>
```

通过这种方式可以共用配置，组件，工具类，当项目成长到一定程度，可以积累出团队的工具库，组件库，就像一块块乐高积木，可以搭建出适合业务的系统

适用于什么样的项目

全栈项目

管理后台 SaaS

H5 落地页

参考

- 项目地址：`https://github.com/levidcd/monorepo`
- Lerna:`https://github.com/lerna/lerna`