平台架构设计

- 架构背景：为了更好的进行开发、维护、使用、打包、调试、管理、可复用，进行了统一 规划。
- 架构思想：分解各个模块功能，使其功能单一（CMD），以达到降低耦合、提高灵活运用的目的
- 实现目标：半定制、半产品

  在业务应用可视上实现高度定制

  在高度定制的情况下仍然能保持产品的核心、开发更加快捷、能更好的把控产品质量
- 开发要求：每个代码块不能超过100行（用户交互、动画类除外），每个代码块要考虑高可用、高复用、高性能
- 实现多项目共存，多项目同时运行、打包，也能单个指定打包
- 解决公司当前应急项目线，因为项目的增加，各个项目客户需求不一，在原来产品上不断使用配置文件、或者前端代码分化，逐渐形成难以管理、维护、功能新增等问题

 # sinvie-3

> 使用vue-cli@4.5搭建的 Vue 3.0 + TypeScript 4.0 多页面项目。

- 公共能力（UI组件库、业务框架、工具库、静态资源等）统一在 `@/common` 里面管理，方便复用和维护

产品

每个产品的开发形态主要有以下几个步骤

组装　　组装

零件　=>　功能块　=>　产品

产品的生产模式，是使用零件组合的方式，从零散的器件到一个功能模块，再使用功能模块组转成产品

就像一台汽车一样，它的组成是由无数的零件，组装成大部件，再用大部件组成汽车

汽车的 “功能块” 可以分为：发动机、轮胎、外壳、座椅、管道等

它的 “功能块” 也是由无数的 “零件” 组成，汽车的改装就像我们项目上客户的需求一样，各式各样，所以做产品的时候我们用 “零件” 组成 “代码块” ，最后再组成产品，它也可以接受像汽车一样的 “改装”，每个产品的每个模块都有一个默认的样子，也就是它最初设计的样子，可以称为标配版。同时也是在这时候解决项目的 模块的耦合问题、高定制需求、臃肿的配置项、分化的代码的问题，如果你疑惑为什么会有这些东西，那么我可以告诉你，这是应急项目面临的问题，一切源于需求，只有经历过应急项目、深受其害的人才能明白这个平台意味着什么，它的收益者除了开发人员，还有公司的所有人员。

每个产品都有对应的“图纸”，让工程师更好的了解产品的构造，方便维护、开发，所有的零件使用全部采用引入式

当你需要定制某个模块时，只需要按照图纸找到你需要的 “零件” ，然后在你自己开发的 “功能块” 上引入你需要的 “零件” ，组装成你想要的样子

降低了代码的耦合度，提高了可定制能力，减少了因为项目增多客户需求不一时过于厚重的配置项，不需要再进行代码分化一个bug只需要改一遍，会花费些时间在组装上。请谨记和理解 “零件” “功能块” 这两个词。

例如我有个指挥大脑3.0的产品

关于文件的归拢

考虑到以后可能还会有大屏可视化产品、soc产品的进入，作为一个平台提供者我要把它们归拢到一个文件下，要方便我们开发时的组装，还要把它突出显示出来告诉后续的人们，这是个产品文件夹，所以我把它放在src下面的第一层。

零件的定义、开发

开发时请在顶部注释好 使用方法

请根据模块的划分去进行零件的制造，请谨记 “根据模块” ，这个对零件的制造和文件归纳管理（“图纸”的一部分）很重要

例如我的产品头部是这样的：

那么我就会在我的指挥大脑3.0的产品的文件下面，新建一个名为header的文件夹（注意语义化和开发时布局方式），来存放这个模块的所有零件，同时也有个header.vue作为当前模块的默认 “功能块”

当你能理解这个概念的时候，你就会疑惑它数据交互、vuex状态管理、router如何处理，没有东西是十全十美的，我们只能尽善尽美！

数据交互

假如你看过公司的架构图你就能明白，它的验证方式是由一个检验服务统一进行检验的，所以在请求方面我们主要考虑的是接口的管理，我决定把它耦合到每个模块中，不对所有的接口进行集中管理，如果你要寻找指定接口，请使用编辑器携带的搜索功能进行。

vuex状态管理

在这个产品下有个vuex的文件夹，是这个产品所要用到vuex状态的管理文件，按照vue的vuex module方式使用

router

router的管理也是一样的有个router的文件夹，专门管理这个项目的router数据

在 product 中生产、制造零件，在 project 下的项目中进行装配

在装配的时候按照 “图纸” 引入router、vuex、axios、公共能力进行使用

当产品交接到项目上时，请不要更改产品中的文件，如有需要请另外组装

```
├─public                    开发时框架指定需要的静态文件夹
├─src                       路径别名 @
|  ├─capacity               能力集成
|  ├─common                 所有项目共享资源
|  ├─product                用于存放产品的文件夹
|  |   ├─CommandBrain3.0    产品名
|  |   |  ├─drawing         图纸
|  |   |  ├─mainCapacity    主要能力文件夹
|  |   |  |  ├─vuex            vuex 文件
|  |   |  |  |  └ module.js    vuex下的模块
```

```
|  |  |  ├─router                router 文件
|  |  |  |  └ index.js           router 数据文件
|  |  |  ├─axios                 请求文件
|  |  |  |  └ index.js           用于定义axios 请求前和请求后、参数等
|  |  |  ├─header                模块名
|  |  |  |  ├─ weather
|  |  |  |  ├─ date
|  |  |  |  └ header.vue
|  |  |  ├─flyRound              飞巡模块
|  ├─project                      项目文件夹
|  |  ├─demo                     项目名
|  |  |  ├─App.vue               当前项目主视口
|  |  |  ├─index.html            当前项目的模板html
|  |  |  ├─index.ts              当前项目的主入口
|  |  |  ├─shims-vue.d.ts        用于typescript 检验
|  |  |  ├─views                 项目vue文件存放夹
|  |  |  |  ├─page1.vue
|  |  |  |  └page2.vue
|  |  |  ├─store                 vuex文件夹
|  |  |  |  └index.ts
|  |  |  ├─static                静态文件夹
|  |  |  |  └demo.config.js
|  |  |  ├─router                router文件夹
|  |  |  |  └index.ts
```

- 每个项目有各自的路由、vuex、全局变量、HTML 模板、脚本入口、静态文件，都可以引用公共能力

一、命令行相关

- 安装依赖：`npm install`
- 启用 devServer：`npm run dev` 或 `npm run serve` 或 `npm start`
- 打包构建：`npm run build`
- 将根据启用的项目进行打包

二、目录树结构

```
├─ .browserslistrc # 浏览器配置
├─ .editorconfig # 编辑器配置
```

```
├─ .eslintrc.js # eslint 配置
├─ .gitignore # git 忽略文件列表
├─ README.md # 说明文档
├─ babel.config.js # babel 配置
├─ config # 整体配置
│ ├─ library.config.js
│ └─ project.config.js
├─ package-lock.json # 包信息
├─ package.json # 包信息
├─ public # 全局静态文件
├─ src # 路径别名 @
│ ├─ common # 所有项目共享资源
│ │ ├─ static # 共享静态文件，根据项目配置
│ │ │ └─ SuperMap # 超图
│ │ └─ utils # 常用工具函数
| ├─product                    用于存放产品的文件夹
| | ├─CommandBrain3.0          产品名
| | | ├─drawing                图纸
| | | ├─vuex                   vuex 文件
| | | | └ module.js            vuex下的模块
| | | ├─router                 router 文件
| | | | └ index.js             router 数据文件
| | | ├─axios                  请求文件
| | | | └ index.js             用于定义axios 请求前和请求后、参数等
| | | ├─header                 模块名
| | | | ├─ weather
| | | | ├─ date
| | | | └ header.vue
│ └─ project # 所有项目
│ ├─ demo # demo 项目
│ │ ├─ App.vue # demo 项目根组件
│ │ ├─ index.html # demo 项目 HTML
│ │ ├─ index.ts # demo 项目 JS 脚本入口
│ │ ├─ shims-vue.d.ts # TypeScript 中定义 *.vue 模块
│ │ ├─ router # demo 项目路由配置
```

```
│  │  └─ index.ts
│  ├─ static # demo 项目静态文件夹
│  │  └─ demo.config.js # demo 项目全局变量配置（不能与其他项目冲突，且不能是*.ts）
│  ├─ store # demo 项目 vuex 配置
│  │  └─ index.ts
│  └─ views # demo 项目视图
│     ├─ page1.vue
│     └─ page2.vue
└─ # 其他项目
├─ tsconfig.json # TypeScript 配置
└─ vue.config.js # vue-cli 配置
```