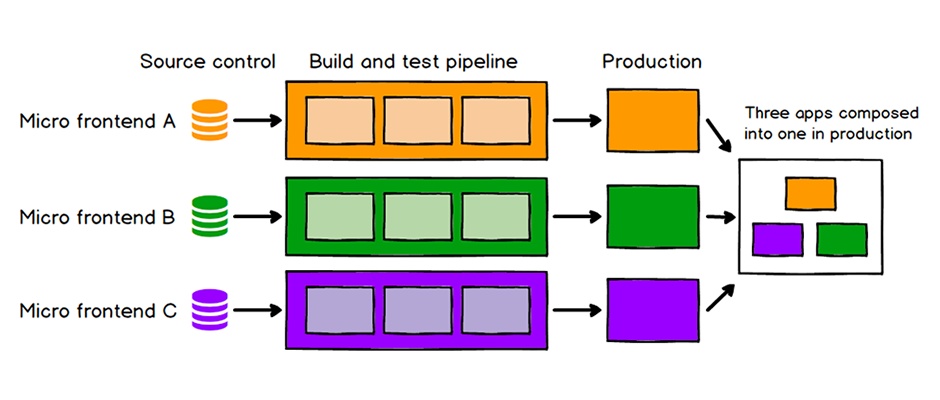
一、项目架构

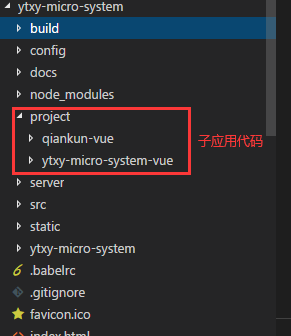


为了解决庞大的一整块后端服务带来的变更与扩展方面的限制，出现了[微服务架构（Microservices）](https://link.zhihu.com/?target=http://www.ayqy.net/blog/%e5%be%ae%e6%9c%8d%e5%8a%a1%e6%9e%b6%e6%9e%84%ef%bc%88microservices%ef%bc%89/" \t "https://zhuanlan.zhihu.com/p/_blank)：

然而，越来越重的前端工程也面临同样的问题，自然地想到了将微服务思想应用（照搬）到前端，于是有了「微前端（micro-frontends）」的概念：

即，一种由独立交付的多个前端应用组成整体的架构风格。具体的，将前端应用分解成一些更小、更简单的能够独立开发、测试、部署的小块，而在用户看来仍然是内聚的单个产品：

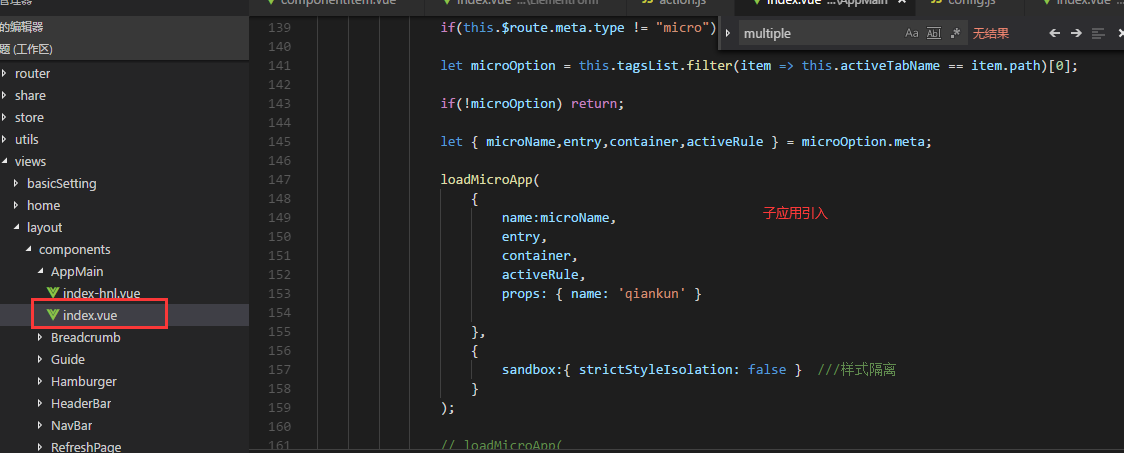
demo <https://demo.microfrontends.com/>



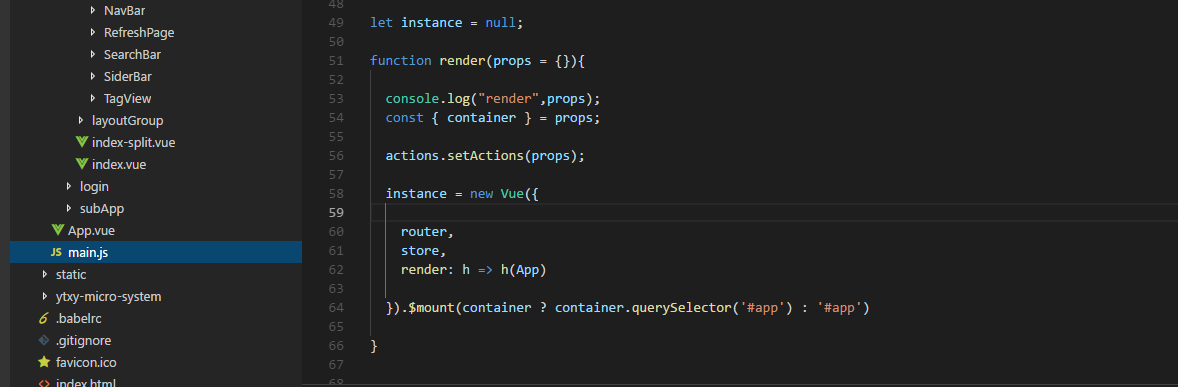
父子交互



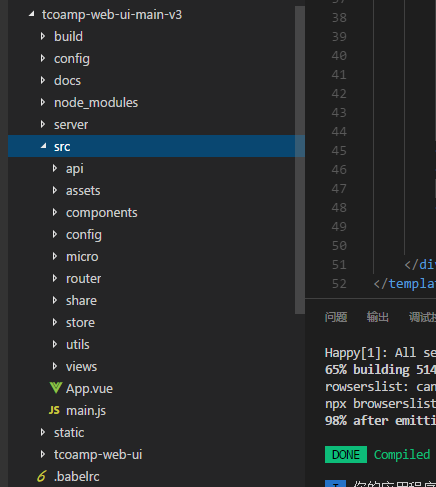
子应用引入



子应用渲染初始化



二、项目目录



docs 文档

Assets 图片，字体图标，公共样式

Components 公共组件

Config 配置文件

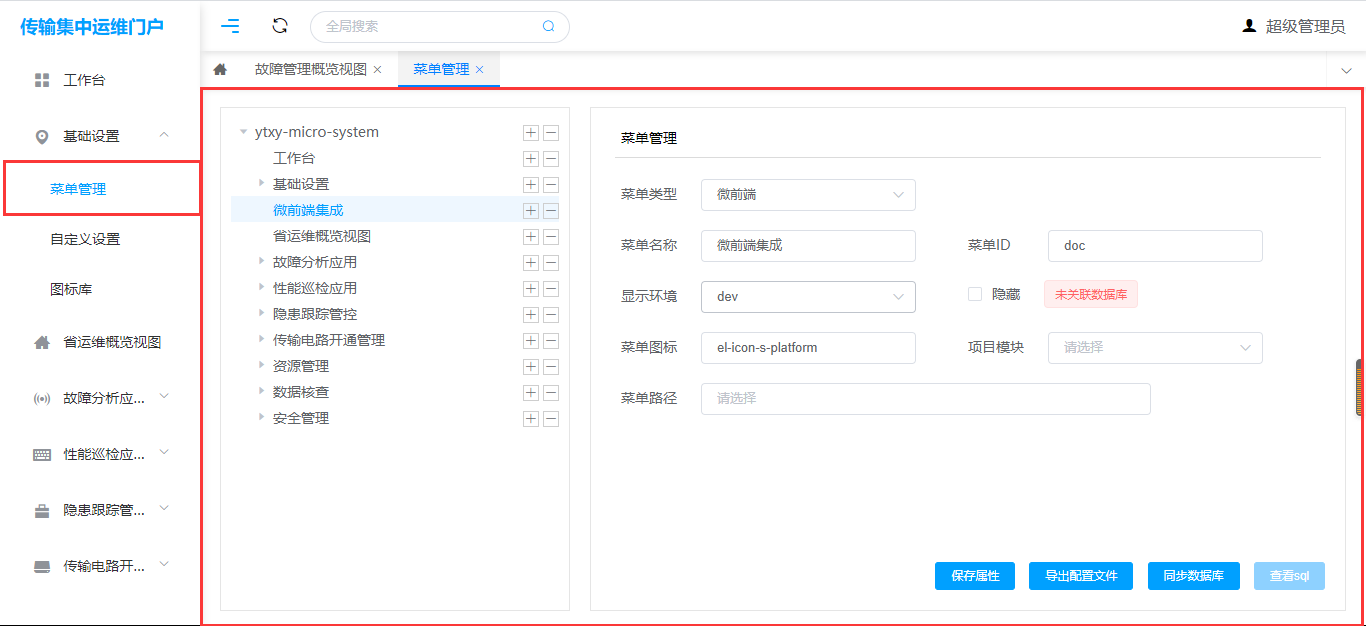
Micro 微前端配置

Router 路由配置处理

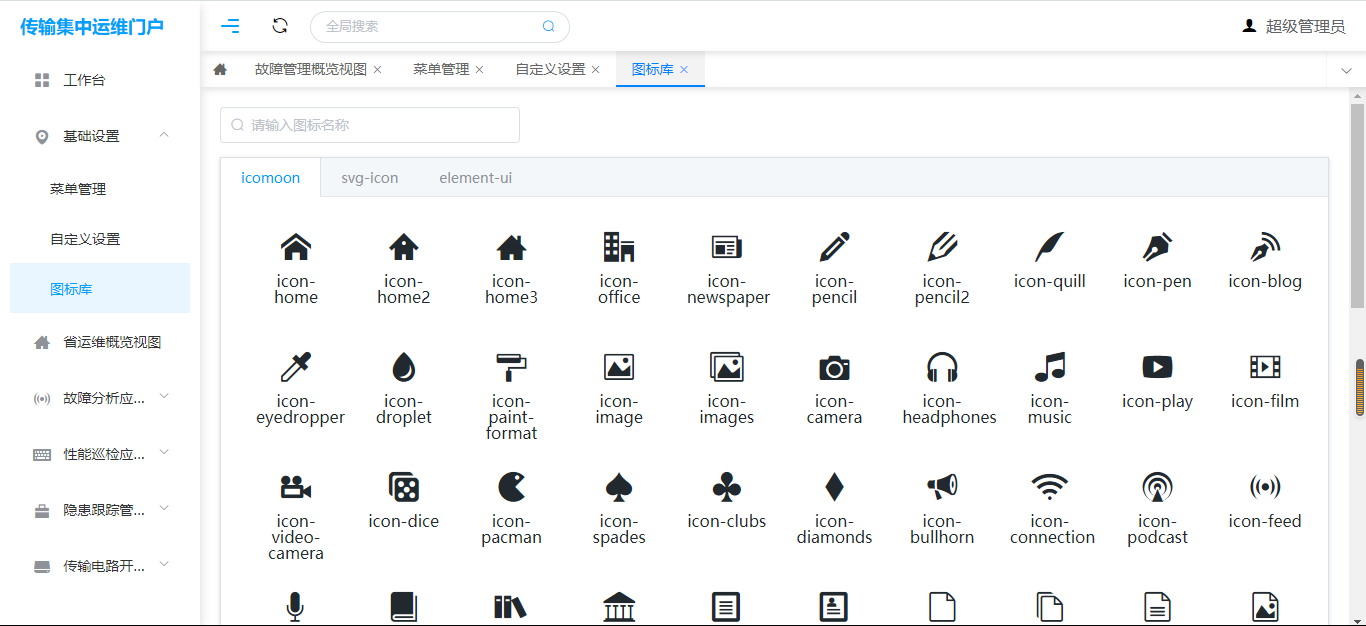
Share 父子应用交互

Store vueX

Utils 公共方法

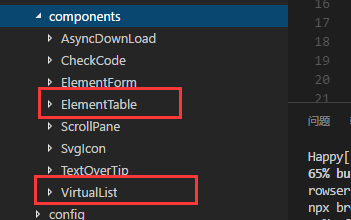


添加菜单



字体图标

1. 公共方法 组件



ElementTable --> 封装表格

入参格式

list\_table:{

cellSpan:false,

client:false,

hasIndex:false,

hasSelect:false,

border:false,

header:[

// {

// name:"", 表头显示值

// key:"",

// isDrill:false, 是否钻取

// width:150, 宽度

// text:false, 钻取显示文本

// fixed:false, 左右对其

// hidden:false 是否隐藏

// sort:false 是否排序

// exporthidden:false 导出隐藏

// }

],

data:[],

pagination:{

pageIndex:1,

pageRowSize:10,

total:0

},

button:[

// {

// id:"",

// name:"",

// export:false,

// option:{

// url:"",

// params:{},

// fileName:""

// }

// }

],

selectOption:"",

},

VirtualList 虚拟列表

Api 文件夹下 封装一些公共接口

四 代码规范 （https://github.com/BingKui/javascript-zh）

1. 每个.vue文件不能超过400行，按块分离组件，minix 抽离代码块js
2. 组件命名以双大写开头例ElementTable
3. Style方法class 名 用 -连接 例.class-fisrt-name
4. If....else 多种判断

If(val == ‘a’) {} else if(val == ‘b’){} else if(val == ‘c’){}

改写成

let option = {

‘a’:()=>{},

‘b’:()=>{},

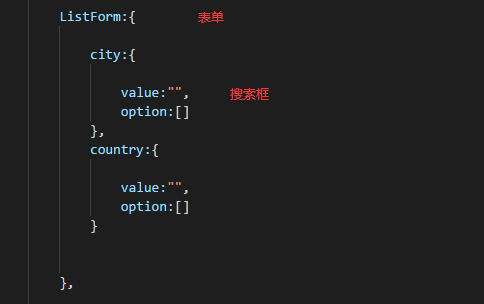
‘c’:()=>{}

}

Option[val]();

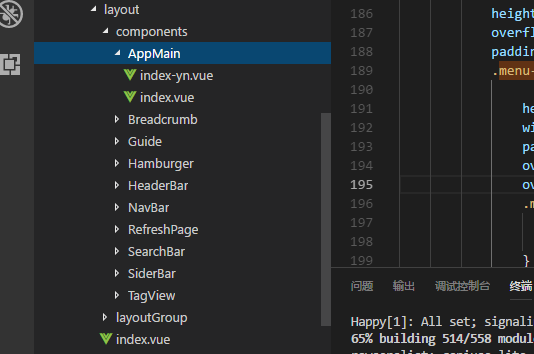
多枚举清空 用配置文件引入格式进行处理

1. 多层处理逻辑写清注释
2. Data 定义对象使用嵌套模式



1. 页面文件命名禁止使用拼音
2. 省份判断 定制化分为 common（公共） 各省份的 按单独文件夹文件拆分

如子应用建立路由 需在文件夹名上加 地市缩写



1. 各省份配置文件 (公共组件头，子应用api)

