**前序准备**

你需要在本地安装 [node](http://nodejs.org/) 和 [git](https://git-scm.com/)

本项目技术栈基于 [ES2015+](http://es6.ruanyifeng.com/)、[vue](https://cn.vuejs.org/index.html)、[vuex](https://vuex.vuejs.org/zh-cn/)、[vue-router](https://router.vuejs.org/zh-cn/) 、[vue-cli](https://github.com/vuejs/vue-cli) 、[axios](https://github.com/axios/axios) 和 [element-ui](https://github.com/ElemeFE/element)

## 安装

请先安装淘宝镜像[<https://developer.aliyun.com/mirror/NPM?from=tnpm>](http://nodejs.org/)

第三方包安装：npm install 淘宝镜像安装：cnpm install

本地运行：npm run dev

构建生产环境：npm run build:prod

## 关于第三方包无法正确安装与版本兼容问题

大部分原因都是网络不稳导致的掉包，所以请先正确安装淘宝镜像

暂时只发现core-js版本存在兼容问题，如在下拉完第三包后存在此问题，请单独安装高版本cnpm install core-js@2

## 目录结构

├── build # 构建相关

├── mock # 项目mock 模拟数据

├── plop-templates # 基本模板

├── public # 静态资源

│ │── favicon.ico # favicon图标

│ └── index.html # html模板

├── src # 源代码

│ ├── api # 所有请求

│ ├── assets # 主题 字体等静态资源

│ ├── components # 全局公用组件

│ ├── directive # 全局指令

│ ├── filters # 全局 filter

│ ├── icons # 项目所有 svg icons

│ ├── Lang # 国际化 language

│ ├── layout # 全局 layout

│ ├── router # 路由

│ ├── store # 全局 store管理

│ ├── styles # 全局样式

│ ├── utils # 全局公用方法

│ ├── vendor # 公用vendor

│ ├── views # views 所有页面

│ ├── App.vue # 入口页面

│ ├── main.js # 入口文件 加载组件 初始化等

│ └── permission.js # 权限管理

├── tests # 测试

├── .env.xxx # 环境变量配置

├── .eslintrc.js # eslint 配置项

├── .babelrc # babel-loader 配置

├── .travis.yml # 自动化CI配置

├── vue.config.js # vue-cli 配置

├── postcss.config.js # postcss 配置

└── package.json # package.json

## 路由

路由是添加新页面的基础，尤其是制作项目导航栏都需通过配置路由，然后渲染成页面导航布局

我们的项目确切的说存在三种路由导航，顶部导航、左侧导航、详情页导航

路由转换成的最终地址为各级路由path拼接的结果，例如以下电子签章下的配置页面的fullPath为/apps/ca/config

Icon: icon均为svg文件，全部保存在icons文件夹，后期新增添加至此，然后引用即可

顶部导航：顶部导航的配置比较常规,以下配置将会生成一个顶部导航，其中Layout与 @/views/pages/apps/inde 均为 <router-view />，仅仅作为页面承载

{

path: '/apps',

component: Layout,

redirect: '/apps/ca/config',

name: 'apps',

meta: {

title: '应用',

icon: 'lock'

},

children: [

{

path: 'ca',

component: () => import('@/views/pages/apps/index'), // <router-view /> 承载子菜单

redirect: '/apps/ca/config', // 重定向 一般指向子菜单的第一个页面

name: 'ca',

meta: {

title: '电子签章'

},

children: [

{

path: 'config',

component: () => import('@/views/pages/apps/ca/config'),

name: 'config',

meta: {

title: '配置'

}

}

]

},

{

path: 'invoice',

component: () => import('@/views/pages/apps/index'),

redirect: '/apps/invoice/config',

name: 'invoice',

meta: {

title: '发票验证'

},

children: [

{

path: 'config',

component: () => import('@/views/pages/apps/invoice/config'),

name: 'config',

meta: {

title: '配置'

}

}

]

},

{

path: 'sms',

component: () => import('@/views/pages/apps/index'),

redirect: '/apps/sms/config',

name: 'sms',

meta: {

title: '短信'

},

children: [

{

path: 'config',

component: () => import('@/views/pages/apps/sms/config'),

name: 'config',

meta: {

title: '配置'

}

}

]

},

{

path: 'mail',

component: () => import('@/views/pages/apps/index'),

redirect: '/apps/mail/config',

name: 'mail',

meta: {

title: '邮件'

},

children: [

{

path: 'config',

component: () => import('@/views/pages/apps/ca/config'),

name: 'config',

meta: {

title: '配置'

}

}

]

}

]

}

左侧导航 - leftSidebar

左侧导航配置的关键在于leftSidebar字段的配置，其中父级和子级均需配置相同leftSidebar

{

path: '/factoring',

component: Layout,

redirect: '/factoring/survey',

name: 'factoring',

meta: {

title: '保理',

icon: 'lock'

},

children: [

{

path: 'survey',

component: () => import('@/views/pages/factoring/survey'),

name: 'survey',

meta: {

title: '概况'

}

},

{

path: 'credit',

component: () => import('@/views/pages/factoring/index'),

redirect: '/factoring/credit/mgr',

leftSidebar: '/factoring/credit',

name: 'credit',

meta: {

title: '授信'

},

children: [

{

path: 'mgr',

component: () => import('@/views/pages/factoring/credit/mgr'),

leftSidebar: '/factoring/credit',

name: 'mgr',

meta: {

title: '授信管理'

}

},

{

path: 'quota',

component: () => import('@/views/pages/factoring/credit/quota'),

leftSidebar: '/factoring/credit',

name: 'quota',

meta: {

title: '额度管理'

}

},

{

path: 'chg',

component: () => import('@/views/pages/factoring/credit/chg'),

leftSidebar: '/factoring/credit',

name: 'chg',

meta: {

title: '授信变更'

}

}

]

}

]

}

详情页导航 - tabsSidebar

详情页导航配置关键在于tabsSidebar字段的配置，其中父级和子级均需配置相同tabsSidebar

以下为授信管理查看的详情页面，需要注意的是它和信管理查是同级的

{

path: 'mgr-detail',

component: () => import('@/views/pages/factoring/index'),

hidden: true,

tabsSidebar: '/factoring/credit/mgr-detail',

redirect: '/factoring/credit/mgr-detail/basic-info',

name: 'mgr-detail',

meta: {

title: '授信管理查看'

},

children: [

{

path: 'basic-info',

component: () => import('@/views/pages/factoring/credit/mgr/mgr-detail/basic-info'),

tabsSidebar: '/factoring/credit/mgr-detail',

name: 'basic-info',

meta: {

title: '基本信息'

}

},

{

path: 'gua-info',

component: () => import('@/views/pages/factoring/credit/mgr/mgr-detail/gua-info'),

tabsSidebar: '/factoring/credit/mgr-detail',

name: 'gua-info',

meta: {

title: '担保信息'

}

},

{

path: 'credit-ft',

component: () => import('@/views/pages/factoring/credit/mgr/mgr-detail/credit-ft'),

tabsSidebar: '/factoring/credit/mgr-detail',

name: 'credit-ft',

meta: {

title: '保理信息'

}

},

{

path: 'receivables',

component: () => import('@/views/pages/factoring/credit/mgr/mgr-detail/receivables'),

tabsSidebar: '/factoring/credit/mgr-detail',

name: 'receivables',

meta: {

title: '应收账款'

}

},

{

path: 'credit-history',

component: () => import('@/views/pages/factoring/credit/mgr/mgr-detail/credit-history'),

tabsSidebar: '/factoring/credit/mgr-detail',

name: 'credit-history',

meta: {

title: '授信记录'

}

},

{

path: 'credit-file',

component: () => import('@/views/pages/factoring/credit/mgr/mgr-detail/credit-file'),

tabsSidebar: '/factoring/credit/mgr-detail',

name: 'credit-file',

meta: {

title: '档案资料'

}

}

]

}

## API

后台请求步骤

第一步 对axios的封装，主要是对baseURL，响应时间，请求拦截器，响应拦截器的封装 src>utils>request.js

第二步 对请求方法的封装，主要对参数的处理与值的返回的处理和一些二次确认等的集中处理 src>api> restmodel\_hz.js

第三步 对各个模块单独封装，对各个接口与传参以及二次确认提示文件 src>api>模块>index.js

第四步 调用 页面引用相关模块请求接口获取返回的数据

## 全局样式

src>styles

样式编写推荐使用scss

注意：页面并不是通过定义一个全局class名称，然后所有页面使用这个class来实现一个class对应一个模块样式的定义，这种方式对应全局规范的要求过于严苛，暂不推荐（想要以此方式来实现全局样式请自定义class，class名称不能用通用名称，避免其他人贪图方便直接使用了这个class导致污染全局）

通用class名称：main container header foot wrap item (后续补充…)

正文：样式也是模块化的，导航是个模块，详情是个模块，编辑页也是模块

样式模块化的关键是样式私有化 关键字-scoped， 通过vue中的scope可以实现组件的私有化

## 开发步骤

第一步：编写静态模块（先做占位使用）以保理（credit）模块为例

├── src # 源代码

│ ├── views # 窗口代码

│ ├── ├── pages # 页面

│ ├── ├── ├── factoring # 保理模块

│ ├── ├── ├── ├── credit # 保理子模块 – 授信

│ ├── ├── ├── ├── ├── mgr # 授信查看页面

│ ├── ├── ├── ├── ├── ├── mgr-detail # 授信 – 查看

│ ├── ├── ├── ├── ├── ├── ├── basic-info # 查看 – 基础信息

│ ├── ├── ├── ├── ├── ├── ├── credit-file # 查看 – 档案资料

│ ├── ├── ├── ├── ├── ├── ├── credit-ft # 查看 – 保理信息

│ ├── ├── ├── ├── ├── ├── ├── credit-history # 查看 – 授信记录

│ ├── ├── ├── ├── ├── ├── ├── gua-info # 查看 – 担保信息

│ ├── ├── ├── ├── ├── ├── ├── receivables # 查看 – 应收账款

│ ├── ├── ├── ├── ├── ├── mgr-editor # 授信 – 编辑

│ ├── ├── ├── ├── ├── ├── ├── basic-info # 编辑 – 基础信息

│ ├── ├── ├── ├── ├── ├── ├── credit-file # 编辑 – 档案资料

│ ├── ├── ├── ├── ├── ├── ├── credit-ft # 编辑 – 保理信息

│ ├── ├── ├── ├── ├── ├── ├── credit-history # 编辑 – 授信记录

│ ├── ├── ├── ├── ├── ├── ├── gua-info # 编辑 – 担保信息

│ ├── ├── ├── ├── ├── ├── ├── receivables # 编辑 – 应收账款

│ ├── ├── ├── ├── ├── ├── index.vue # 授信管理列表

第二步：编写路由，以下是一个完整功能模块包含顶部导航，左侧导航，详情页导航

顶部导航：路由的第一级与第二级分别为顶部导航的一级菜单和二级菜单

左侧导航：左侧导航需要配置leftSidebar

leftSidebar配置规则：包含父级和子级，leftSidebar的值为父级的全路径，子级的值等于父级的值

详情页导航：详情页导航需配置tabsSidebar

tabsSidebar配置规则：包含父级和子级，tabsSidebar的值为父级的全路径，子级的值等于父级的值

{

path: '/factoring',

component: Layout,

redirect: '/factoring/survey',

name: 'factoring',

meta: {

title: '保理',

icon: 'lock' // icon均为svg文件，全部保存在icons文件夹，后期新增添加至此，引用即可生成顶部导航图标

},

children: [

{

path: 'survey',

component: () => import('@/views/pages/factoring/survey'),

name: 'survey',

meta: {

title: '概况'

}

},

{

path: 'credit',

component: () => import('@/views/pages/factoring/index'),

redirect: '/factoring/credit/mgr',

leftSidebar: '/factoring/credit', /// 授信列表页左侧导航配置 – 这是父级

name: 'credit',

meta: {

title: '授信'

},

children: [

{

path: 'mgr',

component: () => import('@/views/pages/factoring/credit/mgr'),

leftSidebar: '/factoring/credit', // 授信列表页左侧导航配置 – 这是子级

name: 'mgr',

meta: {

title: '授信管理'

}

},

{

path: 'mgr-detail',

component: () => import('@/views/pages/factoring/index'),

hidden: true,

tabsSidebar: '/factoring/credit/mgr-detail', // 授信详情页导航配置 – 这是父级

redirect: '/factoring/credit/mgr-detail/basic-info',

name: 'mgr-detail',

meta: {

title: '授信管理查看'

},

children: [

{

path: 'basic-info',

component: () => import('@/views/pages/factoring/credit/mgr/mgr-detail/basic-info'),

tabsSidebar: '/factoring/credit/mgr-detail', // 授信详情页导航配置 – 这是子级

name: 'basic-info',

meta: {

title: '基本信息'

}

},

……

]

},

{

path: 'mgr-editor',

hidden: true,

component: () => import('@/views/pages/factoring/index'),

redirect: '/factoring/credit/mgr-editor/basic-info',

leftSidebar: '/factoring/credit/mgr-editor', // 授信编辑页左侧导航配置 – 这是父级

tabsSidebar: false,

name: 'mgr-editor',

meta: {

title: '授信管理编辑'

},

children: [

{

path: 'basic-info',

component: () => import('@/views/pages/factoring/credit/mgr/mgr-editor/basic-info'),

leftSidebar: '/factoring/credit/mgr-editor', // 授信编辑页左侧导航配置 – 这是子级

name: 'basic-info',

meta: {

title: '基本信息'

}

}

]

}

]

}

]

}

第三步：编写接口

接口的编写需要对应模块

├── src # 源代码

│ ├── api # 所有请求

│ ├── ├── factoring # 保理

│ ├── ├── ├── credit # 授信模块接口封装类

│ ├── ├── ├── ├── index.js # 授信模块接口封装类

│ ├── ├── restmodel\_hz.js # 请求方法的封装，统一处理请求参数，二次确认，method等，授信模块接口需引用此模块的方法，

第四步：页面引用请求模块获取数据

## 权限控制­

权限控制分为菜单与按钮

菜单：通过动态路由配置roles（唯一标识码）与权限接口返回菜单路由的唯一标识码比对之后生成最终路由表

按钮：通过在每级菜单页面下获取接口返回值比对全套按钮最终将有权限的按钮通过v-if显示出来