Vue项目开发总结

目录结构：

**A：开发环境配置**

B：less插件安装

C：axios插件安装使用

**D：vuex插件安装使用**

**E：Element-ui的安装及使用**

**F：vue-awesome-swiper插件的安装及使用**

**G:IE兼容问题-VUEX兼容IE上的报错问题**

**H：axios请求post数据无法传到后台：**

**I:vue-router验证拦截路由**

**J:axios验证路由拦截**

**K:vuex使用案例**

**L:透明颜色背景设置**

**M:jquery的安装及使用**

**N:定义全局组件**

**O:cnpm run build后的dist文件名修改**

**P:vue的get与post请求**

**Q:vue配置公共api**

**R:CSS3实现字体超出隐藏**

**S:CSS3美化滚动条**

**T:vue引入jquery**

**U:设置API组件**

**V:vue项目架构搭建**

**W:路由router.js配置**

**A1:vue使用vuex**

**A：开发环境配置**

**# 全局安装 vue-cli**

**$ cnpm install --global vue-cli**

1. vue init webpack app
2. Cd app
3. Cnpm install或者yarn
4. Cnpm run dev或者yarn run dev
5. Cnpm run build或者yarn run build

**B：less插件安装**

1. 安装less和less-loader：

命令：cnpm install less less-loader --save-dev或者yarn add less less-loader

1. 配置less: 路径：build---webpack.base.conf.js添加

{

test: /\.less$/,

loader: 'style-loader!css-loader!less-loader'

}

**C：axios插件安装使用**

安装：cnpm install axios -S或者yarn add axios

全局使用：

首先在 main.js 中引入 axios

import axios from 'axios'

这时候如果在其它的组件中，是无法使用 axios 命令的。但如果将 axios 改写为 Vue 的原型属性，就能解决这个问题

Vue.prototype.$ajax = axios

//全局引入axios

import Axios from 'axios'

Vue.prototype.$http=Axios.create({

headers: {'content-type': 'application/x-www-form-urlencoded'}

});

**D：vuex插件安装使用**

安装：cnpm install vuex --save或者yarn add vuex

在src里面新建一个store的文件夹，在store里建立index.js

全局使用：

在index.js输入

import Vue from 'vue';

import Vuex from 'vuex';

Vue.use(Vuex);

/\*\*

\* Vuex全局状态管理

\* @param options {Array} 用于渲染tabs的数组

\*/

const store = new Vuex.Store({

state: {

options: [],

activeIndex: '/user',

userInfo: {}

},

mutations: {

// 添加tabs

add\_tabs (state, data) {

this.state.options.push(data);

},

// 删除tabs

delete\_tabs (state, route) {

let index = 0;

for (let option of state.options) {

if (option.route === route) {

break;

}

index++;

}

this.state.options.splice(index, 1);

},

// 设置当前激活的tab

set\_active\_index (state, index) {

this.state.activeIndex = index;

},

// 设置详情信息

save\_detail\_info (state, info) {

this.state.userInfo = info;

}

}

});

export default store;

再在main.js里输入

//引入vuex

import store from '@/store/index';

new Vue({

el: '#app',

router,

store,

components: { App },

template: '<App/>'

})

**E：Element-ui的安装及使用**

安装：cnpm i element-ui -S

yarn add element-ui -S

全局使用：

import Vue from 'vue'

import ElementUI from 'element-ui'

import 'element-ui/lib/theme-chalk/index.css'

Vue.use(ElementUI)

**F：vue-awesome-swiper插件的安装及使用**

安装：cnpm install vue-awesome-swiper --save

全局使用：

在main.js输入

import Vue from 'vue'

import VueAwesomeSwiper from 'vue-awesome-swiper'

引入static放置的css

@import url("../../static/css/swiper.min.css");

组件使用：

import { swiper, swiperSlide } from 'vue-awesome-swiper'

export default{

components:{

swiper,

swiperSlide

}

}

例子：

<template>

<swiper :options="swiperOption" ref="mySwiper" >

<!-- slides -->

<swiper-slide>I'm Slide 1</swiper-slide>

<swiper-slide>I'm Slide 2</swiper-slide>

<swiper-slide>I'm Slide 3</swiper-slide>

<swiper-slide>I'm Slide 4</swiper-slide>

<swiper-slide>I'm Slide 5</swiper-slide>

<swiper-slide>I'm Slide 6</swiper-slide>

<swiper-slide>I'm Slide 7</swiper-slide>

<!-- Optional controls -->

<div class="swiper-button-prev" slot="button-prev"></div>

<div class="swiper-button-next" slot="button-next"></div>

</swiper>

</template>

<script>

import { swiper, swiperSlide } from 'vue-awesome-swiper'

export default{

components:{

swiper,

swiperSlide

},

data (){

return{

swiperOption: {

navigation: {

nextEl: '.swiper-button-next',

prevEl: '.swiper-button-prev',

}

}

}

},

computed: {

swiper() {

return this.$refs.mySwiper.swiper

}

},

mounted() {

this.swiper.slideTo(3, 1000, false)

}

}

</script>

<style type="text/css">

@import url("../../static/css/swiper.min.css");

</style>

**F：echarts插件安装及使用**

安装：cnpm install echarts -S或者yarn add echarts -S

全局使用：

在main.js输入

// 引入echarts

import echarts from 'echarts'

Vue.prototype.$echarts = echarts

例子：

<template>

<div id="myChart"></div>

</template>

<script>

export default {

data () {

return {

}

},

mounted(){

this.drawLine();

},

methods: {

drawLine(){

// 基于准备好的dom，初始化echarts实例

let myChart = this.$echarts.init(document.getElementById('myChart'));

let option = {

title : {

text: '符允的echart',

subtext: '纯属虚构',

x:'center'

},

tooltip : {

trigger: 'item',

formatter: "{a} <br/>{b} : {c} ({d}%)"

},

legend: {

orient : 'vertical',

x : 'left',

data:['直接访问','邮件营销','联盟广告','视频广告','搜索引擎']

},

toolbox: {

show : true,

feature : {

mark : {show: true},

dataView : {show: true, readOnly: false},

magicType : {

show: true,

type: ['pie', 'funnel'],

option: {

funnel: {

x: '25%',

width: '50%',

funnelAlign: 'left',

max: 1548

}

}

},

restore : {show: true},

saveAsImage : {show: true}

}

},

calculable : true,

series : [

{

name:'访问来源',

type:'pie',

radius : '55%',

center: ['50%', '60%'],

data:[

{value:335, name:'直接访问'},

{value:310, name:'邮件营销'},

{value:234, name:'联盟广告'},

{value:135, name:'视频广告'},

{value:1548, name:'搜索引擎'}

]

}

]

};

// 绘制图表

myChart.setOption(option);

}

}

}

</script>

<style>

#myChart{

width: 50%;

height: 600px;

}

</style>

**G:IE兼容问题：VUEX兼容IE上的报错问题**

错误提示：SCRIPT5022: [vuex] vuex requires a Promise polyfill in this browser.

解决办法：

1，先安装babel-polyfill

cnpm install babel-polyfill --save-dev或者yarn add babel-polyfill

2，在webpack.base.config.js文件中，使用

module.exports = {

entry: {

app: ["babel-polyfill", "./src/main.js"]

}

};

替换

module.exports = {

entry: {

app: './src/main.js'

}

}

module.exports = {

context: path.resolve(\_\_dirname, '../'),

entry: {

app: ["babel-polyfill", "./src/main.js"]

},

//entry: {

// app: './src/main.js'

//},

**H：axios请求post数据无法传到后台：**

let qs=require('qs');

let params={

'username':self.ruleForm.username,

'password':self.ruleForm.password

}

let instance = self.$http.create({

headers: {'content-type': 'application/x-www-form-urlencoded'}

});

instance.post(BaseUrl+'/loginRest',qs.stringify(params))

**I:vue-router验证拦截路由**

解决方案：

在router的index.js

export default router

router.beforeEach((to, from, next) => {

if (to.matched.some(res => res.meta.requireAuth)) {// 判断是否需要登录权限

if (store.state.token) {// 判断是否登录

next()

} else {// 没登录则跳转到登录界面

next({

path: '/login',

query: {redirect: to.fullPath}

})

}

} else {

next()

}

})

在main.js：

//引入axios

import axios from 'axios'

// http request 拦截器

axios.interceptors.request.use(

config => {

if (store.state.token) { // 判断是否存在token，如果存在的话，则每个http header都加上token

config.headers['Content-Type']='application/x-www-form-urlencoded';

config.headers['token']=store.state.token;

}else{

config.headers['Content-Type']='application/x-www-form-urlencoded';

}

return config;

},

err => {

return Promise.reject(err);

});

// http response 拦截器

axios.interceptors.response.use(

response => {

return response;

},

error => {

if (error.response) {

switch (error.response.status) {

case 401:

store.commit(types.LOGOUT);

router.replace({

path: '/login',

query: {redirect: router.currentRoute.fullPath}

})

}

}

return Promise.reject(error.response.data) // 返回接口返回的错误信息

});

Vue.prototype.$http=axios

**K:vuex使用案例**

创建vuex文件夹再创建store.js

import Vue from 'vue'

import Vuex from 'vuex'

// 引入 axios

import axios from 'axios'

Vue.use(Vuex)

const store=new Vuex.Store({

state:{author:''},

mutations:{

newAuthor(state,msg){

state.author=msg

}

},

actions:{

}

})

export default store

在具体组件中提交

this.$store.commit('newAuthor',this.selectInput)

在具体组件中引入store.state

<p>{{this.$store.state.author}}</p>

**L:透明颜色背景设置**

设置方法：background: rgba(0, 0, 0,0.5);

**M:jquery的安装及使用**

**安装：vue-cli webpack 引入jquery**

**配置：**

**首先在package.json里的dependencies加入"jquery" : "^2.2.3"，然后npm install或者用npm install --registry=http://registry.npm.taobao.org安装的会更快一点**

**在webpack.base.conf.js里加入**

**var webpack = require("webpack")**

**在module.exports的最后加入**

**plugins: [**

**new webpack.optimize.CommonsChunkPlugin('common.js'),**

**new webpack.ProvidePlugin({**

**jQuery: "jquery",**

**$: "jquery"**

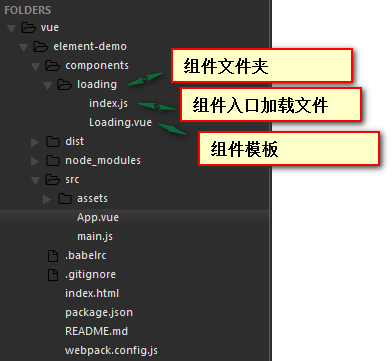
**})**

**]**

**然后一定要重新npm run dev**

**在main.js 引入就ok了import $ from 'jquery'**

**N:定义全局组件**

****

1. **在main.js**

**// 引入自定义组件。index.js是组件的默认入口**

**import Loading from '../components/loading'**

**Vue.use(Loading);**

1. **在Loading.vue里面定义自己的组件模板**

**<template>**

**<div class="loading">**

**loading...**

**</div>**

**</template>**

1. **index.js文件里面添加install方法**

**import MyLoading from './Loading.vue'**

**// 这里是重点**

**const Loading = {**

**install: function(Vue){**

**Vue.component('Loading',MyLoading)**

**}**

**}**

**// 导出组件**

**export default Loading**

1. **App.Vue里面使用组件**

**<template>**

**<div id="app">**

**<!-- 使用element ui的组件 -->**

**<el-button>默认按钮</el-button>**

**<!-- 使用自定义组件 -->**

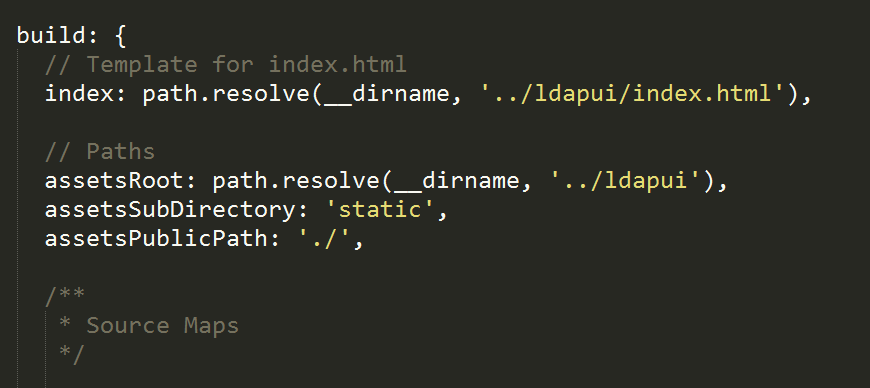
**<Loading></Loading>**

**</div>**

**</template>**

**O:cnpm run build后的dist文件名修改**

**方法：在config的index.js里修改**

****

P:vue的get与post请求

一，在main.js设置

//引入axios

import axios from 'axios'

//axios.defaults.withCredentials = true

Vue.prototype.$http=axios.create({

headers: {'content-type': 'application/x-www-form-urlencoded'}

});

二，get请求

self.$http.get(self.COMMON\_URL.URL\_199+'/runSystem/getRunSystemByProductId',{

params:{productId:productId}

}).then()；

三，post请求

let qs=require('qs');

let params={

'taskId':self.taskId

};

self.$http.post(self.COMMON\_URL.baseApi+'8011/buildController/doVersionBuild',qs.stringify(params)).then()

**Q:vue配置公共api**

**一，在Common\_url里写**

**<script>**

**const baseApi='http://10.4.5.198';**

**const homeApi='http://10.4.5.198';**

**export default {**

**baseApi,homeApi**

**}**

**</script>**

**二，在main.js里设置**

**//访问路径公共组件**

**import common\_url from './Common\_url'**

**Vue.prototype.COMMON\_URL = common\_url**

**R:CSS3实现字体超出隐藏**

**overflow: hidden;**

**text-overflow: ellipsis;**

**white-space: nowrap;**

**R:查看vue版本**

**npm vue -v**

**S:CSS3美化滚动条**

**/\* // 滚动条样式 \*/**

**/\*滚动条 start\*/**

**::-webkit-scrollbar {**

**width: 6px;**

**}**

**/\*定义滚动条轨道 内阴影+圆角\*/**

**::-webkit-scrollbar-track {**

**-webkit-box-shadow: inset 0 0 1px rgba(0, 0, 0, 0.3);**

**background: #fff;**

**}**

**/\*定义滑块 内阴影+圆角\*/**

**::-webkit-scrollbar-thumb {**

**border-radius: 3px;**

**-webkit-box-shadow: inset 0 0 6px rgba(0, 0, 0, 0.3);**

**/\* // background-color: rgba(7, 170, 247, 0.7); \*/**

**background-color: #eee;**

**}**

**::-webkit-scrollbar-thumb:hover {**

**border-radius: 3px;**

**-webkit-box-shadow: inset 0 0 6px rgba(0, 0, 0, 0.3);**

**background-color: #aaa;**

**}**

**/\* // 滚动条 end \*/**

**T:vue引入jquery**

1. npm install jquery –save 安装jquery依赖
2. 修改2处webpack.base.conf.js文件

在webpack.base.conf.js里加入

var webpack = require("webpack")

在module.exports的最后加入

plugins: [

new webpack.optimize.CommonsChunkPlugin('common.js'),

new webpack.ProvidePlugin({

jQuery: "jquery",

$: "jquery"

})

]

1. 在 main.js里，直接 import $ from ‘jquery’
2. Cnpm install

**U:设置API组件**

在src里新建一个API.vue

在main.js输入

//引入api

import Api from './Api'

Vue.prototype.$api=Api

<script>

const baseApi= 'http://10.2.6.198:8015/runsystem'; // 运行系统服务

const commonApi = 'http://10.2.6.198:8015/version'; //版本管理服务

export default {

baseApi,

commonApi

}

</script>

再在具体vue里引入api

$api.baseApi

**V:vue项目架构搭建**

在src里的components建立

common 公共组件库

own 独立但不可复用的小组件库

page 大的页面组件库

reuse 可复用的小组件库

1. vue 404

在common里建立Head.vue，Login.vue

**W:路由router.js配置**

**A1:vue使用vuex**