**Vue2+vue-router2+webpack开发项目**

使用vue init webpack my-project新建项目 >>>>> npm install >>>>>> npm run dev 启动项目

组件使用：

<Header></Header> 使用组件

Impor Header from ‘./components/header’ 引入组件

export default{

components:{Header} //申明

}

**路由使用（router/index.js）：**

import Vue from ‘vue’

import vueRouter from ‘vue-router’

Vue.use(‘vueRouter’)

import Helloworld from ‘@/components/Helloworld’ // 引入组件

export default new Router({

routes:[

{path:’/’,redirect:{name:Helloword}}, // 重定向到 名为Helloworld 路由

{path:’/detail/:id’,components:Detaill,name:’details’},

{

Path:’/login’,

Name:’Login’, // 路由名称

Components:resolve=>{

Require([‘../components/login’],resolve)

}

}

})

**Main.js:**

// 引用工具文件 index.js 中方法名：fortime

import api from ‘./api/index.js’

// 将工具方法绑定到全局

Vue.prototype.$api=api

$api.fortime(‘132165462136’)

**暴露接口方法( 可以暴露多个接口 )**

export default{

fortime(time){

~~~~~~~~

~~~~~~~~

return -----

},

formatTime(time,type){

// var Times=new Date(time).getTime() 不知为何会引起下面 getFullYear() 报错

var Times=new Date(time);

var year=Times.getFullYear();

var month=Times.getMonth()+1;

month=month<10?'0'+month:month;

var day=Times.getDate();

day=day<10?'0'+day:day;

var hour=Times.getHours();

var minu=Times.getMinutes();

var sec=Times.getSeconds();

hour=hour<10?'0'+hour:hour;

minu=minu<10?'0'+minu:minu;

sec=sec<10?'0'+sec:sec;

var newTime='';

if (type=='yyyy-mm-dd') {

newTime+=year+"-"+month+"-"+day+" "+hour+":"+minu+":"+sec;

}else if (type=='yyyy/mm/dd') {

newTime+=year+"/"+month+"/"+day+" "+hour+":"+minu+":"+sec;

}

return newTime;

}

}

调用方式： $utils.formatTime(item.createTime,”yyyy-dd-mm”)

**获取 url 中的id**

this.$route.params.id

this.$api.get(‘topic/’+this.id,null,r=>{

console.log(r.data)

this.objs=r.data

}

**改变路由**

methods:{

goRouter(routerName){

this.$router.push({name:routerName})

}

}

**config--> index.js**

// 接口代理到本地

proxyTable: {

'/api/v1/\*\*': {

target: 'https://cnodejs.org', // 你接口的域名

secure: false,

changeOrigin: false,

}

},

// 打包时防止生成 .map 的文件config>>index.js下

productionSourceMap: false, // true 改成false

**运行打包命令：**

npm run build （打包完直接打开页面是不行的，只能看到加载的静态资源，需要启动一个HTTP服务，然后访问到打包后dist目录下的文件才能看到完整的项目 方法如下：）

1. 在根目录下 建server.js

var express = require('express');//获取express

var config = require('./config/index');//获取根目录config下的index.js

var port = process.env.PORT || config.build.port;//打开的端口号等于当前环境或上面index.js里增加的那个port

var app = express();

var router = express.Router();//express路由

router.get('/', function (req, res, next) {//返回index.html

req.url = '/index.html';

next();

});

app.use(express.static('./dist'));//使用静态文件夹下的文件

module.exports = app.listen(port, function (err) {//监听port端口

if (err) {

console.log(err);

return;

}

console.log('Listening at http://localhost:' + port + '\n')

});

1. 进入根目录文件夹，使用node启动http服务 node server.js
2. 通过localhost:port 访问就可以看到项目了

安装 http-server 启动 http 服务

进入 dist 文件夹，然后启动一个 http 服务

在终端里面输入 http-server

**Cookie 相关方法 cookie.js**

import Vue from 'vue'

Vue.prototype.setCookie = function (name, value, day) {

let date = new Date

date.setTime(date.getTime() + 24\*60\*60\*1000\*day);

window.document.cookie = name + "=" + value + ";path=/;expires=" + date.toGMTString();

}

Vue.prototype.getCookie = function (name) {

let v = window.document.cookie.match('(^|;) ?' + name + '=([^;]\*)(;|$)');

return v ? v[2] : null

}

Vue.prototype.delCookie = function (name) {

let date = new Date

date.setTime(date.getTime() - 1)

let val = this.getCookie(name)

if(val != null){

document.cookie= name + "="+val+";expires="+date.toGMTString();

}

}

this.setCookie('loginState',true,1)

vue.getCookie('loginState')

// 这灵活得亮瞎了

/\*new Vue({

el: '#app',

template: '<App/>',

router,

components: { App }

});

new Vue(Vue.util.extend({

router

}, App)).$mount('#app');

new Vue({

el:'#app',

router,

render:h => h(App)

});\*/

var app = new Vue({

el: '#app',

router,

store,

...App,

});

# **Vue通信、传值的多种方式，详解（都是干货）**

https://blog.csdn.net/qq\_35430000/article/details/79291287

**Vue-vuex基础知识的学习总结**

**https://blog.csdn.net/zhq2005095/article/details/66477902**

<!-- 这个链接只会在地址为 / 的时候被激活 --> <router-link to="/" exact>

.router-link-active

声明式：<router-link :to="...">

编程式：router.push(...)

声明式：<router-link :to="..." replace>

编程式：router.replace(...)

this.$router.go(-1)

Node 项目 用 npm start 启动

Vue:command no fund 解决办法： C:\Users\user\AppData\Roaming\npm

可以看出：更改state的数据并显示在组件中，有几个步骤：1. 在mutations选项里，注册函数 函数里面装逻辑代码。2.在组件里，this.$store.commit('change',payload)  注意：提交的函数名要一一对应  3.触发函数，state就会相应更改 4.在组件的**计算属性**里 this.$store.state 获取你想要得到的数据

https://www.cnblogs.com/first-time/p/6815036.html

**关闭ESLint方法： 在vue项目中找到build文件夹 --> webpack.base.conf.js --> module**

**module: {**

**rules: [**

**// 把 eslint 关闭**

**//...(config.dev.useEslint ? [createLintingRule()] : []),**

**然后，重启服务**

dispatch 带多个参数（默认2个）

this.$store.dispatch('savePlan', {plan:plan,id:this.$route.params.id})

Actions: savePlan({ commit }, {plan:plan,id:id}) {

commit(types.SAVE\_PLAN, {plan:plan,id:id});

}

Mutations: [types.SAVE\_PLAN] (state, {plan, id})

### **在组件中提交mutations**

( https://blog.csdn.net/zhq2005095/article/details/66477902 )

方法1: 在组件的methods中提交

methods: { increment(){ this.$store.commit('increment'); } }

方法2: 使用mapMutaions

用 mapMutations 辅助函数将组件中的 methods 映射为 store.commit 调用。

import { mapMutaions } from 'vuex';

export default {

// ...

methods: {

...mapMutaions([

'increment' // 映射 this.increment() 为 this.$store.commit('increment')

]),

...mapMutaions([

add: 'increment' // 映射 this.add() 为 this.$store.commit('increment')

])

}

}

// 因为mutation相当于一个method，所以在组件中，可以这样来使用

<button @click="increment">+</button>

## **4.5 actions** action是异步的

因为mutations中只能是同步操作，但是在实际的项目中，会有异步操作，那么actions就是为了异步操作而设置的。这样，就变成了在action中去提交mutation，然后在组件的methods中去提交action。只是提交actions的时候使用的是dispatch函数，而mutations则是用commit函数

一个简单的action

const store = new Vuex.Store({

state: {

count: 0

},

mutations: {

increment(state){

state.count++;

}

},

actions: {

increment(context){

context.commit('increment');

}

/\* 可以用参数结构的方法来写action

increment({commit}){

commit('increment');

}

\*/

}

});

// action函数接受一个context参数，这个context具有与store实例相同的方法和属性。

// 分发action

store.dispatch('increment');

4.5.3 组合actions

因为action是异步的，那么我们需要知道这个异步函数什么时候结束，以及等到其执行后，会利用某个action的结果。这个可以使用promise来实现。在一个action中返回一个promise，然后使用then()回调函数来处理这个action返回的结果。

actions:{

actionA({commit}){

return new Promise((resolve, reject) => {

setTimeout(() => {

commit('someMutation');

resolve();

},1000);

})

}

}

// 这样就可以操作actionA返回的结果了

store.dispatch('actionA').then(() => {

// dosomething ...

});

// 也可以在另一个action中使用actionA的结果

actions: {

// ...

actionB({ dispatch, commit }){

return dispatch('actionA').then(() => {

commit('someOtherMutation');

})

}

}

5 mudules

module是为了将store拆分后的一个个小模块，这么做的目的是因为当store很大的时候，分成模块的话，方便管理。

5.1 每个module拥有自己的state, getters, mutation, action

const moduleA = {

state: {...},

getters: {...},

mutations: {....},

actions: {...}

}

const moduleB = {

state: {...},

getters: {...},

mutations: {....},

actions: {...}

}

const store = new Vuex.Store({

modules: {

a: moduleA,

b: moduleB

}

});

store.state.a // 获取moduleA的状态

store.state.b // 获取moduleB的状态

5.2 模块内部的状态

对于模块内部的mutation和getter，接受的第一个参数是模块的局部状态state。顺便说一下，根结点的状态为rootState。

const moduleA = {

state: { count: 0},

getters: {

doubleCount(state){

return state.count \* 2;

}

},

mutations: {

increment(state){

state.count ++ ;

}

},

actions: {...}

}

6 含有vuex的项目的结构

6.1 应该遵循的规则

应用层级的状态都应该集中在store中

提交 mutation 是更改状态state的唯一方式，并且这个过程是同步的。

异步的操作应该都放在action里面

7 表单中使用vuex

在表单中，会使用v-model去进行双向绑定，而在vuex中，修改状态只能是通过提交mutation这种方式，所以在使用了严格模式下，这样就会抛出一个错误。

**解决方法1**: 给input绑定一个value，然后监听input或者onchange事件，通过e.target.value来获取真实的值，将这个值赋给store的state的对应的对象。那么这个过程是在mutation函数里面完成的，然后，**在组件中的methods中，提交mutation，完成这个赋值过程**。实例如下：

<input :value="message" @input="updatemessage"/>

const store = new Vuex.Store({

state: {

obj: { message: '' }

},

mutations: {

updatemessage(state, value){

state.obj.message = value;

}

}

});

export default {

// ...

computed: {

...mapState({

message: state => state.obj.message;

})

},

methods: {

updatemessage (e) {

this.$store.commit('updatemessage', e.target.value);

}

}

}

**解决方法2**：使用带有setter的双向绑定的计算属性.

<input v-model="message"/>

// ...

computed: {

message: {

get () {

return this.$store.state.obj.message;

},

set (value) {

this.$store.commit('updatemessage', value);

}

}

}

new Array(10).fill(obj)

min-height:calc(100vh - 319px)

<template slot-scope="scope">

<i class="el-icon-time"></i>

<span style="margin-left: 10px">{{ scope.row.date }}</span>

</template>

<span class="active" index="1" ref="setPhoneEmail">1.手机/邮箱验证</span>

<span index="2" ref="goSetPassword">2.设置密码</span>

**this.$refs.goSetPassword.classList.add('active');**

**this.$refs.setPhoneEmail.classList.remove('active');**

**第一条命令是追加环境变量，第二个命令是，使我们的追加立即生效。**

**echo 'export PATH="$PATH:/usr/local/Cellar/node/7.6.0/bin/"' >> ~/.bash\_profile**

**. ~/.bash\_profile**

Nginx 操作

Rm home.zip // 删除文件

Rm -rf \* // 删除所有

rz 上传

unzip dist.zip

http://47.94.218.115/wenlinqing/dist/#/red

//登录 函数

submituUserInfo(data){

this.$store.dispatch('Logins',data).then(res => {

this.$router.push({ path: '/' });

}).catch(() => {

})

}

logout(){

this.$store.dispatch('Logout').then(() => {

this.$router.push({ path: '/login' });

}).catch(err => {

this.$message.error(err);

});

}

<el-menu :default-active="$route.path" class="nav\_box" background-color="#545c64" text-color="#fff" active-text-color="#ffd04b" >

filters:{

mktime(t){

let now=new Date();

let diff=Math.floor((now-t)/1000)

if (diff<60) {

return '刚刚'

}else if (diff<3600) {

return Math.floor(diff/60)+'分钟前'

}else if (diff<86400) {

return Math.floor(diff/3600)+'小时前'

}else {

let oDate=new Date(t)

return `${oDate.getFullYear()} -${oDate.getMonth+1}-${oDate.getDate()} `

}

}

}

let data=await( await fetch(`http://localhost:8080/detail?id=${arg}`) ).json()

components:{

articleDetail:import './article\_detail', // (懒加载) 当前模块专用 不跟主模块一起编译

Header,// 跟主模块一起编译

Footer,

},

computed:{

coinData(){

return this.$store.state.coinList;

}

},

<el-switch v-model="item.isOpen" @change="switchChange($event,item.id)"></el-switch>

Axios.get(‘a.txt’).then(function(res){

Console.log(res);

This.message=res.data;

}.bind(this)).catch(function(err){

Console.log(err)

})

<LoadingMore @click.native=”loadingmore” />

.fade-transition{

transition: 1s all ease;

}

.fade-enter{

opacity: 0;

}

.fade-leave{

opacity: 0;

transform: translateX(200px);

}

<div id="div1" v-show="bSign" transition="fade"></div>

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

if (to.meta.requireAuth) {

next({path:'/login'})

}

next();

})

ggetUrlParam: function (name) {

return decodeURIComponent((new RegExp('[?|&]'+name+'='+'([^&;]+?)(&|#|;|$)').exec(location.href)||[,""])[1].replace(/\+/g,'%20'))||null;

},

Vue.filter('breviary',function(value,len){

value=value||'';

if (value.length>len) {

var obj1=value.substring(0,len);

var obj2=value.substring(value.length-len)

return obj1+"..."+obj2

}else{

return value

}

}) {{mainObj.chainAddr|breviary(10)}}

import { Toast } from 'mint-ui';

Toast('复制成功');

import Vue from 'vue';

import VueClipboard from 'vue-clipboard2';

Vue.use(VueClipboard);

methods:{

onCopy(e){

Toast('复制成功');

},

// 复制失败

onError(e){

alert("失败");

},

**qrcode (obj) {**

**var canvas = document.getElementById('canvas')**

**var that=this;**

**QRCode.toCanvas(canvas, JSON.stringify(obj), function (error) {**

**if (error) console.error(error)**

**console.log('success!');**

**})**

**},**

**}**

**import Vue from 'vue';**

**import QRCode from 'qrcode';**

**Vue.use(QRCode);**

**<canvas id="canvas"></canvas>**

mounted(){

this.qrcode(‘fgsdfgsfgsdf’);

},

import { Toast, Loadmore} from 'mint-ui';

Vue.prototype.$toast = Toast;

Vue.component(Loadmore.name, Loadmore);

import 'mint-ui/lib/style.css';

import elementUi from 'element-ui';

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

Vue.use(elementUi);

//highcharts的引入

/\*import Highcharts from 'highcharts';

Vue.use(Highcharts);

\*/

import api from '@/api/index';

Vue.prototype.$api=api;

import \* as cookie from '@/api/cookie';

Vue.prototype.$cookie=cookie;

<head-top :head-title="cityname" go-back='true'>

<router-link to="/home" slot="changecity" class="change\_city">切换城市</router-link>

</head-top>

props: ['signinUp', 'headTitle', 'goBack'],

<template>

<slot name="changecity"></slot>

</template>

async beforeMount(){}

<svg width="4rem" height="100%" xmlns="http://www.w3.org/2000/svg" version="1.1">

<polyline points="12,18 4,9 12,0" style="fill:none;stroke:rgb(0,0,0);stroke-width:3"/>

</svg>

<svg width="14" height="14" xmlns="http://www.w3.org/2000/svg" version="1.1" class="description\_arrow" >

<path d="M0 0 L8 7 L0 14" stroke="#fff" stroke-width="1" fill="none"/>

</svg>

this.$confirm('确认关闭？')

.then(\_ => {

done();

}).catch(\_ => {});

<el-dialog title="提示" :visible.sync="dialogVisible" width="90%" top="45%" :show-close='false' :before-close="handleClose">

routes: [

{

**path: '/',**

**name: 'Home',**

**component: Home,**

**redirect: '/index',**

children: [

{

path: '/index',

name: 'Index',

component: Index

},

{

path: '/order',

name: 'Order',

component: Order

},

{

path: '/find',

name: 'Find',

component: Find

},

{

path: '/user',

name: 'User',

component: User

}

]

},

{

path: '/detail',

name: 'Detail',

component: Detail

}

Main.js 中

import {Toast,MessageBox} from 'mint-ui'

window.Toast=Toast;

window.MessageBox=MessageBox;

// 身份证 手机号校验

var card=(rule, value, callback) =>{

if (value && (!(/\d{17}[\d|x]|\d{15}/).test(value) || (value.length !== 15 && value.length !== 18))) {

callback(new Error('身份证号码不符合规范'))

} else {

callback()

}

};

var phone=(rule, value, callback) =>{

if (value && (!(/^[1][345678]\d{9}$/).test(value) || !(/^[1-9]\d\*$/).test(value) || value.length !== 11)) {

callback(new Error('手机号码不符合规范'))

} else {

callback()

}

};

**Vue+webpack+axiox 踩过的坑**

1 Vue 打包时 路由文件不能包含 mode: 'history' 否则打包后一片空白

**mode: 'history' 作用：路由地址都是以"#"形式展示，但是有些时候，我们又希望路由地址中不出现"#"** // 去掉路由地址的#

2 页面插入图片 打包后地址有误 方法：build/util.js

if (options.extract) {

return ExtractTextPlugin.extract({

use: loaders,

fallback: 'vue-style-loader',

**publicPath:'../../'**

})

} else {

return ['vue-style-loader'].concat(loaders)

}

3 vue+webpack+axios 不能使用swiper wc-swper vue-swiper 存在兼容性 使用wc-swper vue-carousel

// 动态数据时 要对.VueCarousel-inner 强制添加样式

mounted() {

setTimeout(()=>{

document.getElementsByClassName('VueCarousel-inner')[0].style.transform='translate(0px, 0px)';

},100)

},

/////////////////////////////////////////////////////////////////////////////////////////////////////

**登录时记录路由跳转**

// 登录成功执行跳转

**setTimeout(()=>{**

this.$router.replace(this.$api.getUrlParam('redirect'))

**},1500)**

this.$router.replace({ path: '/login',**query:{redirect:this.$route.fullPath}** })

<router-link :to='{path:"/login",query:{redirect:**this.$route.fullPath**}}' v-else>登录/注册</router-link>

// 路由js

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

// console.log("from.fullPath",from.fullPath)

// console.log("to.fullPath",to.fullPath)

if (to.meta.requireAuth) {

if (getCookie('session')) {

next();

}else{

next({

path:'/login',

query:{redirect: to.fullPath} // 传参数 去哪里

});

}

}

next();

})

// axios 返回状态码 控制

if (res.data.code==='100101') { // token无效

// console.log(router)

// console.log(router.app.\_route.fullPath)

router.push({

path: "/login",

query:{redirect:router.app.\_route.fullPath}

});

return false;

}

///////////////////////////////////////////////////////////////////////////////////////////////////////

<carousel style="height:150px;overflow:hidden;" class="index-carousel" :autoplay="true" :autoplayTimeout="3000" :minSwipeDistance="40" :scrollPerPage="true" :speed="500" :perPage="1" :paginationPadding="10" :paginationSize="10" :loop="true" :paginationColor="'#f4f4f4'" :paginationActiveColor="'#F1B624'">

<slide v-for='item,index in advertList' :key='index'>

<img :src="item.advertImg">

</slide>

</carousel>

const \_list = response.body;

this.list = [...this.list, ...\_list];

**复制功能**

// main.js

import VueClipboard from 'vue-clipboard2';

Vue.use(VueClipboard);

// xxx.vue

<p v-clipboard:copy="'0000000000000000000'" v-clipboard:success="onCopy" v-clipboard:error="onError">{{0000000000000000000}}<i class="el-icon-menu"></i></p>

// xxx.vue methods

onCopy(e){

this.$toast({

message:'复制成功',

duration:'1000'

});

},

onError(e){

this.$toast({

message:'失败',

duration:'1000'

});

},

**// 生成二维码**

// main.js

import QRCode from 'qrcode';

Vue.prototype.$QRCode=QRCode;

//xxx.vue view

<canvas id="canvas"></canvas>

// xxx.vue **mounted ps: 调用一定要放在mouted中**

this.qrcode('000000000000000000000000000000000');

// xxx.vue methods

qrcode (obj) {

var canvas = document.getElementById('canvas')

this.$QRCode.toCanvas(canvas, obj, function (error) {

if (error) console.error(error)

// console.log('success!');

})

},

created () {

// 组件创建完后获取数据，

// 此时 data 已经被 observed 了

this.fetchData()

},

watch: {

// 如果路由有变化，会再次执行该方法

'$route': 'fetchData'

},

scrollBehavior (to, from, savedPosition) {

return { x: 0, y: 0 }

}

对于所有路由导航，简单地让页面滚动到顶部。

返回 savedPosition，在按下 后退/前进 按钮时，就会像浏览器的原生表现那样：

scrollBehavior (to, from, savedPosition) {

if (savedPosition) {

return savedPosition

} else {

return { x: 0, y: 0 }

}

}

如果你要模拟『滚动到锚点』的行为：

scrollBehavior (to, from, savedPosition) {

if (to.hash) {

return {

selector: to.hash

}

}

}

图片上传到七牛

/// template

<el-upload

class="avatar-uploader"

:action="uploadQiniuUrl"

:show-file-list="false"

:http-request="upLoadFun"

:before-upload="beforeAvatarUpload">

<img v-if="ruleForm.imtokenQrcode" :src="qiniuDomain+ruleForm.imtokenQrcode" class="avatar">

<i v-else class="el-icon-plus avatar-uploader-icon"></i>

</el-upload>

// js

var axios = require('axios')

// methods

beforeAvatarUpload(file) {

const isJPG = file.type === 'image/jpeg'||

file.type === 'image/png'||

file.type === 'image/JPEG'||

file.type === 'image/PNG'||

file.type === 'image/jpg'||

file.type === 'image/JPG';

const isLt2M = file.size / 1024 / 1024 < 3;

if (!isJPG) {

// this.$message.error('请上传jpg,jpeg,png格式!');

Toast({

mes:'请上传jpg,jpeg,png格式!',

timeout:'1000'

});

return false

}

if (!isLt2M) {

// this.$message.error('图片大小不能超过 2M');

Toast({

mes:'图片大小不能超过 3M',

timeout:'1000'

});

return false

}

return isJPG && isLt2M;

},

upLoadFun(response){

var file=response.file;

lrz( file, [] ).then((rst)=> {

//成功时执行

let formData = new FormData();//创建form对象

formData.append("file", rst.file);

formData.append('token',this.qiniuToken);

//console.log("formData.get('file')",formData.get('file')); //FormData私有类对象，访问不到，可以通过get判断值是否传进去

let config = {

headers:{'Content-Type':'multipart/form-data'}

};

Loading.open(' ');

axios.post(this.uploadQiniuUrl,formData,config)

.then(response=>{

Loading.close();

Toast({

mes:'上传成功',

timeout: 500

});

this.ruleForm.imtokenQrcode=response.data.key;

})

.catch(function (err) {

Loading.close();

let res = err.response

if (err) {

console.error('api error, HTTP CODE: ' + res)

}

})

}).catch((error)=> {

console.log(error)

//失败时执行

}).always(()=> {

//不管成功或失败，都会执行

})

},

bigImg(event){

var curObj=event.currentTarget

if(curObj.className=='send'){

curObj.className="send scaleImg"

}else{

curObj.className="send"

}

}

**Element tree**

<el-tree

:props="props"

:highlight-current="true"

lazy

:load="loadNode">

</el-tree>

data() {

return {

props: {

label: 'label',

children: 'children'

},

count: 1,

labels:'',

}

},

loadNode(node, resolve) {

console.log('node',node);

this.labels=node.label;

console.log('this.labels===========',this.labels);

if (node.level === 0) {

return resolve([{ label: this.userInfo.bindPhone }]);

}

console.log('this.labels',this.labels);

this.$api.post('/group/checkNextTreeInfo',{'label':this.labels },result => {

// this.labels=result.data;

var arrObj=[{label:'15588889999'},{label:'16622223333'}]

setTimeout(() => {

resolve(arrObj);

}, 500);

},err=>{

Toast({

mes:err.msg,

timeout: 1500

});

});

},

**// 图片压缩**

**import lrz from 'lrz';**

**vue 模拟app 页面切换**

**// router**

Router.prototype.goBack = function () {

this.isBack = true

// console.log('this.isBack',this.isBack)

window.history.go(-1)

}

**// App.vue**

<template>

<div style="height:100vh">

<transition :name="direction" >

<router-view class="appView"></router-view>

</transition>

</div>

</template>

<script>

export default {

name: 'App',

data() {

return {

direction: "slide-right"

}

},

watch: {

**'$route' (to, from) {**

**// console.log('this.$router.isBack,this.$router.isBack',this.$router.isBack)**

**let isBack = this.$router.isBack // 监听路由变化时的状态为前进还是后退**

**if(isBack) {**

**this.direction = 'slide-right'**

**} else {**

**console.log(to.meta.index , from.meta.index)**

**if(to.meta.index > from.meta.index){**

**//设置动画名称**

**this.direction = 'slide-left';**

**}else{**

**if (!to.meta.index) {**

**this.direction = 'slide-left';**

**}else{**

**this.direction = 'slide-right';**

**}**

**}**

**}**

**this.$router.isBack = false**

}

},

}

</script>

<style>

.appView {

position: absolute;

width:100%;

transition: transform 0.3s ease-out;

}

.slide-left-enter{

transform: translate(100%, 0);

}

.slide-left-leave-active{

transform: translate(-50%, 0);

}

.slide-right-enter {

transform: translate(-50%, 0);

}

.slide-right-leave-active{

transform: translate(100%, 0);

}

</style>

state.addressList.**unshift**(obj) // unshift 数组前添加

state.addressList.**splice**(ii,1,obj);// 删除元素 新增obj 元素

state.addressList.**splice**(ii,1);// 删除元素

// 使用 vuex 来管理 获取的对象需从新创建存储空间

var newObj=**Object.assign**({},this.ruleForm);

this.ruleForm=**Object.assign**({},item)

**meta: {**

**title: '商品详情'**

**}**

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

/\* 路由发生变化修改页面title \*/

if (to.meta.title) {

**document.title = to.meta.title**

}

next()

})

**router.afterEach**((to, from) => {

window.document.title = constants[to.name];

})

<el-input v-model=*"ruleForm.exchangeAmount"* placeholder=*"兑换数量"* onkeyup="this.value=this.value.replace(/[^\d]/g,'');"></el-input>

if ( !(/^(\d+[a-zA-Z]+|[a-zA-Z]+\d+)([0-9a-zA-Z]\*)$/).test(value) ) {

callback(new Error('必须字母和数字组合'));

}

{ pattern: /^\d+\.?\d{0,2}$/, message: '请输入数字' }

<el-dialog :append-to-body=*"true"*

**webpack 打包编译-webkit-box-orient: vertical 后消失**

解决方案

optimize-css-assets-webpack-plugin这个插件的问题

注释掉webpack.prod.conf.js中下面的代码

new OptimizeCSSPlugin({

cssProcessorOptions: config.build.productionSourceMap

? { safe: true, map: { inline: false } }

: { safe: true }

}),

filterFun(){

var val=this.input.trim()

var searchRegex = new RegExp(val, 'i');

if(val){

this.likeProductList=this.likeProductListBak.filter(item=>{

if (searchRegex.test(item.productName)||searchRegex.test(item.merchantName)) {

return true

}

});

}else{

this.likeProductList=this.likeProductListBak

}

}

backwx() {

WeixinJSBridge.call('closeWindow');

}

var browser = navigator.userAgent.toLowerCase();

if(browser.match(/Alipay/i)=="alipay"){

console.log("支付宝app的浏览器");

}else if(browser.match(/MicroMessenger/i)=="micromessenger"){

console.log("微信app的浏览器");

}else{

console.log("其它浏览器");

}

二.点击事件响应缓慢

(1).安装fastclick (npm install fastclick -S)

(2).在main.js中设置方法

import FastClick from 'fastclick'

FastClick.attach(document.body);

FastClick.prototype.focus = function(targetElement) {

var length;

if (targetElement.setSelectionRange && targetElement.type.indexOf('date') !== 0 && targetElement.type !== 'time' && targetElement.type !== 'month') {

length = targetElement.value.length;

targetElement.focus();

targetElement.setSelectionRange(length, length);

} else {

targetElement.focus();

}

};

this.agentList=[...this.agentList,...result.data.list]

mounted() {

document.getElementById('bodyScroll').addEventListener('scroll', this.handleScroll)

},

beforeDestroy () {

document.getElementById('ScrollBox').removeEventListener('scroll', this.handleScroll)

},

handleScroll(e){

var clientHeight=e.target.clientHeight;

var scrollTop=e.target.scrollTop;

var scrollHeight=e.target.scrollHeight;

// console.log(e)

if ( scrollTop>=(scrollHeight-clientHeight-30)&&this.nextPage) {

this.nextPage=false;

++this.pageNo;

this.getData()

}

},

**vue.js 安卓调js 返回数据**

created(){

**window.appCallBack=this.appCallBack;//将方法暴露个window，Android原生才能访问**

},

// vue事件

wxLogin(){

var u = navigator.userAgent, app = navigator.appVersion;

var isAndroid = u.indexOf('Android') > -1 || u.indexOf('Linux') > -1; //g

var isIOS = !!u.match(/\(i[^;]+;( U;)? CPU.+Mac OS X/); //ios终端

if (isAndroid) {

window.android.toWxLogin('123456');

}else if (isIOS) {

　　　　//这个是ios操作系统

try{

window.webkit.messageHandlers.toWxLogin.postMessage("aaa")

}catch(error){

console.error('The native context not exist ')

}

}else {

this.$toast("你使用移动设备扫一扫");

}

},

appCallBack(code){

alert(JSON.stringify(code))

},

if (this && !this.\_isDestroyed) {

setTimeout(()=>{

this.getTimeInfo();

},1000);

}

var zhengshu=(rule, value, callback) =>{

if (value === '') {

callback(new Error('请输入数量'));

}else if (value && !(/^[0-9]\*$/).test(value) ) {

this.ruleForm.amount=''

callback(new Error('请输入正整数'))

} else if(value>(this.topInfo.totalAmount-this.topInfo.freezeAmount)){

callback(new Error('不能超过'+parseInt((this.topInfo.totalAmount-this.topInfo.freezeAmount))))

}else{

callback()

}

};