Vue小总结

1.路由跳转的几种方式  
 标签——<router-link :to="{name: 'Register'}" >

<router-link :to="{path:’login’}" >

<a v-link="{path: '/login'}">

<a v-link="{name: 'user', params: {userId: 1}">

方法——this.$router.push({name: 'Register'})

// 字符串

router.push('home')

// 对象

router.push({ path: 'home' })

// 命名的路由

router.push({ name: 'user', params: { userId: 123 }})

// 带查询参数，变成 /register?plan=private

router.push({ path: 'register', query: { plan: 'private' }})

2.路由数据传递方式

1）路由直接传递  
 this.$router.push({

name: 'generalAdminOrderFlowAdd',

params: {

type: 'add',

templateType: this.orderTemplateType

}

})

let type = this.$route.params.type

2）使用vuex  
state.js mutations.js mutation-types.js index.js actions.js getters.js

3. try{}catch{}finally{}的用法  
  
 在JavaScript中，我们使用try…catch…finally语句来执行例外处理，即通过它来捕捉错误发生后导致的例外或者执行throw语句产生的例外。它的基本语法如下：

　try {

　　// 此处是可能产生例外的语句

　　} catch(error) {

　　// 此处是负责例外处理的语句

　　} finally {

　　// 此处是出口语句

}

4. async 与 await

async 表示函数里有异步操作，await 表示紧跟在后面的表达式需要等待结果。

5. router.beforeEach() 每次路由切换调用的函数

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

if (to.name !== "Login" && to.name !== "Register") {

let token = window.sessionStorage.getItem('token')

if (!token) next('/login')

else next()

} else {

next()

}

})

6. 一些安装指令

安装淘宝镜像

npm install -g cnpm --registry=https://registry.npm.taobao.org

安装webpack

cnpm install webpack –g

安装vue脚手架

npm install vue-cli –g

7．Router中的{ path: '\*', redirect: '/home' }设置通用路径，即其他不可识别的通用路径

8. Vue.http.options.emulateJSON = true

——emulateJSON(布尔值)

默认值为：false,当值为true并且data为对象时，设置请求头Content-Type的值为application/x-www-form-urlencoded

如果服务器无法处理编码为application/json的请求，可以启用emulateJSON选项。启用之后，请求会以application/x-www-form-urlencoded为MIME type，就像普通的HTML表单一样。

9. Vue.http.options.crossOrigin = true

——crossOrigin（布尔值）

默认值为：null，表示是否跨域，如果没有设置该属性，vue-resource内部会判断浏览器当前URL和请求URL是否跨域。

10. Vue.http.options.emulateHTTP = true;

——emulateHTTP（布尔值）

默认值为:false,当值为true是，用HTTP的POST方式PUT，PATCH，DELETE等请求，并设置请求头字段HTTP\_Method\_Override为原始请求方法。

如果Web服务器无法处理PUT, PATCH和DELETE这种REST风格的请求，你可以启用enulateHTTP现象。启用该选项后，请求会以普通的POST方法发出，并且HTTP头信息的X-HTTP-Method-Override属性会设置为实际的HTTP方法。

11. new Vue({

router,

store,

render: h => h(App)

//es6 render: (function(h){return: h(app)})

}).$mount('#app')  
.$mount('#app')——当Vue实例没有el属性时，则该实例尚没有挂载到某个dom中；假如需要延迟挂载，可以在之后手动调用vm.$mount()方法来挂载。

12. css3画叉  
.a{

display: inline-block;

width: 20px;

height:5px;

background: #ccc;

line-height:0;

font-size:0;

vertical-align:middle;

-webkit-transform: rotate(45deg);

position: absolute;

top: 16px;

right: 13px;

}

.a:after{

content:'/';

display:block;

width: 20px;

height:5px;

background: #ccc;

-webkit-transform: rotate(-90deg);

}

13. webpack 官方文档的介绍[require.context](https://webpack.js.org/guides/dependency-management/" \l "require-context)

简单说就是：有了 require.context，我们可以通过正则匹配引入相应的文件模块。

require.context(directory, useSubdirectories, regExp)

require.context 有三个参数：

* directory：说明需要检索的目录
* useSubdirectories：是否检索子目录
* regExp: 匹配文件的正则表达式

const apis = requireAll(require.context('./', true, /\.js$/))

14. Object.assign()

Object.assign() 方法用于将所有可枚举的属性的值从一个或多个源对象复制到目标对象。它将返回目标对象。

语法——Object.assign(target, ...sources)

如果目标对象中的属性具有相同的键，则属性将被源中的属性覆盖。后来的源的属性将类似地覆盖早先的属性。注意，Object.assign 会跳过那些值为 null 或 undefined 的源对象。

针对深度拷贝，需要使用其他方法，因为 Object.assign() 拷贝的是属性值。假如源对象的属性值是一个指向对象的引用，它也只拷贝那个引用值。

var o1 = { a: 1 };

var o2 = { b: 2 };

var o3 = { c: 3 };

var obj = Object.assign(o1, o2, o3);

console.log(obj); // { a: 1, b: 2, c: 3 }

console.log(o1); // { a: 1, b: 2, c: 3 }, 注意目标对象自身也会改变。

15. reduce 方法 (Array)

对数组中的所有元素调用指定的回调函数。该回调函数的返回值为累积结果，并且此返回值在下一次调用该回调函数时作为参数提供。

直白的说就是将一个多维的数组合并为少一维的新数组。

16. import \* as apis from '../apis'

使用这样的语法可以直接使用 ‘apis.方法名’引用方法

17. import { mapActions } from 'vuex'

18． data() {return {}}, ——有括号

Created(){} ——有括号

Mounted(){} ——有括号

Computed:{}

Methods:{}

Watch:{}

Components:{}

19. HTMl5的sessionStorage和localStorage

html5中的Web Storage包括了两种存储方式：sessionStorage和localStorage。

sessionStorage用于本地存储一个会话（session）中的数据，这些数据只有在同一个会话中的页面才能访问并且当会话结束后数据也随之销毁。因此sessionStorage不是一种持久化的本地存储，仅仅是会话级别的存储。

而localStorage用于持久化的本地存储，除非主动删除数据，否则数据是永远不会过期的。

window.localStorage.setItem window.localStorage.getItem

20. 3d按钮的实现  
 两个类：一个初始样式，一个：active样式。

主要属性：阴影，：active位置偏移。如top:5px;

21. 请求写法，结合vue-resource

Api文件夹写法：  
 export const keywordSearch = keyword => {

return Vue.http.get(`${host}/app/keywordSearch/${keyword}`)

}

带参数写法：

export const repay = (id) => {

return Vue.http.get(`${host}/app/repay`, { params: { id } })

}

export const pay = (parkrecordId) => {

return Vue.http.post(`${host}/app/pay`, { parkrecordId })

}

export const payConfirm = (payrecordId) => {

return Vue.http.put(`${host}/app/pay/confirm`, { payrecordId })

}

原页写法：

this.$http.get('/someUrl', [options]).then(function(response){

// 响应成功回调

}, function(response){

// 响应错误回调

});

vue-resource的请求API是按照REST风格设计的，它提供了7种请求API：

get(url, [options])

head(url, [options]) ——只请求页面的首部。

delete(url, [options]) ——请求服务器删除指定的页面

jsonp(url, [options])

post(url, [body], [options])

put(url, [body], [options]) ——PUT用于替换资源

patch(url, [body], [options]) ——PATCH用于更新部分资源