目录

[二、 基本配置](#_Toc12199_WPSOffice_Level1) [3](#_Toc12199_WPSOffice_Level1)

[1. 基础框架搭建](#_Toc22574_WPSOffice_Level2) [3](#_Toc22574_WPSOffice_Level2)

[1） 安装node](#_Toc22574_WPSOffice_Level3) [3](#_Toc22574_WPSOffice_Level3)

[2） 安装vue cli3](#_Toc20669_WPSOffice_Level3) [3](#_Toc20669_WPSOffice_Level3)

[3） 用cli工具创建项目](#_Toc27910_WPSOffice_Level3) [3](#_Toc27910_WPSOffice_Level3)

[2. 箭头函数Arrow Function](#_Toc20669_WPSOffice_Level2) [3](#_Toc20669_WPSOffice_Level2)

[3. 其它细节](#_Toc27910_WPSOffice_Level2) [4](#_Toc27910_WPSOffice_Level2)

[1） EsLint标准](#_Toc60_WPSOffice_Level3) [4](#_Toc60_WPSOffice_Level3)

[2） 用parseFloat判断数值绝对相等](#_Toc24733_WPSOffice_Level3) [4](#_Toc24733_WPSOffice_Level3)

[3）方法参数(systemType= this.kk) //todo](#_Toc15070_WPSOffice_Level3) [4](#_Toc15070_WPSOffice_Level3)

[4）const {rows, total} = res //todo](#_Toc7580_WPSOffice_Level3) [4](#_Toc7580_WPSOffice_Level3)

[4） 登出](#_Toc13778_WPSOffice_Level3) [4](#_Toc13778_WPSOffice_Level3)

[5） 获取用户信息的方法](#_Toc11539_WPSOffice_Level3) [4](#_Toc11539_WPSOffice_Level3)

[6） 全局注册变量](#_Toc17428_WPSOffice_Level3) [4](#_Toc17428_WPSOffice_Level3)

[三、 其它插件安装及其使用](#_Toc22574_WPSOffice_Level1) [5](#_Toc22574_WPSOffice_Level1)

[1.axios](#_Toc60_WPSOffice_Level2) [5](#_Toc60_WPSOffice_Level2)

[1）安装](#_Toc4984_WPSOffice_Level3) [5](#_Toc4984_WPSOffice_Level3)

[2）文件结构](#_Toc2054_WPSOffice_Level3) [5](#_Toc2054_WPSOffice_Level3)

[3）使用](#_Toc25180_WPSOffice_Level3) [7](#_Toc25180_WPSOffice_Level3)

[2.vue-cookies](#_Toc24733_WPSOffice_Level2) [8](#_Toc24733_WPSOffice_Level2)

[1）安装](#_Toc19143_WPSOffice_Level3) [8](#_Toc19143_WPSOffice_Level3)

[2）使用](#_Toc3311_WPSOffice_Level3) [9](#_Toc3311_WPSOffice_Level3)

[4） Api](#_Toc28458_WPSOffice_Level3) [9](#_Toc28458_WPSOffice_Level3)

[四、 Ant Desgin Vue的后台框架](#_Toc20669_WPSOffice_Level1) [9](#_Toc20669_WPSOffice_Level1)

[1. 安装](#_Toc15070_WPSOffice_Level2) [9](#_Toc15070_WPSOffice_Level2)

[2. 基本标签](#_Toc7580_WPSOffice_Level2) [10](#_Toc7580_WPSOffice_Level2)

[1） Button](#_Toc15762_WPSOffice_Level3) [10](#_Toc15762_WPSOffice_Level3)

[2） Icon](#_Toc25981_WPSOffice_Level3) [10](#_Toc25981_WPSOffice_Level3)

[3） Select](#_Toc7428_WPSOffice_Level3) [11](#_Toc7428_WPSOffice_Level3)

[4） Layer](#_Toc28294_WPSOffice_Level3) [11](#_Toc28294_WPSOffice_Level3)

[5） Menu](#_Toc16366_WPSOffice_Level3) [13](#_Toc16366_WPSOffice_Level3)

[五、 vxe-table表格工具](#_Toc27910_WPSOffice_Level1) [14](#_Toc27910_WPSOffice_Level1)

[1. 安装](#_Toc13778_WPSOffice_Level2) [14](#_Toc13778_WPSOffice_Level2)

[2. vxe-table常规表格 //todo](#_Toc11539_WPSOffice_Level2) [15](#_Toc11539_WPSOffice_Level2)

[3. vxe-grid配置式表格参数详解](#_Toc17428_WPSOffice_Level2) [15](#_Toc17428_WPSOffice_Level2)

[4. vxe-pager配置式表格参数详解](#_Toc4984_WPSOffice_Level2) [16](#_Toc4984_WPSOffice_Level2)

# 基本配置

## 基础框架搭建

1. 安装node
2. 安装vue cli3

npm install -g @vue/clinpm install -g @vue/cli

1. 用cli工具创建项目

vue create [项目名]

## 箭头函数Arrow Function

数组常用方法fliter,map,forEach,some

Fliter //todo

创建一个新的数组，不会改变原始数组的值。

forEach、map

forEach是没有返回值。

map有返回值，可以return 出来。

其它方面基本都一样

遍历一个数组，对象型数组同样适用

//data中定义

data() {  
 return {  
 arr: ['a1', 'a2', 'a3', 'a4'],  
 }  
},

//方法中遍历 只有一个变量

this.arr.forEach(item => {  
 alert(item)  
})

当有3个变量时，第一个为当前item，第二个为序号，第三个为原变量

变量.map((item ,index,arr) => {  
 console.log(item)  
 console.log(index)  
 console.log(arr)  
}

## 其它细节

1. EsLint标准

EsLint采用的是babel-eslint

1. 用parseFloat判断数值绝对相等

parseFloat(res.code) === 0

1. 登出

this.$Utils.removeCookieFn()

this.$router.replace({path: '/Login'})

1. 获取用户信息的方法

userUserInfoAPI

1. 全局注册变量

import 类库名 from '类库名'  
Vue.prototype.$类库名 = 类库名

后边在任意一个模块中，可以this.$类库名引用这个类库

1. utils中重写了trim方法

trim (s) { // 去左右空格  
 if (!s) {  
 return null  
 } else {  
 return s.replace(/(^\s\*)|(\s\*$)/g, '')  
 }  
}

1. click.stop

@click.stop 阻止事件冒泡

@click.prevent 阻止事件的默认行为，

this.systemTypes.forEach(item => {  
 if (item === 'pt') return  
 this.platformAgencyUnAuthListAPIFn(item)  
})

# 其它插件安装及其使用

## Axios

### 1）安装

npm install --save axios

//可以理解为qs 类似于JSON.stringify转换格式的一种方法

npm install qs

//cookies 后边的pageAjax中用得到

### 2）文件结构

在service 目录下新建文件axios.js

这个文件相当于把axios再次封装，里边的axios.defaults.baseURL 是用来配置域名的

import axios from 'axios'  
import Vue from 'vue'  
  
// 全局默认配置  
// 设置 POST 请求头  
axios.defaults.headers.post['Content-Type'] = 'application/x-www-form-urlencoded'  
axios.defaults.timeout = 10000  
axios.defaults.baseURL = '/' // api 即vue.config.js 中配置的地址  
  
// 配置 CORS 跨域  
axios.defaults.withCredentials = true  
axios.defaults.crossDomain = true  
  
// 请求发起前拦截器  
axios.interceptors.request.use(config => {  
 // 全局body设置token  
 /\* if (Utils.getJwt().length > 0) {  
 if (typeof config.data === 'string') {  
 config.data += `&jwt=${Utils.getJwt()}`  
 } else {  
 if (config.data) {  
 config.data.jwt = Utils.getJwt()  
 }  
 }  
 } \*/  
  
 // 全局的header设置token  
 // config.headers['token'] = 'token'  
  
 // 这句不能省，不然后面的请求就无法成功发起，因为读不到配置参数  
 return config  
}, error => {  
 // 异常处理  
 return Promise.reject(error)  
})  
  
// 响应拦截  
axios.interceptors.response.use(response => {  
 // 全局登录过滤，如果没有登录，直接跳转到登录 URL  
 if (response.data.code === '401') {  
 // 未登录  
 // Vue.prototype.$Store.commit('changeStateInfo', {info: {}, isLogin: Math.random()})  
 Vue.$router.replace('/Login')  
 return false  
 }  
 // 这里返回的 response.data 是被 axios 包装过的一成，所以在这里抽取出来  
 return response.data  
}, error => {  
 if (error.response) {  
 const statusCode = parseFloat(error.response.status)  
 const statusMsg = error.response.data.message  
 switch (statusCode) {  
 case 401:  
 Vue.prototype.$Store.commit('changeStateInfo', {info: {}, isLogin: Math.random()})  
 break  
 case 403:  
 Vue.prototype.$Store.commit('changeStateInfo', {info: {}, isLogin: Math.random()})  
 break  
 case 404:  
 case 504:  
 Vue.prototype.$message.error(`【${statusCode}】 - 服务器异常情况，请稍后刷新页面再试`)  
 break  
 default:  
 Vue.prototype.$message.error(`【${statusCode}】 - ${statusMsg}`)  
 break  
 }  
 } else {  
 Vue.prototype.$message.error(`【${error}】 - 网络超时`)  
 }  
  
 return Promise.reject(error)  
})  
  
// 导出  
export default axios

在service 目录下新建文件pageAjax.js

import axios from './axios'  
import qs from 'qs' // 注意: post, put提交方式需要采用该写法: qs.stringify(params)  
  
//全网址  
export const testA = () => axios.get('/v1/platform/org/info/license/2/1')  
  
//网址中带参数  
export const testB = (systemType, id) => axios.get(`/v1/platform/org/image/list/${systemType}/${id}`)  
  
//网址+？后的参数  
export const testC = (systemType, params) => axios.get(`/v1/platform/org/unAuth/list/${systemType}`, params)  
  
// qs.stringify()将对象 序列化成URL的形式，以&进行拼接。  
export const testD = (systemType, params) => axios.get(`/v1/platform/org/unAuth/list/${systemType}`, qs.stringify(params))

### 使用

引用

import {  
 testA,  
 testB,  
 testC,  
 testD  
} from "@/service/pageAjax";

定义于methods中

methods: {  
 async provinceAreaAPIFn () {  
 await testA().then(res => {  
 console.log(res)  
 alert(res)  
 })  
 },  
 async provinceAreaAPIFnB () {  
 await testB(2, 1).then(res => {  
 console.log(res)  
 })  
 },  
 async provinceAreaAPIFnC(){  
 let params={pageNum:1,pageSize:10}  
 await testC(  
 2,  
 params).then(res => {  
 console.log(res)  
 })  
 },  
 async provinceAreaAPIFnD(){  
 let params={pageNum:1,pageSize:10}  
 await testD(  
 2,  
 params).then(res => {  
 console.log(res)  
 })  
 }  
},

调用

this.provinceAreaAPIFnD ()

### 使用put、delete

列表方法一般用axios.get

新增方法一般用axios.post

编辑方法一般用axios.put

删除方法一般用axios.delete

1、GET

get请求是用来获取数据的，只是用来查询数据，不对服务器的数据做任何的修改，新增，删除等操作。

在这里我们认为get请求是安全的，以及幂等的。安全就是指不影响服务器的数据，幂等是指同一个请求发送多次返回的结果应该相同。

特点：

get请求会把请求的参数附加在URL后面，这样会产生安全问题，如果是系统的登陆接口采用的get请求，需要对请求的参数做一个加密。

get请求其实本身HTTP协议并没有限制它的URL大小，但是不同的浏览器对其有不同的大小长度限制

2、POST

post请求一般是对服务器的数据做改变，常用来数据的提交，新增操作。

特点：

post请求的请求参数都是请求体中

post请求本身HTTP协议也是没有限制大小的，限制它的是服务器的处理能力

3、PUT

put请求与post一样都会改变服务器的数据，但是put的侧重点在于对于数据的修改操作，但是post侧重于对于数据的增加。

4、DELETE

delete请求用来删除服务器的资源。

5、OPTIONS

options请求属于浏览器的预检请求，查看服务器是否接受请求，预检通过后，浏览器才会去发get，post，put，delete等请求。至于什么情况下浏览器会发预检请求，浏览器会会将请求分为两类，简单请求与非简单请求，非简单请求会产生预检options请求。

## vue-cookies

### 1）安装

npm install vue-cookies --save

### 2）使用

import Vue from 'Vue'

import VueCookies from 'vue-cookies'

Vue.use(VueCookies)

### 3）Api

设置 cookie：

this.$cookies.set(keyName, time) //return this

获取cookie

this.$cookies.get(keyName) // return value

删除 cookie

this.$cookies.remove(keyName) // return false or true , warning： next version return this； use isKey(keyname) return true/false,please

查看一个cookie是否存在（通过keyName）

this.$cookies.isKey(keyName) // return false or true

获取所有cookie名称

this.$cookies.keys() // return a array

## Moment //todo

## Print //todo

# Ant Desgin Vue的后台框架

地址：<https://www.antdv.com>

## 安装

项目中初始化antd的组件

npm i --save ant-design-vue

导入antd

在根目录src/main.js中

import Antd from 'ant-design-vue';  
import 'ant-design-vue/dist/antd.css';

//下文中引用

Vue.use(Antd);

各个插件分别导入的方法

import {  
 Button,  
 Icon,  
 } from 'ant-design-vue'

//下文中引用

Vue.use(Button)  
Vue.use(Icon)

## 基本标签

### Button

<a-button @click="bb" size="large" :loading="isnot">Hello world!</a-button>

标签为<a-button></a-button>

点击事件为 @click="方法名"

按钮大小 size="large" 默认大小 large大 small小

按钮类型（样子） type="danger"



加载中 :loading="isnot" 参数只能传true false ，字符串不行

图标icon="plus" 具体图片库参照https://www.antdv.com/components/icon/

### Icon

<a-icon type="apple" :style="{ fontSize: '16px', color: '#ff4f02' }" />

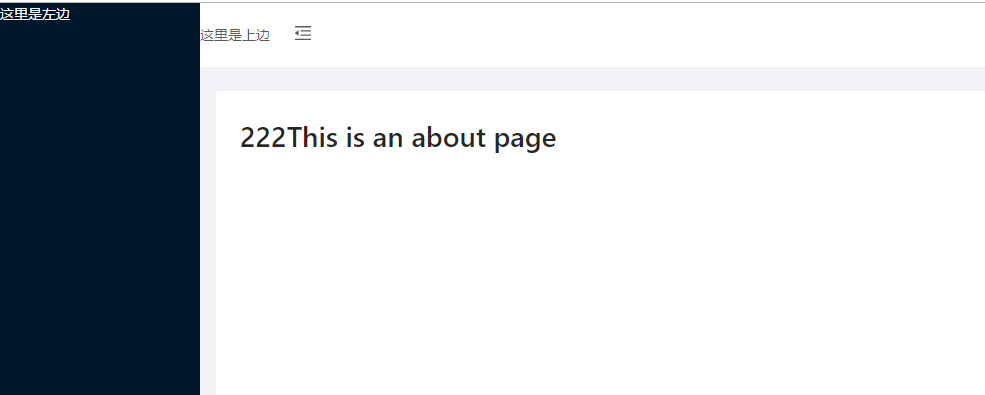
### Select

defaultValue为默认值

@change 为改变选项指事件

<a-select defaultValue="2" @change="fuc">  
 <a-select-option value="0">lucy</a-select-option>  
 <a-select-option value="1">lucy1</a-select-option>  
 <a-select-option value="2">lucy2</a-select-option>  
 <a-select-option value="3">lucy3</a-select-option>  
</a-select>

### Layer



<template>  
 <a-layout id="components-layout-demo-custom-trigger">  
 <a-layout-sider :trigger="null" collapsible v-model="collapsed">  
 <div style="color: #fff">这里是左边</div>  
 </a-layout-sider>  
 <a-layout>  
 <a-layout-header style="background: #fff; padding: 0">  
 <a-icon  
 class="trigger"  
 :type="collapsed ? 'menu-unfold' : 'menu-fold'"  
 @click="()=> collapsed = !collapsed"  
 />  
 <div style="float: left">这里是上边</div>  
 </a-layout-header>  
 <a-layout-content  
 :style="{ margin: '24px 16px', padding: '24px', background: '#fff', minHeight: '280px' }"  
 >  
 <router-view class="ss"/>  
 </a-layout-content>  
 </a-layout>  
 </a-layout>  
</template>  
  
<style lang="scss">  
 .ant-layout-has-sider {  
 position: absolute;  
 bottom: 0;  
 top: 0;  
 left: 0;  
 right: 0px;  
 }  
 #components-layout-demo-custom-trigger .trigger {  
 font-size: 18px;  
 line-height: 64px;  
 padding: 0 24px;  
 cursor: pointer;  
 transition: color 0.3s;  
 }  
 #components-layout-demo-custom-trigger .trigger:hover {  
 color: #1890ff;  
 }  
 #components-layout-demo-custom-trigger .logo {  
 height: 32px;  
 background: rgba(255, 255, 255, 0.2);  
 margin: 16px;  
 }  
 #app {  
 font-family: 'Avenir', Helvetica, Arial, sans-serif;  
 -webkit-font-smoothing: antialiased;  
 -moz-osx-font-smoothing: grayscale;  
 text-align: center;  
 color: #2c3e50;  
 }  
 #nav {  
 padding: 30px;  
 a {  
 font-weight: bold;  
 color: #2c3e50;  
 &.router-link-exact-active {  
 color: #42b983;  
 }  
 }  
 }  
</style>  
<script>  
 export default {  
 data() {  
 return {  
 collapsed: false,  
 };  
 },  
 };  
</script>

### Menu

默认哪一项选中 :defaultSelectedKeys="['key值']" 备注：这里的key不能为有下拉的key，且这里只是选中哪个一项，不是点击这项目后执行的跳转页面

默认哪组下拉是打开状态:defaultOpenKeys="['key值','key值']"

初始化链接去哪个位置需要在方法中实现

this.$router.push(链接地址)

:defaultSelectedKeys="['key值']"

<a-menu  
 :defaultOpenKeys="['sub1','sub2','sub3']"  
 :defaultSelectedKeys="['1']"  
 mode="inline"  
 theme="light"  
 :inlineCollapsed="collapsed"  
 @openChange="openChange"  
 @select="goto">  
 <a-menu-item key="1">  
 <a-icon type="pie-chart"/>  
 <span>选项一</span>  
 </a-menu-item>  
 <a-menu-item key="2">  
 <a-icon type="desktop"/>  
 <span>选项二</span>  
 </a-menu-item>  
 <a-sub-menu key="sub1">  
 <span slot="title"><a-icon type="mail"/><span>选项三</span></span>  
 <a-menu-item key="3">选项三-子选项一</a-menu-item>  
 <a-menu-item key="4">选项三-子选项二</a-menu-item>  
 </a-sub-menu>  
 <a-sub-menu key="sub2">  
 <span slot="title"><a-icon type="appstore"/><span>选项四</span></span>  
 <a-menu-item key="9">选项四-子选项一</a-menu-item>  
 <a-menu-item key="10">选项四-子选项二</a-menu-item>  
 <a-sub-menu key="sub3" title="选项四-子选项三">  
 <a-menu-item key="11">选项四-子选项三-子选项一</a-menu-item>  
 <a-menu-item key="12">选项四-子选项三-子选项二</a-menu-item>  
 </a-sub-menu>  
 </a-sub-menu>  
</a-menu>

# vxe-table表格工具

地址：<https://xuliangzhan_admin.gitee.io/vxe-table>

渲染性能对比：[vxe-grid](https://xuliangzhan_admin.gitee.io/vxe-table/" \l "/grid/api)（性能最优，不需要为每一列创建实例） > [vxe-table](https://xuliangzhan_admin.gitee.io/vxe-table/" \l "/table/api)（性能略差，需要为每一列创建实例）

## 安装

项目中初始化vxe-table的组件

npm install xe-utils vxe-table

导入vxe

在根目录src/main.js中

import Vue from 'vue'

import 'xe-utils'

import VXETable from 'vxe-table'

import 'vxe-table/lib/index.css' Vue.use(VXETable)

各个插件分别导入的方法

import {  
 Button,  
 Icon,  
 } from 'vxe-table'

import zhCNLocat from 'vxe-table/lib/locale/lang/zh-CN'

//下文中引用

Vue.use(Button)  
Vue.use(Icon)

highlight-hover-row

## vxe-table常规表格 //todo

表格内容整体居左/中/右

在vxe-table标签中 align="left" left/center/right

鼠标移上去高亮

在vxe-table标签中 highlight-hover-row

## vxe-grid配置式表格参数详解

API地址：[https://xuliangzhan\_admin.gitee.io/vxe-table/#/grid/api](https://xuliangzhan_admin.gitee.io/vxe-table/" \l "/grid/api)

点击选中该行

highlight-current-row

竖向是否分开

Border

内容过长时省略号

show-overflow

表头内容过长时省略号

show-header-overflow

所有的列是否允许拖动列宽调整大小

resizable

表格是否显示加载中

loading

表格的尺寸 string 三个值 medium, small, mini

Size

每一列的详细属性

columns

双击事件

@cell-dblclick

单击事件

@cell-click

## vxe-pager配置式表格参数详解

API地址：[https://xuliangzhan\_admin.gitee.io/vxe-table/#/pager/api](https://xuliangzhan_admin.gitee.io/vxe-table/" \l "/pager/api)

<vxe-pager  
 :current-page="当前页"  
 :page-size="每页有多少条"  
 :total="总共有多少条"  
 :layouts="['PrevPage', 'JumpNumber', 'NextPage', 'FullJump', 'Sizes', 'Total']"  
 :page-sizes="[10,20,30,40]"  
 @page-change="点击事件">  
</vxe-pager>

## export-xlsx

安装

npm install xe-utils vxe-table vxe-table-plugin-export-xlsx xlsx

导入

import Vue from 'vue'

import VXETable from 'vxe-table'

import VXETablePluginExportXLSX from 'vxe-table-plugin-export-xlsx'

Vue.use(VXETable)

VXETable.use(VXETablePluginExportXLSX)

使用

exportEvent() {  
 this.$refs.[表名].exportData({  
 filename: 'export',  
 sheetName: 'Sheet1',  
 type: 'xlsx'  
 })  
},