1. 前端终极知识体系大法好：<https://github.com/JacksonTian/fks#javascript>
2. node.js安装详：

<https://www.cnblogs.com/zhouyu2017/p/6485265.html>

https://blog.csdn.net/qq\_27626333/article/details/77857223

1. 使用nrm管理镜像：

npm install nrm -g

nrm ls //列出可选的镜像源

nrm add ctsi <http://repo.ssdc.solutions/repository/npmall/> //添加新的镜像源

nrm use ctsi //设置相应的镜像源

1. npm查看配置信息：npm config list
2. iframe显示第三方网页，不想显示部分头部信息，解决方案一：

使用css的margin-top属性为负值来隐藏部分头部的内容。

1. 单页面应用（SPA）和多页面应用（MPA）的优缺点？

<https://www.cnblogs.com/yunyea/p/8824178.html>

1. ISO时间格式转换为标准的时间格式:2018-07-22T16:00:00.000Z=> newDate()

let ds = new Date().toISOString().replace(/-/g,'/').replace(/T/g,' ');

var ts = Date.parse(ds.slice(0,ds.length-1));//获取距离1970/1/1午夜毫秒数

//var ts = Date.parse('2018/07/22 16:00:00.000');

ts = ts + 8 \* 3600 \* 1000;//加上8小时毫秒数

console.log(ts);

var d = new Date(ts); //Mon Jul 23 2018 17:11:25 GMT+0800 (中国标准时间)

1. 获取距离1970/1/1午夜之间的毫秒数

Date.getTime(); //参数为Date对象

Date.parse(‘2018/07/22 12:00:00.000’); //参数为一定时间格式的字符串

1. css-position属性：static, fixed, relative, absolute, sticky

sticky是css3新增的属性，它的行为就像 position:relative; 而当页面滚动超出目标区域时，它的表现就像 position:fixed;，它会固定在目标位置。（效果类似于现在很多网站的导航栏，一开始是正常的，下拉一定距离后后就变成了fixed的效果）。

1. flex布局（首选布局方式，抛弃以前的position+float吧）

Flex 布局教程：语法篇 <http://www.ruanyifeng.com/blog/2015/07/flex-grammar.html>

Flex 布局教程：实例篇 <http://www.ruanyifeng.com/blog/2015/07/flex-examples.html>

1. Axios学习资料 <https://www.jianshu.com/p/df464b26ae58>
2. axios请求会超时，或是其他错误，需要自动重新发送请求：

<https://github.com/softonic/axios-retry>

npm install axios-retry

import axiosRetry from 'axios-retry';

axiosRetry(axios,

{

retries: 3,//设置自动发送请求次数

retryCondition: (error)=>{

return (error.config.method === 'get');//true为打开自动发送请求，false为关闭自动发送请求

}

}

);

1. CSS盒子布局：正常模式和怪异模式

正常模式：width=content-width，padding,border另算

怪异模式：width=content+padding+border

如何切换元素的布局模式：box-sizing: content-box || border-box

1. Vue中数组和对象更新时一下情况无法响应式更新：

数组: arr[i] = newValue

对象：obj[‘newKey’] = newValue

如果arr= newArr 或是 obj = newObj 这样是可以响应式更新

1. Vue 异步组件加载：即在组件渲染的时候才会去加载对应的组件。

全局注册：Vue.component{

‘async-component’,

()=>import(‘./xxx.vue’)

}

局部注册： components:{

‘my-async-component’: ()=>import(‘./xxx.vue’)

}

1. Vue路由懒加载同上 ()=>import()
2. Vue路由传参：params 或 query.

query方式传参参数会显示在URL中，且刷新（F5）不会消失，且可以手动修改URL中的参数值来操作页面内容 （该方式不安全）

params方式传参参数不会显示在URL中，但是刷新后会消失。解决方法：在目标route的path路径中带上要传的参数名（path:’/test/:id’），这样刷新就不会消失

还有一个问题：当两种方法传参为obj类型时，刷新也会无法读取（自动转换为[object object]字符串）。解决方法：先转换为json字符串，在传。

1. 关于Vue-router 如何动态加载权限路由的参考资料：

<https://github.com/zzayne/iview-dynamicRouter>

1. 如何修改他人的项目并提交给作者

先用网页打开别人的项目，然后fork到自己的仓库中去。

再用自己仓库里对应项目的地址克隆到本地。

修改完成后，把在本地把所做的修改push推送到自己的github上。

再用网页打开自己的github项目，找到项目对应的new pull request按钮就可以推送给原作者了。

最后如果原作者接受的话，你的修改就会被合并到原作者的项目里。

1. webpack入门资料：<https://www.jianshu.com/p/b83a251d53db>

<https://www.jianshu.com/p/ef7b6e8ccf89>

<https://www.jianshu.com/p/42e11515c10f>

1. 在使用vue的项目中遇到的跨域问题解决方案(设置proxyTable----设置代理服务器)：
2. 使用vue-cli搭建的项目：

找到config->index.js文件，添加proxyTable属性：

核心：proxyTable配置的代理是把服务器端（另一个域）代理到开发环境的域

dev: { //开发环境下

// 静态资源文件夹

assetsSubDirectory: 'static',

// 发布路径

assetsPublicPath: '/',

// 代理配置表，在这里可以配置特定的请求代理到对应的API接口

// 例如将'localhost:8080/api/xxx'代理到'www.example.com/api/xxx'

proxyTable: {

'/api': {

target: 'http://xxxxxx.com', // 接口的域名

// secure: false, // 如果是https接口，需要配置这个参数

changeOrigin: true, // 如果接口跨域，需要进行这个参数配置

pathRewrite: {

'^/apis': '/api'

}

}

}

在ajax请求的时候，url写成'/apis/health/list'，当程序遇见/apis开头的路径时候，自动会将target补充上去。

1. 本人使用iview-admin为基础搭建的项目，没有找到config->index.js文件，所以采用以下的方案解决：

找到build->webpack.dev.config.js

在plugins:[],后面再添加一个属性(注意，这里的后面不是在plugins内部，而是在外部)：

//设置跨域代理

devServer: {

historyApiFallback: true,

hot: true,

inline: true,

stats: { colors: true },

proxy: {

//匹配代理的url

'/api': {

// 目标服务器地址

target: 'http://jaeger.ingress.ssdc.solutions',

//路径重写

pathRewrite: {'^/api' : '/api'},

changeOrigin: true

}

}

}

然后别忘了重新npm run dev 一下，不然更改的配置文件不会有效。

1. 发现一个不错的前端学习练手网站：<http://www.jqhtml.com/>
2. 如何用vue封装组件，然后发布到npm上：

<http://www.cnblogs.com/max-tlp/p/9338855.html>

<https://segmentfault.com/a/1190000012097778>

https://www.jianshu.com/p/5c20c909ad91

1. 使用vue-cli时采用完整的webpack模板和采用simple-webpack模板的区别（暂时还看不太懂，先留着）: <https://segmentfault.com/a/1190000011402931>
2. NPM资料：<https://www.npmjs.com.cn/cli/deprecate/>
3. Nodejs相关资料 <https://blog.csdn.net/vrix/article/details/51865368>
4. 关于flex布局遇到的一个问题：就是当子元素设为flex:auto时，子元素文本内容过多时不会自动换行，而是把子元素的宽度给撑开了。解决方法是：手动计算设置子元素的宽度：width: calc(100% - 18px);
5. Vue-router addRoutes动态添加路由后刷新无效问题解决。（比翼监管平台为例）
6. 首先在main.js中添加如下代码，目的是监听浏览器的刷新事件，并记录到本地是否触发了刷新事件。

window.addEventListener('beforeunload', e => {

localStorage.isFresh = true;

});

1. 然后是在router文件夹下的index.js文件下的router.beforeEach()添加如下代码。其中，localStroage.isFresh用来判断是否触发了浏览器的刷新事件，如果是，则执行后台接口获取权限路由数据并动态添加到router对象中，并设置localStroage.isFresh为false表示结束本次浏览器的刷新事件。然后判断sessionStorage.currentRouteName。sessionStorage.currentRouteName存放的是当前页面的路由名，用来刷新时导航使用。这里使用session而不是使用local的原因是用来判断用户是否重新打开了一个窗口来登录这个平台。如果sessionStorage.currentRouteName存在有效的值，那么刷新后导航到目标页面即可。如果是无效的值，则统一导航到首页。如果不是刷新事件，那就是普通的路由导航事件。此时需要根据sessionStorage.currentRouteName是否存在来判断用户是否是在新窗口中首次登录，如果是，则需要动态添加权限路由和菜单，如果不是，则不添加。然后直接导航到目标页面即可。

if(localStorage.isFresh == "true") {

addDynamicRouters();

sessionStorage.addedRouter = true;

localStorage.isFresh = false;

if(sessionStorage.currentRouteName && sessionStorage.currentRouteName!='null'){

next({

name: sessionStorage.currentRouteName

});

}else{

next('/');

}

}else{

if(!sessionStorage.currentRouteName){

if(sessionStorage.addedRouter != 'true'){

addDynamicRouters();

}

}

next();

}

1. sessionStorage.currentRouteName的值的改变在Main.vue中：

updated() {

sessionStorage.currentRouteName = this.$route.name;

},

1. iframe页面跳转刷新父页面窗口问题

先假设A,B,C,D都是页面，D是C的iframe，C是B的iframe，B是A的iframe，在D中跳转页面的写法区别如下：

js:跳转

window.location.href、location.href：本页面跳转，即D页面跳转

parent.location.href：上一层页面跳转，及C页面跳转

top.location.href: 最外层的页面跳转，即A页面跳转

页面刷新:

parent.location.reload(): C页面刷新

parent.main.location.reload(): parent是代表父级页面，main可能是父级页面里的一个iframe，整个意思就是让name=”main”的iframe页面刷新

window.opener.document.location.reload():C页面刷新（使用子窗口的opener对象来获得父窗口对对象）

top.location.reload()：A页面刷新

form或者链接:

D页面中有form

<form>: form提交后D页面跳转

<form target=”\_blank”>: form提交后弹出新页面

<form target=”\_parent”>: form提交后C页面跳转

<form target=”\_top”>: form提交后A页面跳转