小程序: WEB问题速查

1. 问题清单: 小程序->首页->VUE->day04,05

2. 高清无码小视频: 小程序->在线->VUE->day04,05

今日扩展视频: 小程序->在线->VUE->day04、05中所有视频

不用今天都看完！！！！！！\*\*\*两个月后\*\*\*鄙视面试前\*\*\*看完就行\*\*\*！

3. 单词列表: 小程序->单词->VUE->day04,05

第三阶段上课约定:

1. 上课千万不要跟着记笔记和写代码！

2. 拼写错误！自己找！让我帮你找，你需要手抄20遍，给我拍照！

3. 现在正处在照猫画虎的阶段！每个VUE效果至少写3遍:

3.1 注释+代码抄一遍！

3.2 保留注释，删除代码，把代码再填回来！

3.3 删除注释和代码，试着用自己的话把注释写回来，根据自己的注释，把代码填回来

一个VUE效果，只有自己能写出人话注释，才算自己会！——标准！

如何提问:

1. 上课时，案例中，笔记中那句话哪行代码看不懂，立刻微信私信问我！

2. 小程序中问题清单里哪个问题和回答看不懂！立刻微信私信问我！

3. 个人项目中哪个功能不会做！立刻微信私信问我！

4. 刷笔试题时，遇到不会的笔试题！立刻微信私信问我！

下载源代码: 右键->链接另存为！

正课:

1. 脚手架:

2. 脚手架文件夹结构

3. ES6的模块化开发

4. 避免组件间样式冲突

5. 学子商城项目的制作过程:

6. 懒加载:

7. http-proxy:

一. 脚手架:

对应小程序视频: 小程序->在线->VUE->day04 2. 使用vue/cli工具创建学子商城项目

1. 问题: 不同的公司，不同的人搭建项目的文件夹结构和文件名都大不相同！

2. 解决: Vue推出了一套标准的文件夹和文件结构——脚手架。

3. 何时: 今后，只要企业中创建项目，都用脚手架。

比如: nodejs express, 微信小程序, AngularJS, React

4. 如何生成脚手架项目代码: 2步:

(1). 先安装一个可以反复生成脚手架项目结构的命令行工具: (老母鸡)

npm i -g @vue/cli

当看到：+ @vue/cli@版本号，就说明安装成功

这一步在一台电脑上，只需要安装一次，即可使用

(2). 用命令行工具，创建一套新的脚手架代码:

vue create 项目名

a. Your connection to the default npm registry seems to be slow.

Use https://registry.npm.taobao.org for faster installation? (Y/n) Y

b. ? Please pick a preset:

Default ([Vue 2] babel, eslint)

Default (Vue 3 Preview) ([Vue 3] babel, eslint)

> Manually select features

c. ? Check the features needed for your project:

( ) Choose Vue version

>(\*) Babel //翻译工具, 将脚手架中时髦的ES6、ES7的新语法，翻译为等效的ES5代码。让绝大多数浏览器都可以运行Vue脚手架代码.

( ) TypeScript

( ) Progressive Web App (PWA) Support

(\*) Router

( ) Vuex

( ) CSS Pre-processors

( ) Linter / Formatter

( ) Unit Testing

( ) E2E Testing

d. Use history mode for router? (Requires proper server setup for index fallback in production) (Y/n) n

1). vue-router默认导航方式是hash方式： #/相对路径

2). 如果改为history模式，则地址栏中不包含#: /相对路径

必须服务器端工程师提供首页重定向配置.

e. Where do you prefer placing config for Babel, ESLint, etc.?

In dedicated config files

> In package.json

f. Save this as a preset for future projects? (y/N) N

g. 等待...

当看到Successfully created project 项目名，说明生成脚手架代码成功

将来，每做一个新项目，都可以重新运行一次，生成一套全新的脚手架代码副本.

5. 如何用vscode运行脚手架项目代码：

(1). 用vscode打开脚手架项目文件夹

(2). 右键点击package.json文件，选择"在集成终端/命令行中打开"

(3). 在打开的命令行窗口中：输入npm run serve //强调: 结尾没有r

看到: DONE Compiled successfully in 3426ms

并出现了Local: http://localhost:8080，说明脚手架项目运行成功

(4). 按住ctrl点http:// localhost:8080，自动打开浏览器

看到了脚手架自带的示例页面

从此，再不会使用live server了.

6. 脚手架项目支持热编译，就是只要一修改，一保存，脚手架不需要重启，就可自动重新编译新内容，自动刷新页面。所以, npm run serve不用反复运行！

特殊:

(1). 如果所做的修改，没有自动显示在网页中，可能是因为缓存，有必要重新运行npm run serve

(2). 如果修改了脚手架项目的配置文件，必须要重启npm run serve

二. 脚手架文件夹结构: (SPA)

对应小程序视频: 小程序->在线->VUE->day04 3. vue脚手架文件夹结构(含组件间样式冲突)

1. 唯一完整的HTML页面: 在脚手架中被一分为三:

(1). 唯一完整的HTML页面: public/index.html

(2). 我们SPA应用中的<div id="app"><router-view/></div>被提取到一个独立的src/App.vue文件中

(3). 我们SPA应用中的new VUe()也被提取到一个独立的src/main.js文件中

2. 页面组件: src/views/

(1). 每个页面组件不再是js文件，而是.vue文件

(2). js文件的问题:

a. 写HTML没提示

b. 没地方写css样式

(3). 什么是.vue文件: 集中保存一个组件的HTML+CSS+组件对象js代码的特殊的文件格式

(4). 每个.vue文件按标准包含3部分:

a. <template></template>元素: 专门写组件的template，HTML片段

只能有唯一父元素包裹

b. <script></script>元素: 专门写组件的对象，但是必须以模块化的方式来写

export default {

组件的内容

}

c. <style></style>元素：专门编写这个组件中所需的所有css样式.

(5). 好处:

a. HTML有提示了!

b. 也有专门的区域写CSS样式了

(6). 说明: 如果有的.vue文件特别简单:

a. 如果没有样式，可省略<style>

b. 如果没有js代码，可省略<script>

3. 路由器对象: src/router/index.js

const routes=[ ... ]

const router=new VueRouter({ ... , routes })

4. 公共组件或子组件: src/components/

每个公共组件或子组件，也都是一个.vue文件。

5. 示例: 封装一个页头组件:

(1). 在src/components文件夹下新建一个MyHeader.vue文件

(2). 将App.vue中<div id="nav">...</div>及其子内容剪切到MyHeader.vue中<template>元素下

(3). 将App.vue中<style>内所有#nav开头的css样式，剪切到MyHeader.vue中<style>内

强调: <style>不要加任何属性！

src/components/MyHeader.vue

|  |
| --- |
| <template>    <div id="nav">      <router-link to="/">Home</router-link> |      <router-link to="/about">About</router-link>    </div>  </template>  <style>    #nav {      padding: 30px;    }    #nav a {      font-weight: bold;      color: #2c3e50;    }    #nav a.router-link-exact-active {      color: #42b983;    }  </style> |

(4). 将MyHeader.vue引入到main.js中，将MyHeader转化为一个全局组件

a. 在main.js开头: import MyHeader from "./components/MyHeader.vue"

b. 在main.js中new Vue之前，Vue.component("my-header",MyHeader)

src/main.js

|  |
| --- |
| import Vue from 'vue'  import App from './App.vue'  import router from './router'  import store from './store'  import MyHeader from "./components/MyHeader"  Vue.config.productionTip = false  Vue.component("my-header",MyHeader)  new Vue({    router,    store,    render: h => h(App)  }).$mount('#app') |

(5). 在App.vue中<router-view/>上方添加<my-header></my-header>

src/App.vue

|  |
| --- |
| <template>    <div id="app">      <my-header></my-header>      <router-view/>    </div>  </template>  <style>  #app {    font-family: 'Avenir', Helvetica, Arial, sans-serif;    -webkit-font-smoothing: antialiased;    -moz-osx-font-smoothing: grayscale;    text-align: center;    color: #2c3e50;  }  </style>  运行结果: |

三. ES6的模块化开发:

1. 问题: 旧.html文件中，必须用<script>将所有网页要用的js文件都引入到HTML中。即长，又不直观。

2. 解决: 模块化开发

3. 什么是: 将程序的各个功能分别保存在模块对象中，谁，用到哪个模块，就可以直接引入哪个模块。不用都引入.html文件中。

4. 何时: 今后几乎所有的前端框架，都使用模块化开发  
5. 如何:

(1). 定义模块: 2种:

a. 如果模块内没有js代码，只有HTML+CSS，则这个.vue文件天生就是一个模块

b. 如果模块内包含js代码，则js的对象或函数，必须用export default关键字抛出到内存

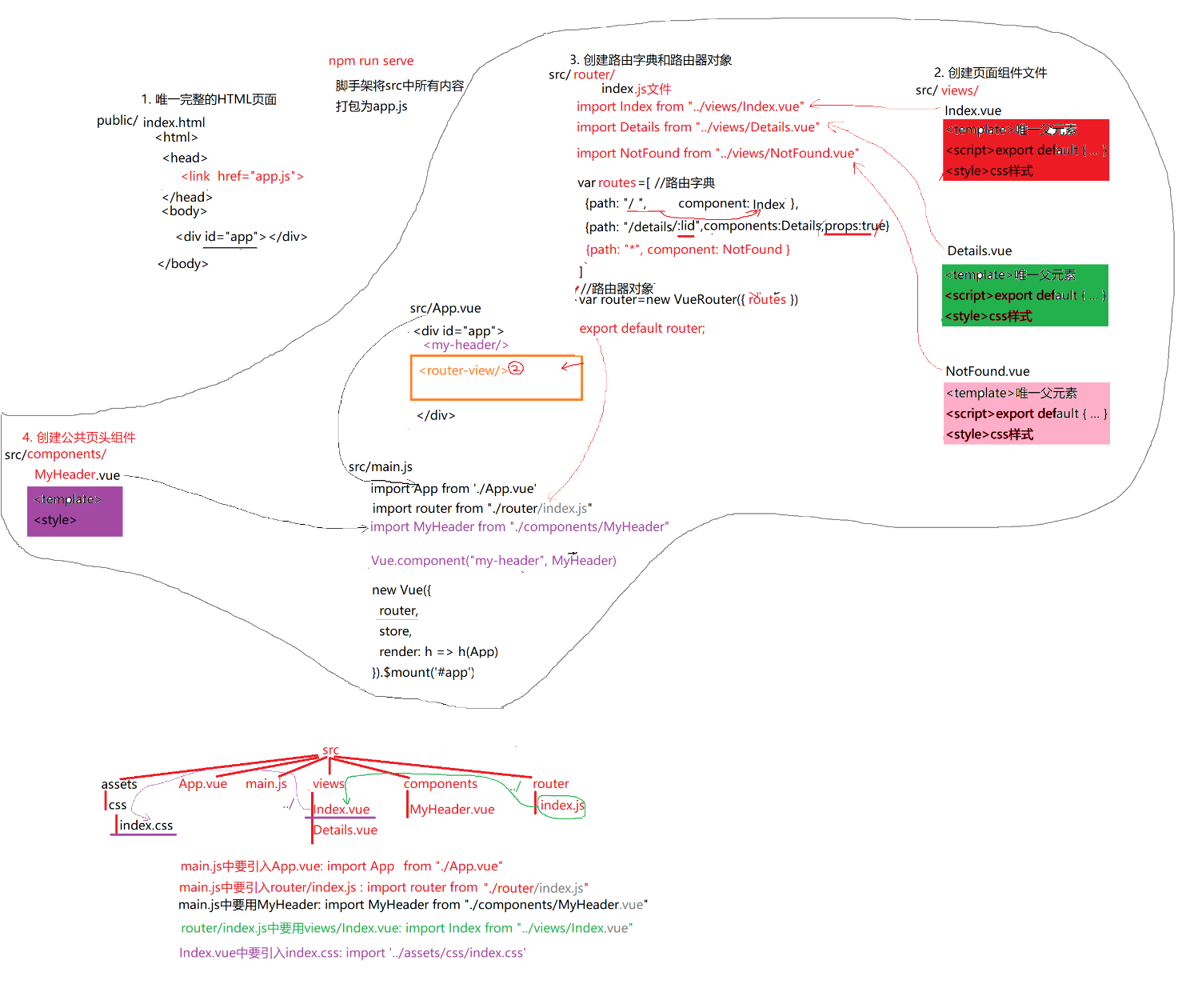
(2). 引入模块: 如果一段代码中需要使用另一个模块的功能:

import 自定义变量名 from "相对路径"

比如: 在src/main.js中，要用到src/components/MyHeader.vue

import MyHeader from "./components/MyHeader.vue" //.vue .js都可省略

结果: MyHeader中保存的就是一个模块对象，其中保存着引入来的模块内的内容.



四. 避免组件间样式冲突

对应小程序视频: 小程序->在线->VUE->day04 4. 扩展: 避免组件间样式冲突

1. 问题: 在两个.vue页面中分别定义相同的css选择器，但是css属性值不同。结果，只有一个.vue中的同名css选择器保留下来

2. 原因: 将来运行时，脚手架会把src中所有文件打包为一个app.js文件。无论写在哪个.vue中的css选择器最终都会合并到一个文件中并存。如果选择器相同，一定会发生冲突！

3. 解决:

(1). 偷懒的，但是不好的解决方法:

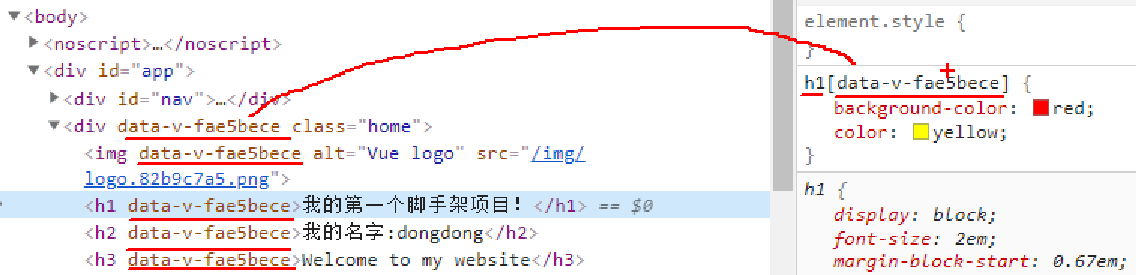
a. 给<style>元素添加scoped(范围内的)属性.

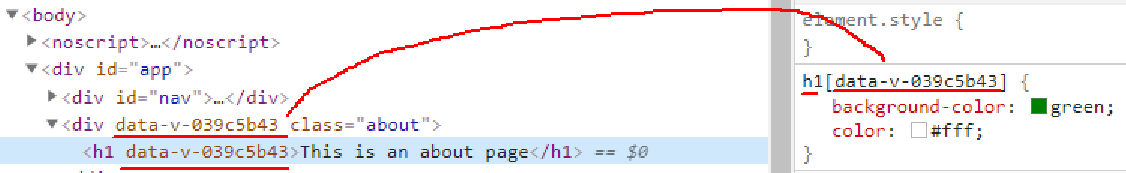
b. 原理:

1). scoped会给当前组件的所有元素都加"data-v-随机名"自定义属性

2). 为当前页面中的选择器自动添加一个[data-v-随机名]的属性选择器

3). 因为不同页面data-v-后的随机名不同，就可以保证不同页面，使用不同的选择器中的样式





c. 问题:

1). 效率低：为每个元素都加自定义扩展属性，工程量非常大

2). scoped不总是有效的:

i. vue中，也可以引入外部的css样式:

①. 在assets文件夹下css文件夹下定义每个页面所需的css文件

②. 两种方式引入: 二选一就行

在<script>中，import '../assets/css/xxx.css' —— vue的写法

在<style>中，@import url(../assets/css/xxx.css) —— css的写法

ii. scoped无法修改外部引入的css。

(2). 最好的解决组件间样式冲突的办法:

a. 2步:

1). 每创建一个组件，就在组件的唯一父元素上定义一个唯一的class名，比如:

i. home.vue中:

<div class="home">

...

ii. about.vue中

<div class="about">

2). 将来无论在哪里编写css，只要是这个组件的css选择器，一律从父元素class开始写

i. home.css中: .home>h1{ ... }

ii. about.css中: .about>h1{ ... }

b. 因为两个选择器不同了，所以可以并存，而不发生冲突。

c. 从此，<style>上不用加scoped了！

五. 学子商城项目的实现步骤:

对应小程序视频: 小程序->在线->VUE->day05 1. 学子商城首页

1. 先安装和配置axios

(1). 安装: 右键单击package.json，选择在集成终端中打开，在终端中输入npm i -save axios。

(2). 配置基础路径，并将axios对象添加到vue原型对象中:

a. 在main.js中new Vue()之前, 引入已经安装的axios

import axios from "axios"

b. 配置基础路径:

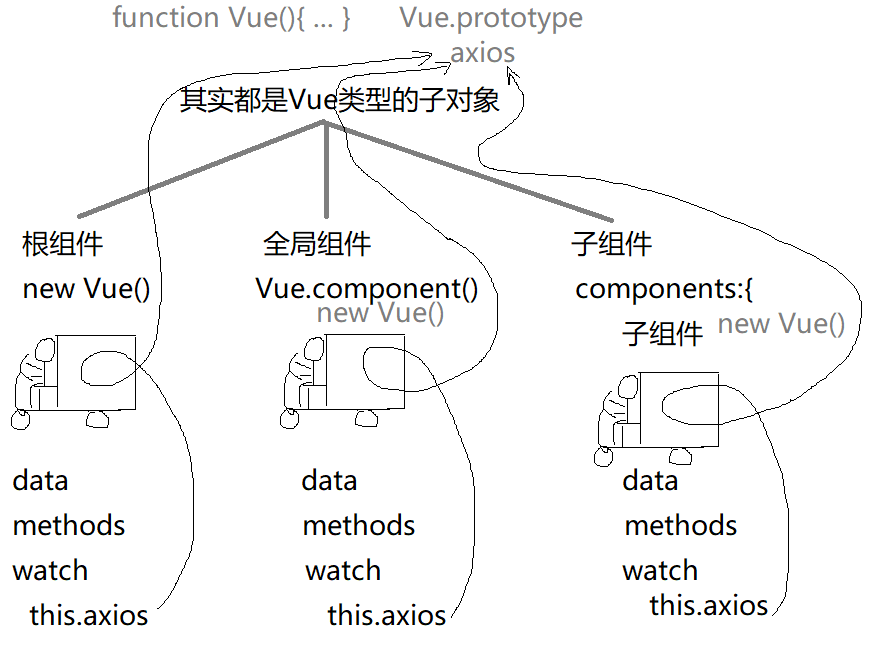
axios.defaults.baseURL="http://xzserver.applinzi.com"

c. 将axios对象添加到Vue.prototype中：

1). 无论根组件，还是全局组件，还是子组件，本质上都是Vue类型的子对象

2). 如果希望所有组件中都可以使用axios对象发送ajax请求，则可以将axios对象集中保存在vue原型对象中一份

3). 所有组件对象中都可用this.axios.get/post()发送请求



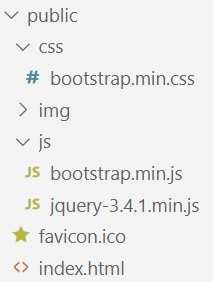
2. 基本的准备工作:

(1). 将项目要用的图片，都放在public文件夹下的img文件夹

(2). 将网站需要的第三方css和js文件，拷贝到public文件夹下的css文件夹和js文件夹。

a. 包括: bootstrap.min.css、jquery-1.11.3.min.js、bootstrap.min.js

b. 因为放在public中的问题，不会被打包和压缩，所以最好选择压缩版



3. 删除脚手架中views文件夹中旧页面，新建首页页面组件文件:Index.vue

4. 去router/index.js中路由字典中删除旧页面的路由地址，添加新页面的路由地址

import Index from "../views/Index"

import Details from "../views/details"

...

{path:"/", component: Index},

{path:"/details",component:Details}

5. 先将旧项目中所有页面都需要的基础样式（css/base.css），复制App.vue中<style>里

6. 将旧项目中index.html中的内容迁移到脚手架中Index.vue文件里

(1). 将旧项目中index.html中主体部分(<main id="main"></main>)元素整体复制到Index.vue中<template>下

(2). 将旧项目中css/index.css中所有css样式，都复制到Index.vue中<style>内。<style>必须加scoped。

7. 将旧项目中详情页product\_details.html内容迁移到Details.vue文件里

(1). 旧项目中product\_details.html中主体内容复制到Details.vue文件中<template>中

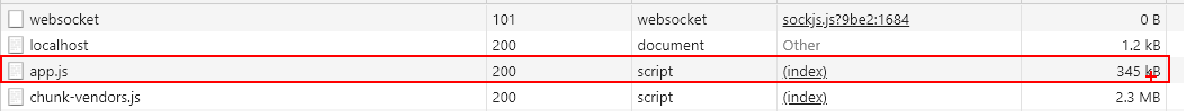
(2). 旧项目中css/product\_details.css中全部内容，复制到Index.vue中<style>内，必须加scoped。

六. 懒加载:

对应小程序视频: 小程序->在线->VUE->day05 -1. 懒加载

1. 问题: SPA应用最大的问题，就是首屏加载慢

2. 原因: 在首屏加载时就要下载所有页面组件内容，不管用户会不会看其它页面。



3. 解决: 懒加载

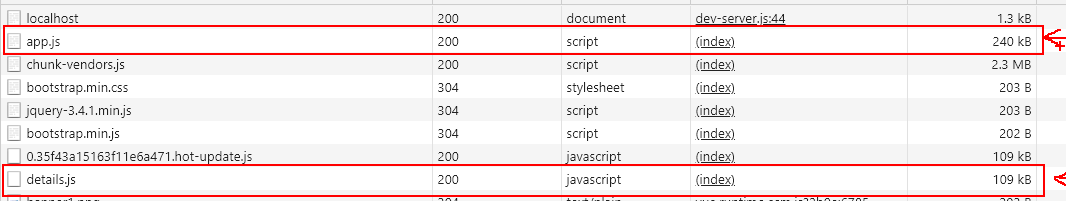
4. 什么是懒加载: 用户想看哪个页面，就只下载哪个页面。用户暂时不看的页面，要么暂时不下载，要么在后台异步悄悄下载。

5. 何时: 今后，除首页之外，其余页面都应该采用懒加载

6. 如何: 2种:

(1). 默认: 异步延迟加载

a. 什么是: 在用户查看首页内容时，在底层悄悄异步下载其他页面组件的内容。



b. 如何: router/index.js中

1). 不要提前import引入页面组件:

2). 将页面组件的路由字典项改为:

{

path: '/details',

component: () => import(/\* webpackChunkName: "details" \*/ '../views/Details.vue')

}

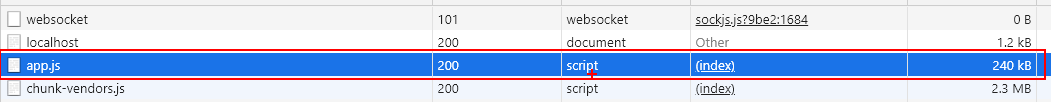
c. 结果: 首屏只下载首页组件，其他页面组件采用异步延迟下载方式，不影响首屏加载速度.

d. 优点: 当用户想查看其他页面时，很可能页面已经异步下载好了！——快

e. 缺点: 如果用户不想看其他页面，就是偷跑流量。

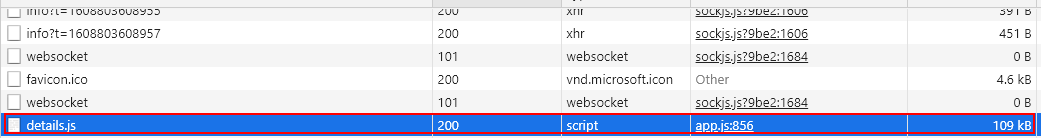
(2). 彻底懒加载:

a. 什么是: 如果用户首屏不需要看哪些页面，就一点儿也不下载.



b. 优点: 剩流量

c. 缺点: 当用户想看其他页面时，需要临时请求，下载——略慢



d. 如何:

1). 还要保留异步延迟加载的两步做法

i. 也不要提前import引入页面组件

ii. 路由字典项也要改为component:()=>import(...)

2). 额外为脚手架添加配置信息

i. 在脚手架根目录床架vue.config.js文件

ii. 在vue.config.js文件中，配置彻底懒加载

module.exports={

chainWebpack:config=>{

config.plugins.delete("prefetch")

//删除index.html开头的带有prefetch属性的link，不要异步下载暂时不需要的页面组件文件

},

}

七. http-proxy代理跨域:

对应小程序视频: 小程序->在线->VUE->day05 扩展0: Vue脚手架中使用http-proxy跨域

总结  
10. SPA

(1). 3步:

a. 先创建唯一完整的HTML页面

1). 包含vue基本的页面结构

<div id="app"> new Vue({el:"#app"})

2). 引入所有必要的文件和组件

vue-router.js, 其它页面或组件文件, router.js路由器对象所在文件

3). <div id="app">中用<router-view/>为今后页面组件预留空位

b. 再为每个页面组件创建独立的文件。每个页面组件其实都是一个子组件

c. 创建router.js文件，创建路由器对象

1). 创建路由字典对象:

var routes=[

{path:"/", component:首页组件对象名},

{path:"/相对路径" , component: 其它页面组件对象名},

{path:"\*", component: 404页面组件对象 }

]

2). 创建路由器对象，并将路由字典对象转入路由器对象中  
 var router=new VueRouter({ routes })

3). 将router对象加入到new Vue()中

回到唯一完整的HTML页面中: new Vue({ el:"#app", router })

(2). 页头等全局组件：

a. 创建独立的文件保存页头组件的内容

b. 使用Vue.component("my-header",{ ... })将页头创建为全局组件

c. 在唯一完整的HTML页面中引入页头组件文件

d. 使用页头组件标签<my-header/>: 2种:

1). 如果所有页面都有统一的页头:

就放在唯一完整的html页面中<router-view>外部上方

2). 如果有的页面有页头，有的页面没有页头:

就只放在需要页头的组件中的template中

(3). 路由跳转: 2种:

a. html中: <router-link to="/相对路径">文本<router-link>

b. js中: this.$router.push("/相对路径")

(4). 路由传参:

a. 修改路由字典:

{path:"/相对路径/:自定义参数名", component:页面组件对象, props:true}

b. 跳转时:

<router-link to="/相对路径/参数值"

或

this.$router.push("/相对路径/参数值")

c. 下一个页面接:

1). props:[ '自定义参数名' ]

2). 可将"自定义参数名"用作绑定或程序中都行

11. 脚手架文件夹结构:

(1). 唯一完整的HTML页面: 一分为三:

a. public文件夹

1). 图片img文件夹放在public文件夹下

2). 第三方css的压缩版和第三方js的压缩版都放在public文件夹下

3). 唯一完整的HTML文件index.html中，head中引入第三方的css和js

b. src/App.vue

1). <template>下只包含公共的页头组件和<router-view>

2). <style>下包含所有网页都要用到的公共css样式，比如css重置代码

c. src/main.js

1). import引入App.vue，router，axios，以及其他全局组件

2). 将全局组件对象转为真正的全局组件: Vue.component( "组件标签名", 全局组件对象 )

3). 配置axios并放入原型对象中:

axios.defaults.baseURL="服务器端基础路径"

Vue.prototype.axios=axios;

(2). 为每个页面创建.vue组件文件，都放在src/views文件夹下。每个.vue文件中:

a. <template>标签内，包含这个页面的HTML内容

b. <script>export default{ ... }</script>中包含组件对象的js内容。

c. <style>标签内包含仅这个页面组件内才使用的css

d. <template>中的HTML内容以及<script>export default{...}</script>中的js内容，和前四天将的是完全一样的写法，绑定，指令，函数，生命周期，axios请求等都一样。前四天怎么用，这里就怎么用。

(3). 路由字典和路由器对象，在src/router/index.js文件中

a. 仅import首页组件对象，不要过早引入其它页面组件

b. 路由字典中首页组件: { path:"/", component:首页组件对象}

c. 其余页面组件都做成懒加载:

{

path: '/相对路径',

component: () => import(/\* webpackChunkName: "组件名" \*/ '../views/其它页面组件.vue')

}

(4). 全局组件都放在src/components文件夹下，每个全局组件.vue文件中。但是，全局组件必须在main.js引入，并用Vue.component()转化为真正的全局组件，才能在其它组件的HTML中使用。

(5). 运行时: 路由器router对象监视浏览器地址栏路径变化，并查找对应的页面组件内容，先代替App.vue中的<router-view>，然后main.js再将包含要加载的页面内容的App.vue组件内容，替换到唯一完整的index.html中空<div id="app">位置。

12. 懒加载:

(1).异步延迟下载:

src/router/index.js

//不要import Details from "../views/Details"

const routes=[  
 ...

{

path:"/details/:lid",

component: () => import(

/\* webpackChunkName: "details" \*/

'../views/Details.vue'

),

props:true  
]

(2).彻底懒加载:   
保留异步懒加载2步

再在项目根目录创建vue.config.js

module.exports={

chainWebpack:config=>{

config.plugins.delete("prefetch")

//删除index.html开头的带有prefetch属性的link，不要异步下载暂时不需要的页面组件文件

},

}

13. http-proxy方式跨域:

在服务器端没有配置CORS或JSONP跨域的情况下:   
(1). vue.config.js中

module.exports={  
 ... ,

devServer: {

proxy: {

// detail: https://cli.vuejs.org/config/#devserver-proxy

'/api': {

target: `http://服务器端接口地址统一前缀部分`,

changeOrigin: true,

pathRewrite: {

'^/api' : '' //将程序中的/api，替换为空字符串，再和target中的基础路径拼接起来作为发送到服务器的最终请求地址。

}

}

}

}  
}  
(2). main.js中:   
...

axios.defaults.baseURL="/api"

14. 避免组件间样式冲突:

(1)<style scoped>  
(2)<template>  
 <组件父元素 class="组件名">  
 </template>  
 <style>  
 .组件名>子元素选择器{ ... }  
 </style>

15. watch+事件修饰符+防抖:

(1). 希望变量一变化就自动执行一项操作时:

data(){  
 return { 变量: 值 }  
},

watch:{  
 变量(){  
 要执行的操作

}  
}  
(2). 事件修饰符:

@事件.13 按回车时才执行

@事件.stop 阻止冒泡

@事件.prevent 阻止默认行为

(3). 防抖:

data(){

return {

... ,

timer:null

}

},

methods:{

查找方法(){

if(this.timer!=null){

clearTimeout(this.timer)

}

this.timer=setTimeout(()=>{

正式的查找操作

},等待时间)

}

}

今日对应小程序视频:

小程序->在线->VUE->day04

1. SPA单页面应用 all in one...

2. 使用vue/cli工具创建学子商城项目脚手架

3. vue脚手架文件夹结构(含组件间样式冲突)

4. 扩展: 避免组件间样式冲突

小程序->在线->VUE->day05

-1. 懒加载

1. 学子商城首页  
 扩展0: Vue脚手架中使用http-proxy跨域

作业:

1. 复习今日问题清单: 小程序->在线->VUE->day04、day05

2. 看小程序视频使用vue脚手架完成学子商城首页、详情页

先下载: 小程序->在线->VUE->day05 0. VUE版学子商城 起始空项目 中

旧js项目: public.zip

vue脚手架起始项目: xzvue\_start\_with\_axios.zip

观看并跟着做: 小程序->在线->VUE->day05

1. 学子商城首页

2. 学子商城详情页 。。。（不带放大镜）

3. 学子商城详情页 。。。（放大镜效果）

家里环境xampp和nodejs能用，且第一阶段学的好的同学:

可下载0. VUE版学子商城 起始空项目 中xzserver.zip，用自己的服务器端代码

家里环境xampp和nodejs不能用，或者第一阶段学的不好的同学：

无需下载xzserver.zip

视频中所有axios请求的服务器端接口地址，都改为我的新浪云服务器地址，就不需要你再运行服务器端了。也不需要你了解任何服务器端知识了:

http://localhost:5050/index 改为 http://xzserver.applinzi.com/index

http://localhost:5050/details 改为 http://xzserver.applinzi.com/details?lid=xxx