vue面试题总汇

vue的底层原理?

vue组件之间的通信?

JS中判断数据类型的方法有几种? 最常见的判断方法：typeof

判断已知对象类型的方法： instanceof

根据对象的constructor判断： constructor

无敌万能的方法：jquery.type()

vue与angular的区别? 1.vue仅仅是mvvm中的view层，只是一个如jquery般的工具库，而不是框架，而angular而是mvvm框架。 2.vue的双向邦定是基于ES5 中的 3.getter/setter来实现的，而angular而是由自己实现一套模版编译规则，需要进行所谓的“脏”检查，vue则不需要。因此，vue在性能上更高效，但是代价是对于ie9以下的浏览器无法支持。 4.vue需要提供一个el对象进行实例化，后续的所有作用范围也是在el对象之下，而angular而是整个html页面。一个页面，可以有多个vue实例，而angular好像不是这么玩的。 5.vue真的很容易上手，学习成本相对低，不过可以参考的资料不是很丰富，官方文档比较简单，缺少全面的使用案例。高级的用法，需要自己去研究源码，至少目前是这样。

说说你对angular脏检查理解？ 在angular中你无法判断你的数据是否做了更改，所以它设置了一些条件，当你触发这些条件之后,它就执行一个检测来遍历所有的数据，对比你更改的地方，然后执行变化。 这个检查很不科学。而且效率不高，有很多多余的地方，所以官方称为 脏检查。

active-class是哪个组件的属性？ vue-router模块的router-link组件。

嵌套路由怎么定义？ 在实际项目中我们会碰到多层嵌套的组件组合而成，但是我们如何实现嵌套路由呢？因此我们需要在 VueRouter 的参数中使用 children 配置，这样就可以很好的实现路由嵌套。index.html，只有一个路由出口

```
<!-- router-view 路由出口，路由匹配到的组件将渲染在这里 -->
<router-view></router-view>
```

main.js，路由的重定向，就会在页面一加载的时候，就会将home组件显示出来，因为重定向指向了

home组件，redirect的指向与path的必须一致。children里面是子路由，当然子路由里面还可以继续嵌套子路由。

```
import Vue from 'vue'
import VueRouter from 'vue-router'
Vue.use(VueRouter)

//引入两个组件

import home from "./home.vue"
import game from "./game.vue"
//定义路由
const routes = [
```

```
    { path: "/", redirect: "/home" },//重定向,指向了home组件
    {
        path: "/home", component: home,
        children: [
            { path: "/home/game", component: game }
        ]
    }
```

```
]
//创建路由实例
const router = new VueRouter({routes})

new Vue({
```

```
    el: '#app',
    data: {
    },
    methods: {
    },
    router
```

```
}) home.vue，点击显示就会将子路由显示在出来，子路由的出口必须在父路由里面，否则子路由无法显示。

game.vue

怎么定义vue-router的动态路由？怎么获取传过来的动态参数？ 在router目录下的index.js文件中，对path属性加上/:id。 使用router对象的params.id。
```

vue-router有哪几种导航钩子？ 三种， 第一种：是全局导航钩子：router.beforeEach(to,from,next)，作用：跳转前进行判断拦截。 第二种：组件内的钩子 第三种：单独路由独享组件

scss是什么？在vue.cli中的安装使用步骤是？有哪几大特性？ css的预编译。

使用步骤：

第一步：用npm 下三个loader（sass-loader、css-loader、node-sass）

第二步：在build目录找到webpack.base.config.js，在那个extends属性中加一个拓展.scss

第三步：还是在同一个文件，配置一个module属性

第四步：然后在组件的style标签加上lang属性 ，例如：lang=”scss”

有哪几大特性:

1、可以用变量，例如（$变量名称=值）； 2、可以用混合器，例如（） 3、可以嵌套

mint-ui是什么？怎么使用？说出至少三个组件使用方法？ 基于vue的前端组件库。npm安装，然后import样式和js，vue.use（mintUi）全局引入。在单个组件局部引入：import {Toast} from ‘mint-ui’。组件一：Toast(‘登录成功’)； 组件二：mint-header； 组件三：mint-swiper

v-model是什么？怎么使用？ vue中标签怎么绑定事件？ 可以实现双向绑定，指令（v-class、v-for、v-if、v-show、v-on）。vue的model层的data属性。绑定事件：

iframe的优缺点？ iframe也称作嵌入式框架，嵌入式框架和框架网页类似，它可以把一个网页的框架和内容嵌入在现有的网页中。

优点：

解决加载缓慢的第三方内容如图标和广告等的加载问题 Security sandbox 并行加载脚本 方便制作导航栏 缺点：

iframe会阻塞主页面的Onload事件 即时内容为空，加载也需要时间 没有语意 简述一下Sass、Less，且说明区别？ 他们是动态的样式语言，是CSS预处理器,CSS上的一种抽象层。他们是一种特殊的语法/语言而编译成CSS。 变量符不一样，less是@，而Sass是$; Sass支持条件语句，可以使用if{}else{},for{}循环等等。而Less不支持; Sass是基于Ruby的，是在服务端处理的，而Less是需要引入less.js来处理Less代码输出Css到浏览器

axios是什么？怎么使用？描述使用它实现登录功能的流程？ 请求后台资源的模块。npm install axios -S装好，然后发送的是跨域，需在配置文件中config/index.js进行设置。后台如果是Tp5则定义一个资源路由。js中使用import进来，然后.get或.post。返回在.then函数中如果成功，失败则是在.catch函数中

axios+tp5进阶中，调用axios.post(‘api/user’)是进行的什么操作？axios.put(‘api/user/8′)呢？ 跨域，添加用户操作，更新操作。

vuex是什么？怎么使用？哪种功能场景使用它？ vue框架中状态管理。在main.js引入store，注入。新建了一个目录store，….. export 。场景有：单页应用中，组件之间的状态。音乐播放、登录状态、加入购物车

mvvm框架是什么？它和其它框架（jquery）的区别是什么？哪些场景适合？ 一个model+view+viewModel框架，数据模型model，viewModel连接两个

区别：vue数据驱动，通过数据来显示视图层而不是节点操作。

场景：数据操作比较多的场景，更加便捷

自定义指令（v-check、v-focus）的方法有哪些？它有哪些钩子函数？还有哪些钩子函数参数？ 全局定义指令：在vue对象的directive方法里面有两个参数，一个是指令名称，另外一个是函数。组件内定义指令：directives

钩子函数：bind（绑定事件触发）、inserted(节点插入的时候触发)、update（组件内相关更新）

钩子函数参数：el、binding

说出至少4种vue当中的指令和它的用法？ v-if：判断是否隐藏；v-for：数据循环出来；v-bind:class：绑定一个属性；v-model：实现双向绑定

vue-router是什么？它有哪些组件？ vue用来写路由一个插件。router-link、router-view

导航钩子有哪些？它们有哪些参数？ 导航钩子有：

a/全局钩子和组件内独享的钩子。b/beforeRouteEnter、afterEnter、beforeRouterUpdate、beforeRouteLeave

参数：

有to（去的那个路由）、from（离开的路由）、next（一定要用这个函数才能去到下一个路由，如果不用就拦截）最常用就这几种

Vue的双向数据绑定原理是什么？ vue.js 是采用数据劫持结合发布者-订阅者模式的方式，通过Object.defineProperty()来劫持各个属性的setter，getter，在数据变动时发布消息给订阅者，触发相应的监听回调。

具体步骤：

第一步：需要observe的数据对象进行递归遍历，包括子属性对象的属性，都加上 setter和getter 这样的话，给这个对象的某个值赋值，就会触发setter，那么就能监听到了数据变化

第二步：compile解析模板指令，将模板中的变量替换成数据，然后初始化渲染页面视图，并将每个指令对应的节点绑定更新函数，添加监听数据的订阅者，一旦数据有变动，收到通知，更新视图

第三步：Watcher订阅者是Observer和Compile之间通信的桥梁，主要做的事情是: 1、在自身实例化时往属性订阅器(dep)里面添加自己 2、自身必须有一个update()方法 3、待属性变动dep.notice()通知时，能调用自身的update()方法，并触发Compile中绑定的回调，则功成身退。

第四步：MVVM作为数据绑定的入口，整合Observer、Compile和Watcher三者，通过Observer来监听自己的model数据变化，通过Compile来解析编译模板指令，最终利用Watcher搭起Observer和Compile之间的通信桥梁，达到数据变化 -> 视图更新；视图交互变化(input) -> 数据model变更的双向绑定效果。

请详细说下你对vue生命周期的理解？ 总共分为8个阶段创建前/后，载入前/后，更新前/后，销毁前/后

创建前/后： 在beforeCreated阶段，vue实例的挂载元素$el和数据对象data都为undefined，还未初始化。在created阶段，vue实例的数据对象data有了，$el还没有。

载入前/后：在beforeMount阶段，vue实例的$el和data都初始化了，但还是挂载之前为虚拟的dom节点，data.message还未替换。在mounted阶段，vue实例挂载完成，data.message成功渲染。

更新前/后：当data变化时，会触发beforeUpdate和updated方法。

销毁前/后：在执行destroy方法后，对data的改变不会再触发周期函数，说明此时vue实例已经解除了事件监听以及和dom的绑定，但是dom结构依然存在 请说下封装 vue 组件的过程？ 首先，组件可以提升整个项目的开发效率。能够把页面抽象成多个相对独立的模块，解决了我们传统项目开发：效率低、难维护、复用性等问题。

然后，使用Vue.extend方法创建一个组件，然后使用Vue.component方法注册组件。子组件需要数据，可以在props中接受定义。而子组件修改好数据后，想把数据传递给父组件。可以采用emit方法。

你是怎么认识vuex的？ vuex可以理解为一种开发模式或框架。比如PHP有thinkphp，java有spring等。 通过状态（数据源）集中管理驱动组件的变化（好比spring的IOC容器对bean进行集中管理）。

应用级的状态集中放在store中； 改变状态的方式是提交mutations，这是个同步的事物； 异步逻辑应该封装在action中。

vue-loader是什么？使用它的用途有哪些？ 解析.vue文件的一个加载器，跟template/js/style转换成js模块。

用途：js可以写es6、style样式可以scss或less、template可以加jade等

请说出vue.cli项目中src目录每个文件夹和文件的用法？ assets文件夹是放静态资源；components是放组件；router是定义路由相关的配置;view视图；app.vue是一个应用主组件；main.js是入口文件

vue.cli中怎样使用自定义的组件？有遇到过哪些问题吗？ 第一步：在components目录新建你的组件文件（smithButton.vue），script一定要export default {

第二步：在需要用的页面（组件）中导入：import smithButton from '../components/smithButton.vue'

第三步：注入到vue的子组件的components属性上面,components:{smithButton}

第四步：在template视图view中使用， 问题有：smithButton命名，使用的时候则smith-button。

聊聊你对Vue.js的template编译的理解？ 简而言之，就是先转化成AST树，再得到的render函数返回VNode（Vue的虚拟DOM节点）

详情步骤：

首先，通过compile编译器把template编译成AST语法树（abstract syntax tree 即 源代码的抽象语法结构的树状表现形式），compile是createCompiler的返回值，createCompiler是用以创建编译器的。另外compile还负责合并option。

然后，AST会经过generate（将AST语法树转化成render funtion字符串的过程）得到render函数，render的返回值是VNode，VNode是Vue的虚拟DOM节点，里面有（标签名、子节点、文本等等）

vue的历史记录 history 记录中向前或者后退多少步

vuejs与angularjs以及react的区别？ 1.与AngularJS的区别 相同点：

都支持指令：内置指令和自定义指令。

都支持过滤器：内置过滤器和自定义过滤器。

都支持双向数据绑定。

都不支持低端浏览器。

不同点：

1.AngularJS的学习成本高，比如增加了Dependency Injection特性，而Vue.js本身提供的API都比较简单、直观。

2.在性能上，AngularJS依赖对数据做脏检查，所以Watcher越多越慢。

Vue.js使用基于依赖追踪的观察并且使用异步队列更新。所有的数据都是独立触发的。

对于庞大的应用来说，这个优化差异还是比较明显的。

2.与React的区别 相同点：

React采用特殊的JSX语法，Vue.js在组件开发中也推崇编写.vue特殊文件格式，对文件内容都有一些约定，两者都需要编译后使用。

中心思想相同：一切都是组件，组件实例之间可以嵌套。

都提供合理的钩子函数，可以让开发者定制化地去处理需求。

都不内置列数AJAX，Route等功能到核心包，而是以插件的方式加载。

在组件开发中都支持mixins的特性。

不同点：

React依赖Virtual DOM,而Vue.js使用的是DOM模板。React采用的Virtual DOM会对渲染出来的结果做脏检查。

Vue.js在模板中提供了指令，过滤器等，可以非常方便，快捷地操作DOM。

vue生命周期面试题 什么是vue生命周期？ Vue 实例从创建到销毁的过程，就是生命周期。也就是从开始创建、初始化数据、编译模板、挂载Dom→渲染、更新→渲染、卸载等一系列过程，我们称这是 Vue 的生命周期。

vue生命周期的作用是什么？ 它的生命周期中有多个事件钩子，让我们在控制整个Vue实例的过程时更容易形成好的逻辑。

vue生命周期总共有几个阶段？ 它可以总共分为8个阶段：创建前/后, 载入前/后,更新前/后,销毁前/销毁后

第一次页面加载会触发哪几个钩子？ 第一次页面加载时会触发 beforeCreate, created, beforeMount, mounted 这几个钩子

DOM 渲染在 哪个周期中就已经完成？ DOM 渲染在 mounted 中就已经完成了

简单描述每个周期具体适合哪些场景？ 生命周期钩子的一些使用方法： beforecreate : 可以在这加个loading事件，在加载实例时触发 created : 初始化完成时的事件写在这里，如在这结束loading事件，异步请求也适宜在这里调用 mounted : 挂载元素，获取到DOM节点 updated : 如果对数据统一处理，在这里写上相应函数 beforeDestroy : 可以做一个确认停止事件的确认框 nextTick : 更新数据后立即操作dom

arguments是一个伪数组，没有遍历接口，不能遍历

cancas和SVG的是什么以及区别 SVG

SVG 是一种使用 XML 描述 2D 图形的语言。 SVG 基于 XML，这意味着 SVG DOM 中的每个元素都是可用的。您可以为某个元素附加 JavaScript 事件处理器。 在 SVG 中，每个被绘制的图形均被视为对象。如果 SVG 对象的属性发生变化，那么浏览器能够自动重现图形。

Canvas

Canvas 通过 JavaScript 来绘制 2D 图形。 Canvas 是逐像素进行渲染的。 在 canvas 中，一旦图形被绘制完成，它就不会继续得到浏览器的关注。如果其位置发生变化，那么整个场景也需要重新绘制，包括任何或许已被图形覆盖的对象。

Canvas 与 SVG 的比较

Canvas

依赖分辨率 不支持事件处理器 弱的文本渲染能力 能够以 .png 或 .jpg 格式保存结果图像 最适合图像密集型的游戏，其中的许多对象会被频繁重绘 SVG

不依赖分辨率 支持事件处理器 最适合带有大型渲染区域的应用程序（比如谷歌地图） 复杂度高会减慢渲染速度（任何过度使用 DOM 的应用都不快） 不适合游戏应用

vue 知识总结

一.v-model 只能用于表单元素的双向绑定,以后后面会学习的组件.

model:负责数据存储 view:负责页面展示 view model:负责业务逻辑处理(比如 Ajax 请求等)，对数据进行加工后交给视图展示

vue 编写步骤: 1.导入 vue 文件 2.确定一个范围来表示此范围内的所有指令都是被 vue 来解析的 3.通过 vue 来实例化一个 vue 对象,用这个对象来解析 id=app 中的所有指令.

改变 name 值 上面这个 v-on:可以简写为@,这是 Vue 里面的简写方法,

v-text 和 v-html 绑定

//

黑马程序员

//黑马程序员 它会将标签解析

new Vue({data:{

msg:'

黑马程序员

'}

});

v-cloak 解决表达式闪烁问题

{{msg}}new Vue({

1

data:{ msg:'hello ivan'

}})

{{}}插值表达式,双向数据绑定 v-text v-html v-cloak 解决表达式闪烁问题

v-model 双向数据绑定 //只能用于表单和组件的双向数据绑定 v-bind 给 html 元素或者组件动态地绑定一个或多个特性，例如动态绑定 style 和 class v-bind:src='imageSrc' 简写为 :src='imageSrc' v-for 根据数组中的元素遍历指定模板内容生成内容 v-if if结果是移除元素,和添加元素 根据表达式的值的真假条件来决定是否渲染元素，如果条件为false不渲 染(达到隐藏元素的目的)，为 true 则渲染。在切换时元素及它的数据绑定被销毁并重建

v-show show 是隐藏和显示来实现是否可视 表达式的真假值，切换元素的 display CSS 属性，如果为 false，则在元素上添加 display:none 来隐藏元素，否则 移除 display:none 实现显示元素

v-on 绑定事件监听，表达式可以是一个方法的名字或一个内联语句，如果没有修饰符也可以省略，用在普通的html 元素上时，只能监听 原生 DOM 事件。用在自定义元素组件上时，也可以监听子组件触发的自定义事件。

filterBy sname in 'name' 遍历 name,查找是否有与 sname 相同的内容,有就选出来,vue1 版本支持,2 版本不支持,会 报错. list.findIndex() //findIndex 内部会对数组进行遍历 遍历的过程中执行回调函数 只要当前遍历的这一项数据 满足了回调函数中的条件 那么就会终止循环 同时会把当前遍历的这个元素的索引进行返回 if(!confirm('是否要删除数据?')){ //提醒用户是否要删除数据

//当用户点击的取消按钮的时候，应该阻断这个方法中的后面代码的继续执行

return;}

2

二.《过滤器》

C:\Users\Administrator\AppData\Local\Google\Chrome\Application\chrome.exe 谷歌装插件--enable-easy-off-store-extension-install 844581

1:定义私有的过滤器、filters 在某一个 Vue 对象内部定义的过滤器称之为私有过滤器， 这种过滤器只有当前 Vue 对象 el 指定的监管区域有用。

//在某一个 vue 对象内部定义的过滤器称之为私有过滤器，//这种过滤器只有在当前 vue 对象 el 指定的监管的区域有用

filters:{ input 是自定义过滤器的默认参数，input 的值永远都是取自于 | 左边的内容

datefmt:function(input){ var year=input.getFullYear

返回的值就是过滤后显示在{{new Date() | datefmt }}中的数据return year

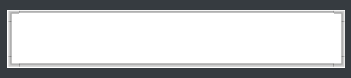
}}

2:全局过滤器、filter 注意没有加 s 一:在 Vue 里面添加 filter 方法的时候要写在 new Vue 前面，否则报错

Vue.filter('datefmt',function(input){ 执行体 }) 二:在 datefnt 里面可以传几个参数，参数可以拿来做判断，if 符合格式那么

return 这种格式的数据。 {{new Date() | datefmt('yyyy-mm-dd')}}

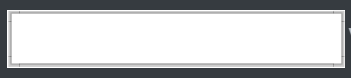
Vue.filter('datefmt',function(input){ 执行体 })3:按键修饰符:



vue 1.0 查看修饰符 Vue.directive('on').keyCodes.f2 = 113vue 2.0 Vue.config.keyCodes

可以这样来添加修饰符 Vue.config.keyCodes.f2 = 113

4:自定义指令:vue 1.0----------

 Vue.directive('focus',function(){

this.el 表示获取添加 v-focus 的当前标签this.vm 表示 var app=new Vue()当前模块

3

this.expression})

在使用指令的时候必须先 new Vue，创建模板进行控制。对象传入。。。

Vue.directive('focus',{inserted:function(el,binding){} })

5:数据传输请求报文行

6:ajax 请求 Get 请求数据

1:先引文件 vue-resource121.js，含有 get post 方法

2:methods:{click:function(){

this.$http.get('url').then(function(response){console.log(response.body)

})}

}

POST 请求数据

methods:{click:function(){

--这里是请求是要传的数据，类似于 data{ name:‘daiyu’ }this.$http.post('url',{name:name}, {emulateJSON:true}).then(function(response){

console.log(response.body)})

}}

7:生命周期、

在 new Vue 里面进行 ajax 请求的时候，只是在 methods 里面创建了一个 ajax 方法然后赋值数据是不会渲染到页面。

4

因为我们只是定义了一个 ajax 请求数据。 需要在添加 created:{ this.get() }来调用这个 ajax，那么请求就会发出

8:动画、 因为.show-enter-active,.show-leave-active 在整个动画过程中都是存在于元素上面的，所以

transition:all 1s ease;必须写在这个上面才能持续作用于元素，才能产生动画效果.show-enter-active,.show-leave-active{

transition:all 1s ease;.show-enter:标记动画开始的位置为 padding-left: 100px;

.show-leave-to:标记动画离开的结束位置为 padding-left: 100px.show-enter,.show-leave-to{

padding-left: 100px;.show-leave:标记动画离开时的初始位置为:padding-left: 10px;;

.show-enter-to:标记动画开始的结束位置为:padding-left: 10px;;.show-enter-to,.show-leave{

padding-left: 10px;

三.vue 中的组件定义和注册 一个组件相当于一个vue实例 但是注意data的区别，vue实例中的data是一个对象，里面是数据，但是组件中 的 data 是一个函数，必须返回一个对象，这个对象里面的才是数据。

一个模块可以包含多个组件.

1.引入 vue.js 2.创建一个全局的组件 必须写在 new Vue 上面 ，第一个参数是组件名，后面会写在 div 里面，用来展示，第 二个对象是指的是自己写的一个标签，标签 id 为#account ,标签内部是你要写的内容

路由 引入 vue221.js 文件,引入 Vue-router231.js 文件,vue-router 必须在 vue221.js 后面

四.npm install cnpm -g 安装这个 这是安装 cnpm

npm 使用方法介绍全局安装 -g:

全局安装的包位于 Node.js 环境的 node_modules 目录下，全局安装的包一般用于命令行工具 C:\Users\LEO\AppData\Roaming\npm 本地安装: 本地安装的包在当前目录下的 node_modules 里面，本地安装的包一般用于实际的开发工作 npm 常用的命令:

1、安装包(如果没有指定版本号，那么安装最新版本) npm install -g 包名称 (全局安装) npm install 包名称 (本地安装) 2、安装包的时候可以指定版本

npm install -g 包名称@版本号 3、卸载包 npm uninstall -g 包名 4、更新包(更新到最新版本) npm update -g 包名 5、查看包信息

6

npm info 包名开发环境(平时开发使用的环境)生产环境(项目部署上线之后的服务器环境)--save 向生产环境添加依赖 dependencies--save-dev 向开发环境添加依赖 DevDependencies

《day4》1:路由的传参

2:watch 检测自己本身的改变，自己改变则改变 computer 会监视依耐的 data 里面的属性，如果一个发生改变就会改变

3:webpack淘宝安装

npm install nrm -g nrm ls nrm use taobao nrm ls

npm install webpack -g

直接安装

npm install webpack -g4:打包

export()

引入 require('')cmd 命令:

webpack main.js build.js5:安装 cnpm

npm install cnpm -g

webpack.config.js 中的配置文件module.exports={ //export:主配置文件

entry:'./src/main.js', //指定打包的入口文件output:{

//_ _dirname 是查找当前所在文件的绝对路径 path : __dirname+'/dist', // 注意:webpack1.14.0 要求这个路径是一个绝对路径 filename:'build.js'

}}

我如果来到你们公司,能给我什么样的发展平台?五.vue 脚手架应用步骤:

vue-router 用来路由配置的 js 文件. 路由 main.js 项目入口文件 安装 vue-loader 解析.vue 文件

//记住这里是双下划线.

7

vue 脚手架安装步骤: vue-cli 命令行工具搭建 spa 应用

1、在 cmd 命令面板中执行:npm install --global vue-cli 全局安装 vue-cli 2、利用:vue init webpack projectName(自定义项目名称) 创建一个基于 webpack 模板的新项目 3、进入到项目名称文件夹中执行 npm install 安装项目所需依赖 4、运行 npm run dev 运行项目

--save-dev 和--save 的区别? npm install vue-loader vue-template-compiler babel-plugin-transform-runtime --save-dev 回车即可完成安装

包开发阶段会使用,依赖,运行阶段不使用了 npm install vue --save 回车即可完成安装 这个包发布以后运行阶段会依赖

vue axios 请求数据的步骤: //axios 集成(qs 序列化数据) 1.npm install axios --save; 2.npm install qs --save; 3.引入

import axios from 'axios';

import qs from 'qs';axios 中 get 请求:

这 个

axios.get(url).then(function(res){ // res.data 就是后台返回的数据

,在 resource 中是 res.bodyaxios.post(url,qs.stringify(数据对象)).then(function(res){

}) axios 中 post 请求:

}) 注意:axios 请求的回调函数中的 this 是 undefined 不是 vue 实例.尽量使用 s6 语法,避免因 this 指向问题而报错.

axios 使用: \1. 安装 axios npm install axios --save \2. 因为 axios 不支持 Vue.use()方法,所以为了所有组件能够使用,需要绑定到 Vue 原型中 import axios from 'axios'; Vue.prototype.$http = axios \3. 发送请求参照 vue-resource // get 请求 this.$http.get(url).then(function(res){

// 返回的响应数据在 res.data 属性中 console.log(res.data); // 回调函数内部的 this 不是当前数据请求方法的调用者 // 通常内部 this 是 window 在 node 平台 内部 this 是 undefined

}) // 一般使用箭头函数写法 this.$http.get(url).then((res)=>{

8

console.log(res.data);})

// post 请求(Content-Type:application/x-www-form-urlencoded 时需要配合 qs.stringify 使用) npm install qs --save; import qs from 'qs'; this.$http.post(url,qs.stringify({数据}).then(function(res){

// 返回的响应数据在 res.data 属性中

console.log(res.data);})

数组的拼接,apply 后面传入的是一个数组,call 后面传入的是一个一个的值.s6 中的箭头函数没有自己的作用域,

vue-resource/axios 请求数据,get,post,jsonp.

路由规则的一些问题:routes:[

{path:'/',redirect:'/home'},//匹配根目录 {path:'*',redirect:'/home'},//其他路由规则都匹配不到的路径会被这条路由规则匹配 {path:'/home',component:home}, {path:'/shopcar',component:shopcar}, {path:'/news/newslist',component:newslist}

]

六.es6 中变量与字符串的拼接 var url= `${common.apidomain}/api/getimageInfo/${this.id}`

如上面所示,拼接的内容都在 tab 键上方的``这个符号内部,并且变量用${}包括,不需要用+号.

当需要点击跳转的元素添加点击事件 fn,在函数 fn 内部使用 `this.$router.push('/字符串路径')` router-link 不能做逻辑判断就跳转了，可以使用 this.$router.push('url'); 这是路由自带的 push 方法,我们可以做一些逻辑判断再跳转，常用于登陆页面

$router 是全局路由,$route 指的是当前页面的理由.v-router 路由

v-resource 数据请求(ajax 请求)

如何获取一个真正的 id 的值? this.$route.params.id v-bind 动态绑定跳转 v-bind="{to:'/news/newsinfo/'+item.id}"

//item.id 是变量.前面常量用引号引起来

在 vue 模板中,

9

import {Toast} from 'mint-ui'; Vue.prototype.abc=Toast; 如果想全局使用 Toast,首先需要全局引入 Toast,再讲 Toast 绑定在 Vue 的原型上.

当页面写完时需要进行打包的时候,需要在 package.json 中的"scripts":{"dev": "webpack-dev-server --inline --hot --open --port 5008""dist":"webpack -p"

} 然后在 cmd 中运行 npm run dist 这里的 dist 就是在 scripts 中的命名,就可以实现打包, 和 npm run dev 热加载,自动刷新的功能类似.

1.ajax 请求串行和并行问题:考察的就是同步和异步

有三个 ajax 请求，如何让这三个 ajax 请求串行执行，即第一个执行完成后在执行另一个?

> 如何让他们并行执行，然后三个请求都执行完成后，再执行某个操作?

1.2.

串行执行分两种 一是用同步模式 async: false，三个 ajax 请求连着写就可以了。 $.ajax({

url: "ajax 请求 1", async: false, success: function (data) {

console.log("ajax 请求 1 完成");}

});$.ajax({

10

url: "ajax 请求 2", async: false, success: function (data) {

console.log("ajax 请求 2 完成");}

});$.ajax({

url: "ajax 请求 3", async: false, success: function (data) {

console.log("ajax 请求 2 完成");}

}); 二是用异步模式 async: true，三个 ajax 请求嵌套写。 $.ajax({

url: "ajax 请求 1", async: true, success: function (data) {

console.log("ajax 请求 1 完成");$.ajax({

url: "ajax 请求 2", async: true, success: function (data) {

console.log("ajax 请求 2 完成");$.ajax({

url: "ajax 请求 3", async: true, success: function (data) {

console.log("ajax 请求 3 完成");}

});}

});}

});

并行执行就只能用异步模式。并设置变量进行计数

var num = 0; function isAllSuccess() {

num++; if (num>=3)

console.log("三个 ajax 请求全部完成");}

$.ajax({

11

url: "ajax 请求 1", async: true, success: function (data) {

console.log("ajax 请求 1 完成");

isAllSuccess();}

});$.ajax({

url: "ajax 请求 2", async: true, success: function (data) {

console.log("ajax 请求 3 完成");

isAllSuccess();}

});$.ajax({

url: "ajax 请求 3", async: true, success: function (data) {

console.log("ajax 请求 3 完成");

isAllSuccess();}

});

2.请写出一些前端性能优化的方式，越多越好

3.一个页面从输入 URL 到页面加载显示完成，这个过程中都发生了什么?(流程说的越详细越好)

1.减少 dom 操作 2.部署前，图片压缩，代码压缩 3.优化 js 代码结构，减少冗余代码 4.减少 http 请求，合理设置 HTTP 缓存 5.使用内容分发 cdn 加速 6.静态资源缓存 7.图片延迟加载

输入地址 1.浏览器查找域名的 IP 地址 2.这一步包括 DNS 具体的查找过程，包括:浏览器缓存->系统缓存->路由器缓存... 3.浏览器向 web 服务器发送一个 HTTP 请求 4.服务器的永久重定向响应(从 http://example.com 到 http://www.example.com) 5.浏览器跟踪重定向地址 6.服务器处理请求 7.服务器返回一个 HTTP 响应 8.浏览器显示 HTML 9.浏览器发送请求获取嵌入在 HTML 中的资源(如图片、音频、视频、CSS、JS 等等) 10.浏览器发送异步请求

12

4.请大概描述下页面访问 cookie 的限制条件1.跨域问题

2.设置了 HttpOnly 5.描述浏览器重绘和回流，哪些方法能够改善由于 dom 操作产生的回流 1.直接改变 className，如果动态改变样式，则使用 cssText

Vue 实例有一个完整的生命周期，也就是从开始创建、初始化数据、编译模板、挂载 Dom→渲染、更新→渲染、卸载等一系列过程，我们称这是 Vue 的生命周期。通俗说就是 Vue 实例从创建到销毁的过程，就是生命周期。

钩子函数:一个系统内置的一批函数,这些函数会在系统运行的某一阶段或者某一时间点自动触发,进行事件的处理后又接着执行 后续任务

2.让要操作的元素进行”离线处理”，处理完后一起更新

a) 使用 DocumentFragment 进行缓存操作,引发一次回流和重绘; b) 使用 display:none 技术，只引发两次回流和重绘; c) 使用 cloneNode(true or false) 和 replaceChild 技术，引发一次回流和重绘

6.vue 生命周期钩子

beforecreate : 可以在这加个 loading 事件，在加载实例时触发 created : 初始化完成时的事件写在这里，如在这结束 loading 事件，异步请求也适宜在这里调用 mounted : 挂载元素，获取到 DOM 节点 updated : 如果对数据统一处理，在这里写上相应函数

beforeDestroy : 可以做一个确认停止事件的确认框 常用生命周期函数: // 执行时机:在 data 中的数据和 methods 中的方法绑定到实例对象完毕是执行 // 发送请求一般都写在这个方法中 created:function(){

console.log('2.0 created',this.msg);

console.log(this.$el);},

// 执行时机:真实 DOM 渲染完毕时执行 // 如果需要操作真实 DOM,那么代码写在 mounted 方法中 mounted:function(){

console.log('4.0 mounted',this.msg);

console.log(this.$el);},

// 执行时机:data 中的数据发生变化会去执行 // 监测数据的变化 updated:function(){

console.log('5.0 updated',this.msg);}

1.beforcreate

13

2.created3.beformount4.mounted5.beforeUpdate6.updated7.actived8.deatived9.beforeDestroy10.destroyed

7.js 跨域请求的方式，能写几种是几种

1、通过 jsonp 跨域 2、通过修改 document.domain 来跨子域 3、使用 window.name 来进行跨域 4、使用 HTML5 中新引进的 window.postMessage 方法来跨域传送数据(ie 67 不支持) 5、CORS 需要服务器设置 header :Access-Control-Allow-Origin。 6、nginx 反向代理 这个方法一般很少有人提及，但是他可以不用目标服务器配合，不过需要你搭建一个中转 nginx 服务器，用于转发请求

8.对前端工程化的理解

开发规范
模块化开发
组件化开发
组件仓库
性能优化
项目部署
开发流程
开发工具

9.js 深度复制的方式

10.js 设计模式

总体来说设计模式分为三大类:1)创建型模式，共五种:工厂方法模式、抽象工厂模式、单例模式、建造者模式、原型模式。2)结构型模式，共七种:适配器模式、装饰器模式、代理模式、外观模式、桥接模式、组合模式、享元模式。3)行为型模式，共十一种:策略模式、模板方法模式、观察者模式、迭代子模式、责任链模式、命令模式、备忘录模式、状态模式、访问者模式、中介者模

11iframe 有那些缺点? 1)iframe 会阻塞主页面的 Onload 事件;

2)搜索引擎的检索程序无法解读这种页面，不利于 SEO;

1.使用 jq 的$.extend(true, target, obj) 2.newobj = Object.create(sourceObj)，// 但是这个是有个问题就是 newobj 的更改不会影响到 sourceobj 但是 sourceobj 的更改会影响到 newObj 3.newobj = JSON.parse(JSON.stringify(sourceObj))

14

3)iframe 和主页面共享连接池，而浏览器对相同域的连接有限制，所以会影响页面的并行加载。使用 iframe 之前需要考虑这两个缺点。如果需要使用 iframe，最好是通过 javascript 动态给 iframe4)添加 src 属性值，这样可以绕开以上两个问题。

12.Javascript 图片预加载详解 预加载图片是提高用户体验的一个很好方法。图片预先加载到浏览器中，访问者便可顺利地在你的网站上冲浪， 并享受到极快的加载速度。这对图片画廊及图片占据很大比例的网站来说十分有利，它保证了图片快速、无缝地 发布，也可帮助用户在浏览你网站内容时获得更好的用户体验。本文将分享三个不同的预加载技术，来增强网站 的性能与可用性。 方法一:用 CSS 和 JavaScript 实现预加载 缺点:在实际实现过程中会耗费太多时间 方法二:仅使用 JavaScript 实现预加载 1)只需简单编辑、加载所需要图片的路径与名称即可，该方法尤其适用预加载大量的图片 比如:预加载图片数量达 50 多张。将该脚本应用到登录页面，只要用户输入登录帐号，大部分画廊图片将被预 加载。 2)将该脚本封装入一个函数中，并使用 addLoadEvent()，延迟预加载时间，直到页面加载完毕。 方法三:使用 Ajax 实现预加载 使用 Ajax 实现图片预加载的方法。该方法利用 DOM，不仅仅预加载图片，还会预加载 CSS、JavaScript 等相关 的东西。使用 Ajax，比直接使用 JavaScript，优越之处在于 JavaScript 和 CSS 的加载不会影响到当前页面。该方 法简洁、高效。

13.图片懒加载:延迟加载图片或符合某些条件才开始加载图片 懒加载原理:浏览器会自动对页面中的 img 标签的 src 属性发送请求并下载图片。通过动态改变 img 的 src 属性 实现。 当访问一个页面的时候，先把 img 元素或是其他元素的背景图片路径替换成 loading 图片地址(这样就只需请求 一次) 等到一定条件(这里是页面滚动到一定区域)，用实际存放 img 地址的 laze-load 属性的值去替换 src 属性，即可 实现’懒加载’。 //即使 img 的 src 值为空，浏览器也会对服务器发送请求。所以平时做项目的时候，如果 img 没有用到 src，就不 要出现 src 这个属性

先上三个重要的知识点:

1.获取屏幕可视窗口大小: document.documentElement.clientHeight 标准浏览器及低版本 IE 标准模式 document.body.clientHeight 低版本混杂模式 2.元素相对于文档 document 顶部 element.offsetTop 3.滚动条滚动的距离 ocument.documentElement.scrollTop 兼容 ie 低版本的标准模式

document.body.scrollTop 兼容混杂模式; 滚动加载:当图片出现在可视区域时，动态加载该图片。 原理:当图片元素顶部是否在可视区域内，(图片相对于文档 document 顶部-滚动条滚动的距离) 实现原理: 1 首先从所有相关元素中找出需要延时加载的元素，放在 element_obj 数组中

15

2..判断数组中的 img 对象，若满足条件，则改变 src 属性 3..滚动的时候触发事件，1000 毫秒后执行 lazy()方法。 整部分代码位于闭包自执行函数中。相应的方法放在 init 中 使用格式 :src 填默认 loading 图片地址，真实的图片地址填在 lazy-src 属性里，切记需指定宽高。在外部调用 lazyLoad.init();

14.从输入 url 到页面展示到底发生了什么 1、输入地址 2、浏览器查找域名的 IP 地址 3、浏览器向 web 服务器发送一个 HTTP 请求 4、服务器的永久重定向响应 5、浏览器跟踪重定向地址

6、服务器处理请求 7、服务器返回一个 HTTP 响应 8、浏览器显示 HTML 9、浏览器发送请求获取嵌入在 HTML 中的资源(如图片、音频、视频、CSS、JS 等等)

什么是 DNS?Domain Name System，域名系统 因特网上作为域名和 IP 地址相互映射的一个分布式数据库，能够使用户更方便的访问互联网，而不用去记住能 够被机器直接读取的 IP 数串。通过主机名，最终得到该主机名对应的 IP 地址的过程叫做域名解析(或主机名解 析) 2)DNS 查询的两种方式:递归查询和迭代查询 客户端向服务器发起 http 请求的时候，会有一些请求信息，请求信息包含三个部分: 请求方法 URI 协议/版本 请求头(Request Header) 请求正文: 一.TCP 三次握手 第一次握手:客户端 A 将标志位 SYN 置为 1,随机产生一个值为 seq=J(J 的取值范围为=1234567)的数据包到服 务器，客户端 A 进入 SYN_SENT 状态，等待服务端 B 确认;

第二次握手:服务端 B 收到数据包后由标志位 SYN=1 知道客户端 A 请求建立连接，服务端 B 将标志位 SYN 和ACK 都置为 1，ack=J+1，随机产生一个值 seq=K，并将该数据包发送给客户端 A 以确认连接请求，服务端 B 进入 SYN_RCVD 状态。

第三次握手:客户端 A 收到确认后，检查 ack 是否为 J+1，ACK 是否为 1，如果正确则将标志位 ACK 置为 1，ack=K+1，并将该数据包发送给服务端 B，服务端 B 检查 ack 是否为 K+1，ACK 是否为 1，如果正确则连接建立成功，客户端 A 和服务端 B 进入 ESTABLISHED 状态，完成三次握手，随后客户端 A 与服务端 B 之间可以开始传输数据了。为什需要三次握手?三次握手”的目的是“为了防止已失效的连接请求报文段突然又传送到了服务端，因而产生错误”

二、TCP 四次挥手 第一次挥手:Client 发送一个 FIN，用来关闭 Client 到 Server 的数据传送，Client 进入 FIN_WAIT_1 状态。

第二次挥手:Server 收到 FIN 后，发送一个 ACK 给 Client，确认序号为收到序号+1(与 SYN 相同，一个 FIN 占 用一个序号)，Server 进入 CLOSE_WAIT 状态。 第三次挥手:Server 发送一个 FIN，用来关闭 Server 到 Client 的数据传送，Server 进入 LAST_ACK 状态。

16

第四次挥手:Client 收到 FIN 后，Client 进入 TIME_WAIT 状态，接着发送一个 ACK 给 Server，确认序号为收到序号+1，Server 进入 CLOSED 状态，完成四次挥手。

三.为什么建立连接是三次握手，而关闭连接却是四次挥手呢? 这是因为服务端在 LISTEN 状态下，收到建立连接请求的 SYN 报文后，把 ACK 和 SYN 放在一个报文里发送给 客户端。而关闭连接时，当收到对方的 FIN 报文时，仅仅表示对方不再发送数据了但是还能接收数据，己方也未 必全部数据都发送给对方了，所以己方可以立即 close，也可以发送一些数据给对方后，再发送 FIN 报文给对方 来表示同意现在关闭连接，因此，己方 ACK 和 FIN 一般都会分开发送。

四.服务器的永久重定向响应: 服务器给浏览器响应一个 301 永久重定向响应，这样浏览器就会访问“http://www.google.com/” 而非 “http://google.com/”。

为什么服务器一定要重定向而不是直接发送用户想看的网页内容呢?其中一个原因跟搜索引擎排名有关。如果一个页面有两个地址，就像 http://www.yy.com/和 http://yy.com/，搜索引擎会认为它们是两个网站，结果造成每个搜索链接都减少从而降低排名。而搜索引擎知道 301 永久重定向是什么意思，这样就会把访问带 www 的和不带 www的地址归到同一个网站排名下。还有就是用不同的地址会造成缓存友好性变差，当一个页面有好几个名字时，它可能会在缓存里出现好几次。

1)301 和 302 的区别:

301 和 302 状态码都表示重定向，就是说浏览器在拿到服务器返回的这个状态码后会自动跳转到一个新的 URL 地址，这个地址可以从响应的 Location 首部中获取(用户看到的效果就是他输入的地址 A 瞬间变成了另一个地址 B)——这是它们的共同点。

不同在于。301 表示旧地址 A 的资源已经被永久地移除了(这个资源不可访问了)，搜索引擎在抓取新内容的同时也将旧的网址交换为重定向之后的网址;

302 表示旧地址 A 的资源还在(仍然可以访问)，这个重定向只是临时地从旧地址 A 跳转到地址 B，搜索引擎会抓取新的内容而保存旧的网址。 SEO302 好于 301

2)重定向原因: (1)网站调整(如改变网页目录结构); (2)网页被移到一个新地址; (3)网页扩展名改变(如应用需要把.php 改成.Html 或.shtml)。 这种情况下，如果不做重定向，则用户收藏夹或搜索引擎数据库中旧地址只能让访问客户得到一个 404 页面错误 信息，访问流量白白丧失;再者某些注册了多个域名的网站，也需要通过重定向让访问这些域名的用户自动跳转 到主站点等。

3)什么时候进行 301 或者 302 跳转呢? 当一个网站或者网页 24—48 小时内临时移动到一个新的位置，这时候就要进行 302 跳转，而使用 301 跳转的场 景就是之前的网站因为某种原因需要移除掉，然后要到新的地址访问，是永久性的。 清晰明确而言:使用 301 跳转的大概场景如下: 1、域名到期不想续费(或者发现了更适合网站的域名)，想换个域名。 2、在搜索引擎的搜索结果中出现了不带 www 的域名，而带 www 的域名却没有收录，这个时候可以用 301 重定

17

向来告诉搜索引擎我们目标的域名是哪一个。3、空间服务器不稳定，换空间的时候

项目开发流程 第一步 拿到项目，整个团队先开个会，团队成员有 前端 ，后端 ui 产品设计，项目经理 第二步 讨论整个项目的需求，以及整个项目用什么技术去开发 ，是用 jq 去写 ，还是用模块化开发

去完成 比如用 Vue react angular 去实现

第三步 在会议上去要讨论，整个项目的技术难点在哪里，后期准备怎么去处理这个技术难点，需要后端

人员怎么样的配合

第四步 根据 ui 设计的原型，前端开始分配任务，初步估计出大概需要多少个工作日完成

第五步 开始干活，还原 ui 设计图，等把静态页面写完成后，再开始准备实现动态的数据处理跟内容展示

如果，后台数据接口，还没有写好，那么自己会先去写一些本地 json 数据，自己会先进行数据的展示跟处理，保证自己

写的静态页面，整体数据展示，以及 ajax 去请求 都没有问题

第六步 合并前端同事的代码，保证整个项目的排版，布局都没有问题

第七步 等后台的数据接口写好后，就开始前后端联调。

第八步，一个页面或者一个模块联调完成后，就交给测试人员去测试，然后接下来进行下一个页面的联调

第九步 如果测试人员测试出来了问题，就会以邮件的形式通知我们开发团队，哪里需要改，改成什么样的

最后 直到所有的页面都调试完成

第十步 开始准备打包，压缩，部署项目上线(webpack 进行项目的自动化处理). 通过 git 进行代码的上传跟下载

第十一步 项目上线后，开始进行性能的优化(图片的懒加载，预加载优化，抽取公共样式，优化代码)以及后期的项目的维护跟迭代更新