负责项目的搭建，与产品、UI沟通，规范代码书写，页面布局的实现并完成登录功能，权限管理页面，数据统计页面

1. 使用 vue-cli快速构建后台管理系统单页应用；
2. 根据UI组设计稿，快速搭建后台管理系统中登录功能页面、后台首页页面、用户管理页面、权限管理页面、用户反馈页面、以及数据统计页面六个页面的基本布局
3. 登陆功能完成获取数据，验证登陆信息，验证是否非空，用户名是否正确，用户密码是否正确。提交完数据并验证成功后就实行页面跳转。
4. 首页的侧边栏是通过element组件做的,通过后端给的数据渲染,根据不同的权限给到不同的功能,以达到侧边栏栏的页面渲染。 完成请求拦截器与响应拦截器，实现实时请求；
5. 完成权限管理中页面的数据渲染与用户列表的结构样式及数据渲染
6. 实现数据统计页面的数据驱动视图；
7. 实现数据统计页面页面渲染，动画实现，使用ECharts实现数据统计分析
8. 利用Vue的双向数据绑定，组件间的数据交互，减少DOM操作，提升代码维护性

***vue全家桶：***

Vue-cli项目构建工具 vue-router 路由 vuex状态管理 axios http 请求工具 webpack

**1 vue命令**

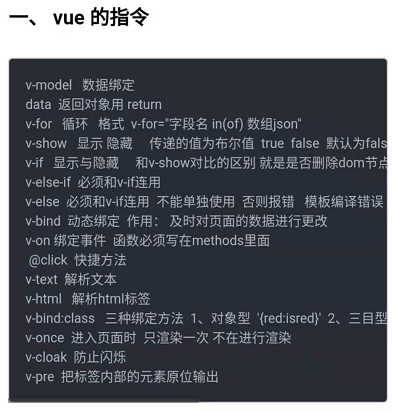
v-if：判断是否隐藏；

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

v-on 指令：添加一个事件监听器 ；



**2 Vue 项目如何启动**

命令npm run build bou 的

**3 如何使用vue-cli构建项目**

开始运行 npm run dev

访问项目 ocalhost:8080 楼购后死特

**4 表单验证**

## 首先在el-form标签加上ref和:rules属性配置校验规则对象，并且在每个el-form-item标签里定义prop属性校验规则，prop属性跟:rules里面的每项对应，在data里面定义rules属性值 ， 然后 element-ui表单组件rule属性校验规则对象，最后给表单组件加ref属性，获取组件实例（dom对象）在调用校验函数对整个表单进行校验

el-form el 佛木 rules 入了死 el-form-item el 佛木 哎特木 element-ui 哎了门 优哎 Prop 普弱普

**5导航守卫**

*vue怎么实现页面的权限控制*

利用 **vue-router** 的 **beforeEach** 事件，可以在跳转页面前判断用户的权限（利用 cookie 或 token），是否能够进入此页面，如果不能则提示错误或重定向到其他页面，在后台管理系统中这种场景经常能遇到。

使用 router.beforeEach 注册一个全局前置守卫，判断用户是否登陆

beforeEach 比佛义尺 cookie 克kei

# 6 vue 路由跳转四种方式

### **1 router-link**

编程式导航router.push(location) push破吃

### 2 this.$router.push() (函数里面调用)

**跳转到指定url路径，并想****history栈中添加一个记录，点击后退会返回到上一个页面 history kei 死特瑞**

### 3 this.$router.replace()  replace 瑞普雷斯

**跳转到指定url路径，但是history栈中不会有记录，点击返回会跳转到上上个页面 (就是直接替换了当前页面)**

### 4 this.$router.go(n)

**向前或者向后跳转n个页面，n可为正整数或负整数**

**7 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 Path 排特 params排儿门斯

vue-router的原理就是通过对URL地址变化的监听，继而对不同的组件进行渲染。每当URL地址改变时，就对相应的组件进行渲染。原理是很简单，实现方式可能有点复杂，主要有hash模式和history模式

**history kei 死特瑞**

**8 vue-router 有哪几种导航钩子？**

1、全局导航钩子：router.beforeEach(to,from,next)，作用：跳转前进行判断拦截。 next 耐四特

2、组件内的钩子；

3、单独路由独享组件

**8 Vue项目中实现token验证大致思路如下：**

1、第一次登录的时候，前端调后端的登陆接口，发送用户名和密码

2、后端收到请求，验证用户名和密码，验证成功，就给前端返回一个token

3、前端拿到token，将token存储到localStorage和vuex中，并跳转路由页面

4、前端每次跳转路由，就判断 localStroage 中有无 token ，没有就跳转到登录页面，有则跳转到对应路由页面

5、每次调后端接口，都要在请求头中加token

6、后端判断请求头中有无token，有token，就拿到token并验证token，验证成功就返回数据，验证失败（例如：token过期）就返回401，请求头中没有token也返回401

7、如果前端拿到状态码为401，就清除token信息并跳转到登录页面

localStorage 楼购 斯到瑞之

**9 如何实现权限管理**

在根组件created钩子里将路由定向到登录页，用户登录成功后前端拿到用户token，设置axios实例统一为请求headers添加{"Authorization":token}实现用户鉴权，然后获取当前用户的权限数据，主要包括路由权限和资源权限，之后动态添加路由，生成菜单，实现权限指令和全局权限验证方法，并为axios实例添加请求拦截器，至此完成权限控制初始化。动态加载路由后，路由组件将随之加载并渲染，而后展现前端界面。

Created 可睿的 headers海的 Axios 哎克车死 Authorization奥不睿贼神

**10 如何配置请求拦截器与响应拦截器**

### 先进行axios配置，封装到了api ，然后进行默认配置基准地址与请求头携带token

使用axios配置请求拦截器(在每次请求的头部加上一个token)，每次发送请求之前 执行拦截器，拦截成功，加token否则拦截失败 (成功的业务中出现报错)

使用axios配置响应拦截器（在每次响应失败的时候判断token是否失效）err 获取 状态码 token失效 清除存储token 重新登录 登录成功 跳转到登录页面

**11 如何完成实时请求**

了setInterval()定时调用ajax请求数据

通过vue-axios发送XHR请求，通过progress事件,我们可以据此在浏览器接收新数据期间添加实时数据 progress 普绕古ruai思

**12 如何实现数据驱动视图**

数据统计页面也由element-ui搭建，其中数据图表采用了echarts插件，将axios获取数据及echarts的配置项代码放入methods中的一个方法里，并用钩子函数created调用这个方法，即可点开页面展示图表。Axios 哎克车死 Echarts 义车死 methods买色斯 Created 哭睿忒的

**13 请说下封装 vue 组件的过程？**

首先，组件可以ᨀ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

Extend 爱克斯den的 component 看破嫩特 emit亿美特

**14 vue 中组件直接的通信是如何实现的？**

组件关系可分为父子组件通信、兄弟组件通信。

1、父组件向子组件： 通过 props 属性来实现

2、子组件向父组件：

子组件用$emit （）来触发事件，父组件用$on（）来监昕子组件的事件。

父组件可以直接在子组件的自定义标签上使用 v-on 来监昕子组件触发的自定义事件。

3、兄弟之间的通信：

通过实例一个 vue 实例 Bus 作为媒介，要相互通信的兄弟组件之中都引入 Bus，之后通过分别调用 Bus 事件触发和监听来实现组件之间的通信和参数传递。 Bus 八四

**15 vue双向数据绑定**

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。

数据驱动 是vue.js最大的特点。在vue.js中，所谓的数据驱动就是当数据发生变化的时候，用户界面发生相应的变化，开发者不需要手动的去修改dom。

Object.defineProperty 偶不摘可特 . 笛梵 普绕普特

setter （赛的） getter （该的）

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。 observe 额不则

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。 compile 康拍奥

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。 Watcher 我吃 update 啊不得特

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。

**16 前端性能优化**

（1） 减少http请求次数：CSS Sprites, JS、CSS源码压缩、图片大小控制合适；网页Gzip，CDN托管，data缓存 ，图片服务器。

（2） 前端模板 JS+数据，减少由于HTML标签导致的带宽浪费，前端用变量保存AJAX请求结果，每次操作本地变量，不用请求，减少请求次数

（3） 用innerHTML代替DOM操作，减少DOM操作次数，优化javascript性能。

（4） 当需要设置的样式很多时设置className而不是直接操作style。

（5） 少用全局变量、缓存DOM节点查找的结果。减少IO读取操作。

（6） 避免使用CSS Expression（css表达式)又称Dynamic properties(动态属性)。

（7） 图片预加载，将样式表放在顶部，将脚本放在底部 加上时间戳。

（8） 避免在页面的主体布局中使用table，table要等其中的内容完全下载之后才会显示出来，显示比div+css布局慢。

Style 死袋奥 inner A那 class块死 Name内幕 table tei bout

**17 DOM操作**

**Vue可以减少dom操作**

Vue通过Vue对象将数据和View完全分离开来了。对数据进行操作不再需要引用相应的DOM对象，可以说数据和View是分离的，他们通过Vue对象这个vm实现相互的绑定。

#### **18 VUE是什么?**

vue.js（读音 /vjuː/，类似于 view） 是一套构建用户界面的渐进式框架与其他重量级框架不同的是，Vue 采用自底向上增量开发的设计。Vue 的核心库只关注视图层，它不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与单文件组件和 Vue 生态系统支持的库结合使用时，Vue 也完全能够为复杂的单页应用程序提供驱动。

**19 Vue.js 核心思想：**

（1）、Dom 是数据的一种自然映射（双向数据绑定）手动改变 DOM 非常麻烦使用 vue.js

之后，只需要改变数据，通过改变 Directives 指令,当数据发生变化，会通过数据指令去修

改对应的 DOM

（2）、Vue.js 还会对事件进行一定的监听，当我们改变视图（view）的时候通过 DOM

Listeners 来改变数据

通过以上两点就实现了数据的双向绑定

2、组件化

Vue-cli:(它是 vue 的脚手架工具)

作用：帮助我们完成基础的代码（包括：目录结构、本地调试、代码部署、热加载、单元测

试）

组件的设计原则：

1 页面上每一个独立的可视/可交互区域视为一个组件

2 每个组件对应一个工程目录，组件所需要的各种资源在这个目录下就近维护

3 展示面不过是组件的容器，组件可以嵌套自由组合形成完整的页面

**一 前端相关**

**1模块化理解**

所谓的模块化开发就是封装细节，供使用接口，彼此之间互不影响，每个模块都是实现某一特定的功能。模块化开发的基础就是函数

**2 根据设备尺寸做页面自适应**

1. 页面中关乎布局的元素都不设置绝对的宽和高，都按百分比来设置。

2. 字体也使用相对大小的字体

3. 用div+css+浮动 来布局 各个盒子的位置用浮动的，不是固定不变的。

4. 尽量不要使用绝对定位，即position:absolute的定位。

5. 根据浏览器的版本做一些css的处理

7. css文件中，根据分辨率设置不同的css风格

8. 图片自动缩放，自适应大小

如：img{max-width:100%;}

设置meta标签

**3 px，em，rem 的区别？**

1）px 像素（Pixel）。绝对单位。像素 px 是相对于显示器屏幕分辨率而言的，是一个虚拟长 度单位，是计算机系统的数字化图像长度单位，如果 px 要换算成物理长度，需要指定精度 DPI。

2）em 是相对长度单位，相对于当前对象内文本的字体尺寸。如当前对行内文本的字体尺寸 未被人为设置，则相对于浏览器的默认字体尺寸。它会继承父级元素的字体大小，因此并不是一 个固定的值。

3）rem 是 CSS3 新增的一个相对单位（root em，根 em），使用 rem 为元素设定字体大小时， 仍然是相对大小，但相对的只是 HTML 根元素。

4）区别：IE 无法调整那些使用 px 作为单位的字体大小，而 em 和 rem 可以缩放，rem 相对的只是 HTML 根元素。这个单位可谓集相对大小和绝对大小的优点于一身，通过它既可以做到只修 改根元素就成比例地调整所有字体大小，又可以避免字体大小逐层复合的连锁反应。目前，除了IE8 及更早版本外，所有浏览器均已支持 rem。

**4 用 div+css 网站布局的好处**

1：表现和内容相分离

将设计部分剥离出来放在一个独立样式文件中，HTML 文件中只存放文本信息。

2：高搜索引擎对网页的索引效率

用只包含结构化内容的 HTML 代替嵌套的标签，搜索引擎将更有效地搜索到你的网页内容，并 可能给你一个较高的评价。

3：高页面浏览速度

对于同一个页面视觉效果，采用 CSS+DIV 重构的页面容量要比 TABLE 编码的页面文件容量小 得多，前者一般只有后者的1/2大小。

4：易于维护和改版

你只要简单的修改几个 CSS 文件就可以重新设计整个网站的页面。从以上的᧿述来看，采用 CSS+DIV 对网站重构可以大大ᨀ升网站用户与搜索引擎的友好度,CSS+DIV 所以成为目前网页布局 主流.

**5 Sass、LESS 是什么？大家为什么要使用他们？**

他们是 CSS 预处理器。他是 CSS 上的一种抽象层。他们是一种特殊的语法/语言编译成 CSS。

例如 Less 是一种动态样式语言. 将 CSS 赋予了动态语言的特性，如变量，继承，运算， 函数.

LESS 既可以在客户端上运行 (支持 IE 6+, Webkit, Firefox)，也可一在服务端运行 (借助 Node.js)。

为什么要使用它们？

结构清晰，便于扩展。

可以方便地屏蔽浏览器私有语法差异。这个不用多说，封装对浏览器语法差异的重复处理， 减少无意义的机械劳动。

可以轻松实现多重继承。

完全兼容 CSS 代码，可以方便地应用到老项目中。LESS 只是在 CSS 语法上做了扩展，所以老的 CSS 代码也可以与 LESS 代码一同编译。

**6 实现图片懒加载的原理**

先将img标签的src链接设为同一张图片（比如空白图片），然后给img标签设置自定义属性（比如 data-src）,然后将真正的图片地址存储在data-src中，当JS监听到该图片元素进入可视窗口时，将自定义属性中的地址存储到src属性中。达到懒加载的效果。这样做能防止页面一次性向服务器发送大量请求，导致服务器响应面，页面卡顿崩溃等。

**7 请描述一下 cookies，sessionStorage 和 localStorage 的区别？**

cookie是网站为了标示用户身份而储存在用户本地终端（Client Side）上的数据（通常经过加密）。

cookie数据始终在同源的http请求中携带（即使不需要），记会在浏览器和服务器间来回传递。

sessionStorage和localStorage不会自动把数据发给服务器，仅在本地保存。

存储大小：

cookie数据大小不能超过4k。

sessionStorage和localStorage 虽然也有存储大小的限制，但比cookie大得多，可以达到5M或更大。

有期时间：

localStorage 存储持久数据，浏览器关闭后数据不丢失除非主动删除数据；

sessionStorage 数据在当前浏览器窗口关闭后自动删除。

cookie 设置的cookie过期时间之前一直有效，即使窗口或浏览器关闭

**8 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？**

简洁版

1 浏览器根据请求的URL交给DNS域名解析，找到真实IP，向服务器发起请求；

2 服务器交给后台处理完成后返回数据，浏览器接收文件（HTML、JS、CSS、图象等）；

3 浏览器对加载到的资源（HTML、JS、CSS等）进行语法解析，建立相应的内部数据结构（如HTML的DOM）；

4 载入解析到的资源文件，渲染页面，完成。

详细版：

1、浏览器会开启一个线程来处理这个请求，对 URL 分析判断如果是 http 协议就按照 Web 方式来处理;

2、调用浏览器内核中的对应方法，比如 WebView 中的 loadUrl 方法;

3、通过DNS解析获取网址的IP地址，设置 UA 等信息发出第二个GET请求;

4、进行HTTP协议会话，客户端发送报头(请求报头);

5、进入到web服务器上的 Web Server，如 Apache、Tomcat、Node.JS 等服务器;

6、进入部署好的后端应用，如 PHP、Java、JavaScript、Python 等，找到对应的请求处理;

7、处理结束回馈报头，此处如果浏览器访问过，缓存上有对应资源，会与服务器最后修改时间对比，一致则返回304;

8、浏览器开始下载html文档(响应报头，状态码200)，同时使用缓存;

9、文档树建立，根据标记请求所需指定MIME类型的文件（比如css、js）,同时设置了cookie;

10、页面开始渲染DOM，JS根据DOM API操作DOM,执行事件绑定等，页面显示完成。

**二 HTML CSS**

**1 CSS3新特性**

1.颜色：新增 RGBA，HSLA 模式

2. 文字阴影（text-shadow、）

3.边框： 圆角（border-radius）边框阴影： box-shadow

4. 盒子模型：box-sizing

5.背景：background-size 设置背景图片的尺寸 background-origin 设置背景图片的原点 background-clip 设置背景图片的裁切区域，以”，”分隔可以设置多背景，用于自适应布局

6.渐变：linear-gradient、radial-gradient

7. 过渡：transition，可实现动画

8. 自定义动画 animate @**keyframe**  爱了门特 开飞木

9. 在 CSS3 中唯一引入的伪元素是 ：：selection.

10. 媒体查询，多栏布局 @media screen and (width:800px){ … }

11. border-image

12.2D 转换：transform：translate(x，y) rotate(x，y) skew(x，y) scale(x，y)

13. 3D 转换

14 字体图标 font-face

15 弹性布局 flex

## 2 CSS3 盒子模型(Box Model)

所有HTML元素可以看作盒子，在CSS中，"box model"这一术语是用来设计和布局时使用。

CSS盒模型本质上是一个盒子，封装周围的HTML元素，它包括：边距，边框，填充，和实际内容。

盒模型允许我们在其它元素和周围元素边框之间的空间放置元素。

**Margin(外边距)** - 清除边框外的区域，外边距是透明的。

**Border(边框)** - 围绕在内边距和内容外的边框。

**Padding(内边距)** - 清除内容周围的区域，内边距是透明的。

**Content(内容)** - 盒子的内容，显示文本和图像。

## 3 CSS3 弹性盒子(Flex Box)

CSS3 弹性盒（ Flexible Box 或 flexbox），是一种当页面需要适应不同的屏幕大小以及设备类型时确保元素拥有恰当的行为的布局方式。引入弹性盒布局模型的目的是提供一种更加有效的方式来对一个容器中的子元素进行排列、对齐和分配空白空间。

弹性盒子由弹性容器(Flex container)和弹性子元素(Flex item)组成。

弹性容器通过设置 display 属性的值为 flex 或 inline-flex将其定义为弹性容器。

弹性容器内包含了一个或多个弹性子元素。

**注意：** 弹性容器外及弹性子元素内是正常渲染的。弹性盒子只定义了弹性子元素如何在弹性容器内布局。

弹性子元素通常在弹性盒子内一行显示。默认情况每个容器只有一行。

**5 HTML5新特性**

新特性：

1. 拖拽释放(Drag and drop) API ondrop

自定义属性 data-id 获取 li.getAttribute('data-id')或者 li.dataset.type = 'guoji'

2. 语义化更好的内容标签（header,nav,footer,aside,article,section）

3. 音频、视频 API(audio,video) 如果浏览器不支持自动播放怎么办？

4. 画布(Canvas) API 热 canvas 和 image 的区别？

5. 地理(Geolocation) API

6. 本地离线存储 localStorage 长期存储数据，浏览器关闭后数据不丢失；

7. sessionStorage 的数据在浏览器关闭后自动删除

8. 表单控件，calendar、date、time、email、url、search 、tel、file、number

9. 新的技术 webworker, websocket, Geolocation

10．文件读取

移除的元素-纯表现的元素：basefont，big，center，font, s，strike，tt，u；

支持 HTML5 新标签：

\* IE8/IE7/IE6 支持通过 document.createElement 方法产生的标签，

可以利用这一特性让这些浏览器支持 HTML5 新标签，

**5 CSS 都有哪些选择器？CSS 选择器的优先级是怎么样定义的？**

important >行内样式> ID > 类 > 标签 | 伪类 | 属性选择 > 伪对象 > 继承 > 通配符。

**6 CSS选择符有哪些？哪些属性可以继承？**

\* 1.id选择器（ # myid）

2.类选择器（.myclassname）

3.标签选择器（div, h1, p）

4.相邻选择器（h1 + p）

5.子选择器（ul > li）

6.后代选择器（li a）

7.通配符选择器（ \* ）

8.属性选择器（a[rel = "external"]）

9.伪类选择器（a:hover, li:nth-child）

\* 可继承的样式： font-size font-family color, UL LI DL DD DT;

\* 不可继承的样式：border padding margin width height ;

#### **7** HTML常见兼容性问题

****1. png24位的图片在iE6浏览器上出现背景****

解决方案：做成PNG8，也可以引用一段脚本处理.

****2. 浏览器默认的margin和padding不同****

解决方案：加一个全局的 \*{margin:0;padding:0;} 来统一。

****3. IE6双边距bug：在IE6下，如果对元素设置了浮动，同时又设置了margin-left或margin-right，margin值会加倍。****

|  |  |
| --- | --- |
| 1 | #box{ **float**:left; **width**:10px; **margin**:0 0 0 10px;} |

这种情况之下IE会产生20px的距离

解决方案：在float的标签样式控制中加入 \_display:inline; 将其转化为行内属性。( \_ 这个符号只有ie6会识别)

****4. 渐进识别的方式，从总体中逐渐排除局部。****

首先，巧妙的使用“\9”这一标记，将IE游览器从所有情况中分离出来。

接着，再次使用 "+" 将IE8和IE7、IE6分离开来，这样IE8已经独立识别。

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | .bb{  **background-color**:#f1ee18; /\*所有识别\*/      .**background-color**:#00deff\9; /\*IE6、7、8识别\*/      +**background-color**:#a200ff; /\*IE6、7识别\*/  **\_background-color**:#1e0bd1; /\*IE6识别\*/  } |

****5.IE下，可以使用获取常规属性的方法来获取自定义属性，也可以使用 getAttribute() 获取自定义属性；Firefox下,只能使用getAttribute()获取自定义属性****

解决方法：统一通过getAttribute()获取自定义属性

****6. IE下，event对象有 x、y 属性，但是没有 pageX、pageY属性; Firefox下，event对象有 pageX、pageY 属性，但是没有 x、y 属性****

解决方法：（条件注释）缺点是在IE浏览器下可能会增加额外的HTTP请求数。

****7. Chrome 中文界面下默认会将小于 12px 的文本强制按照 12px 显示****

解决方法：可通过加入 CSS 属性 -webkit-text-size-adjust: none; 解决

****8. 超链接访问过后 hover 样式就不出现了，被点击访问过的超链接样式不在具有 hover 和 active 了****

解决方法：改变CSS属性的排列顺序 L-V-H-A

|  |  |
| --- | --- |
| 1  2  3  4 | a:link {}  a:visited {}  a:hover {}  a:active {} |

****9. 怪异模式问题：****漏写 DTD 声明，Firefox 仍然会按照标准模式来解析网页，但在 IE 中会触发怪异模式。为避免怪异模式给我们带来不必要的麻烦，最好养成书写 DTD 声明的好习惯。现在可以使用[html5](http://www.w3.org/TR/html5/single-page.html) 推荐的写法：<!DOCTYPE html>

****10. 上下margin重合问题：****ie和ff都存在，相邻的两个div的margin-left和margin-right不会重合，但是margin-top和margin-bottom却会发生重合。

解决方法：养成良好的代码编写习惯，同时采用margin-top或者同时采用margin-bottom。

****11. ie6对png图片格式支持不好****

解决方案：引用一段脚本处理

**8 清除浮动有几种方式?**

1. 额外标签法（在最后一个浮动标签后，新加一个标签，给其设置clear：both；）
2. 父级添加overflow属性（父元素添加overflow:hidden）（不推荐）

通过触发BFC方式，实现清除浮动

1. 使用after伪元素清除浮动（推荐使用）

4.使用before和after双伪元素清除浮动

**9 对 BFC 规范的理解有哪些？**

****什么是BFC？****

BFC 全称为 块级格式化上下文（Block Fromatting Context），是Web页面的可视化CSS渲染出的一部分。它是块级盒布局出现的区域，也是浮动层元素进行交互的区域。简单的来说，它就是一种会影响元素与元素之间的位置、间距的属性。

****BFC 的特性****

1.内部的box会在垂直方向，从顶部一个接一个的放置

2.box垂直方向的距离由margin决定，属于同一个BFC的两个相邻box的margin会发生叠加

3.每个元素的margin box的左边，与包含块border box的左边相接触

4.BFC的区域不会与float box叠加

5.BFC就是页面上的一个隔离的独立容器，容器里面的子元素不会影响到外面的元素，反之亦然

6.计算BFC的高度时，浮动元素也会参与计算

**10 XHTML 与 HTML 有什么区别？**

1. 所有的标记都必须要有一个相应的结束标记
2. 所有标签的元素和属性的名字都必须使用小写
3. 所有的 XML 标记都必须合理嵌套
4. 所有的属性必须用引号""括起来
5. 把所有<和&特殊符号用编码表示
6. 给所有属性赋一个值=""

不要在注释内容中使“--”=""

**11 定位 position的值relative和absolute定位原点是？**

absolute生成绝对定位的元素，相对于值不为 static的第一个父元素进行定位。

fixed （老IE不支持）生成绝对定位的元素，相对于浏览器窗口进行定位。

relative生成相对定位的元素，相对于其正常位置进行定位。

static默认值。没有定位，元素出现在正常的流中（忽略 top, bottom, left, right z-index 声明）。

Inherit 规定从父元素继承 position 属性的值。

**12 用纯CSS创建一个三角形的原理是什么？**

把上、左、右三条边隐藏掉（颜色设为 transparent）#demo {

width: 0;

height: 0;

border-width: 20px;

border-style: solid;

border-color: transparent transparent red transparent;}

**13、如何居中div?**

回答：（1）、水平居中：给DIV设置一个宽度，然后添加margin:0 auto属性。

div{

width:200px;

margin:0 auto;

}

（2）、让绝对定位的div居中：

div {

position: absolute;

width: 300px;

height: 300px;

margin: auto;

top: 0;

left: 0;

bottom: 0;

right: 0;

background-color: pink; /\* 方便看效果 \*/

}

（3）、水平垂直居中一：

确定容器的宽高 宽500 高 300 的层

设置层的外边距

div {

position: relative; /\* 相对定位或绝对定位均可 \*/

width:500px;

height:300px;

top: 50%;

left: 50%;

margin: -150px 0 0 -250px; /\* 外边距为自身宽高的一半 \*/

background-color: pink; /\* 方便看效果 \*/

}

（4）、水平垂直居中二：

未知容器的宽高，利用 `transform` 属性

div {

position: absolute; /\* 相对定位或绝对定位均可 \*/

width:500px;

height:300px;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

background-color: pink; /\* 方便看效果 \*/

}

（5）、水平垂直居中三：

利用 flex 布局

实际使用时应考虑兼容性

.container {

display: flex;

align-items: center; /\* 垂直居中 \*/

justify-content: center; /\* 水平居中 \*/

}

.container div {

width: 100px;

height: 100px;

background-color: pink; /\* 方便看效果 \*/

}

**三 JQUERY**

***1 给DOM元素绑定事件有哪些方法？***

a、在DOM元素中直接绑定；

b、在JavaScript代码中绑定；

c、绑定事件监听函数。

**四 JavaScript**

**1、new 操作符具体干了什么呢?**

1、创建一个空对象，并且 this 变量引用该对象，同时还继承了该函数的原型。

2、属性和方法被加入到 this 引用的对象中。

3、新创建的对象由 this 所引用，并且最后隐式的返回 this 。

**2、js 中深拷贝和浅拷贝的区别？**

**浅拷贝**只复制指向某个对象的指针，而不复制对象本身，新旧对象还是共享同一块内存。

**深拷贝**会另外创造一个一模一样的对象，新对象跟原对象不共享内存，修改新对象不会改到原 对象。

深拷贝的方法：1、递归拷贝

2、使用 Object.create()方法

3、jquery 有ᨀ供一个$.extend 也可以实现

4、函数库 lodash，也有ᨀ供 cloneDeep 用来实现

**3、什么是回调地狱，怎么解决回调地狱？**

由于回调函数是异步的，每一层的回调函数都需要依赖上一层的回调执行完，所以形成了

层层嵌套的关系最终形成了回调地狱。例如：定时器中再写定时器再写定时器，这种就形成了 通常所说的回调地狱。

解决 1、避免函数的嵌套 2、模块化开发 3、使用 Promise 解决

**4 谈谈你对 this 理解？**

普通函数中：this->window 定时器中：this->window 构造函数中：this->当前实例化的对象 事件处理函数中：this->事件触发对象 在 js 中一般理解就是谁调用这个 this 就指向谁

**5 、Js 的原型和原型链？原型链的应用？**

每个对象都会在其内部初始化一个属性，就是 prototype(原型)，当我们访问一个对象的属性时， 如果这个对象内部不存在这个属性，那么他就会去 prototype 里找这个属性，这个 prototype 又会 有自己的 prototype，于是就这样一直找下去，也就是我们平时所说的原型链的概念。

应用：

原型链是实现继承的主要方法。

**6 Javascript如何实现继承？**

1、构造继承2、原型继承3、实例继承4、拷贝继承

原型prototype机制或apply和call方法去实现较简单，建议使用构造函数与原型混合方式。

function Parent(){

this.name = 'wang';

}

function Child(){

this.age = 28;

}

Child.prototype = new Parent();//继承了Parent，通过原型

var demo = new Child();

alert(demo.age);

alert(demo.name);//得到被继承的属性

}

**构造函数继承**

**1 构造函数绑定 2** **prototype模式 3 直接继承prototype 普肉特太普**

**4 利用空对象作为中介 5 拷贝继承**

****非构造函数继承****

**1 object()方法 2 浅拷贝 3 深拷贝**

1、扩展原型对象实现继承

2、利用 apply(),和 call 实现继承 apply 额普来 call 看奥

3、组合 call+prototype，最常用的一种方式 prototype 剖头泰普

***7 闭包***

闭包是指有权访问另外一个函数作用域中的变量的函数.可以理解为(能够读取其他函数内部变量的函数)

闭包的作用: 正常函数执行完毕后,里面声明的变量被垃圾回收处理掉,但是闭包可以让作用域里的变量,在函数执行完之后依旧保持没有被垃圾回收处理掉  
**.闭包的缺陷**

闭包会导致内存占用过高,因为变量都没有释放内存

**闭包的用途:**

1.读取函数内部的变量

2.让变量始终保持在内存中，延长变量的生命周期

闭包形成原因？

**8 JS 的数据类型有哪些？**

简单数据类型：Undefined、Null、Boolean、Number 和 String。

复杂数据类型：Object

Undefined 安低饭的 Null nou Boolean偶链 Number南玻 String四坠影

**9 Js 的原型和原型链？原型链的应用？**

每个对象都会在其内部初始化一个属性，就是 prototype(原型)，当我们访问一个对象的属性时， 如果这个对象内部不存在这个属性，那么他就会去 prototype 里找这个属性，这个 prototype 又会 有自己的 prototype，于是就这样一直找下去，也就是我们平时所说的原型链的概念。

应用： 原型链是实现继承的主要方法。 prototype 剖头泰普

#### 10 什么是事件的冒泡和捕获？

事件冒泡：子元素事件的触发会影响父元素事件；

开关事件冒泡： 爱了门特 add 一问特 雷四那

A，开启事件冒泡：element.addEventListener(eventName，handler，false)；

B，关闭事件冒泡：假设传统方式事件的返回值为 e，就可以通过 e.stopPropagation()来关闭事件冒泡；

事件捕获：父元素的事件会影响子元素的事件；

开启事件捕获：element.addEventListener(eventName，handler，true)

**11 jsonp 的原理**

ajax 请求受同源策略影响，不允许进行跨域请求，而 script 标签 src 属性中的链接却可以访问跨域的 Js 脚本，利用这个特性，服务端不再返回 JSON 格式的数据，而是返回一段调用某个函数的 js 代码，在 src 中进行了调用，这样实现了跨域。 script四滚特

**12 跨域的几种解决方式**

**方案1** 通过 jsonp 跨域

JSONP是JSON with padding的简写。JSONP由两部分组成:回调函数和数据。回调函数是客户端和服务端约定好一个函数名，一般在请求中指定。数据是传入回调函数中的JSON数据。

**方案2***、CORS*

CORS 是跨域资源分享（Cross-Origin Resource Sharing）的缩写。它是 W3C 标准，属于跨源 AJAX 请求的根本解决方法。跨域资源共享定义了在必须访问跨域资源的时，浏览器与服务器应该如何沟通。他的原理是使用自定义的 HTTP 头部，让服务器与浏览器进行沟通，主要是通过设置响应头的 Access-Control-Allow-Origin 来达到目的的。

### **方案3-document.domain** 通过修改document.domain来跨子域

浏览器的同源策略使得不同域的框架是不能进行JS的交互操作的。比如：有一个页面是http://www.example.com/a.html,在这个页面中还有一个http://example.com/b.html，很显然，a.html与b.html是不同域的，所以我们无法通过在页面中书写js代码来获取iframe中的东西，但是，如果我们把这2个页面的document.domain都设置成相同的域名就可以了，需要注意的是，我们只能把document.domain设置成自身或更高一级的父域，且主域名必须相同。

### **方案4-window.name** 使用window.name来进行跨域；

window对象有个name属性，该属性有一个特征：即在一个窗口的生命周期内，窗口载入的所有页面都是共享一个window.name的，每个页面对window.name都有读写的权限，window.name是持久存在一个窗口载入过的所有页面中的。

### **方案5-window.postMessage**

window.postMessage（message,targetOrigin）方法，可以用来向其他的window对象发送消息，无论这个window对象是属于同一个源还是不同源

**方案六 使用HTML5新引进的window.postMessage方法来进行跨域（ie6 7不支持）；**

***请你谈谈Cookie的弊端***

a. 每个特定的域名下最多生成的cookie个数有限制

b. IE和Opera 会清理近期最少使用的cookie，Firefox会随机清理cookie

c. cookie的最大大约为4096字节，为了兼容性，一般不能超过4095字节

d. 安全性问题。如果cookie被人拦截了，那人就可以取得所有的session信息。

***13 .请解释一下 JavaScript 的同源策略***

同源策略指的是：协议，域名，端口相同，同源策略是一种安全协议。

指一段脚本只能读取来自同一来源的窗口和文档的属性。

***14.一次js请求一般情况下有哪些地方会有缓存处理？***

a. 浏览器端存储

b. 浏览器端文件缓存

c. HTTP缓存304

d. 服务器端文件类型缓存

e. 表现层&DOM缓存

15 ***、get、 post的区别***

get获取数据；post提交数据

get 用于获取信息，是无副作用的，是幂等的，且可缓存

post 用于修改服务器上的数据，有副作用，非幂等，不可缓存

get 传参方式是通过地址栏URL传递，能直接看到get传递的参数，post传参方式参数URL不可见，

get 把请求的数据在URL后通过？连接，通过&进行参数分割。

post 将从参数存放在HTTP的包体内

get 对传递的数据长度受URL的限制，URL最大长度是2048个字符。

post 没有长度限制

get后退不会有影响，post后退会重新进行提交

get请求可以被缓存，post不可以被缓存

get请求只支持URL编码，post支持多种编码方式

get请求的记录会留在历史记录中，post请求不会留在历史记录

get只支持ASCII字符，post没有字符类型限制

**五 AJAX**

**1 Ajax 是什么? 如何创建一个Ajax？**

Ajax 的原理简单来说通过 XmlHttpRequest 对象来向服务器发异步请求，从服务器获得数据，然后

用 javascript 来操作 DOM 而更新页面。这其中最关键的一步就是从服务器获得请求数据

ajax的全称：Asynchronous Javascript And XML。

异步传输+js+xml。

所谓异步，在这里简单地解释就是：向服务器发送请求的时候，我们不必等待结果，而是可以同时做其他的事情，等到有了结果它自己会根据设定进行后续操作，与此同时，页面是不会发生整页刷新的，提高了用户体验。

1. 创建XMLHttpRequest对象,也就是创建一个异步调用对象

(2)创建一个新的HTTP请求,并指定该HTTP请求的方法、URL及验证信息

(3)设置响应HTTP请求状态变化的函数

(4)发送HTTP请求

(5)获取异步调用返回的数据

(6)使用JavaScript和DOM实现局部刷新

**同步**：同步任务指的是，在主线程上排队执行的任务，只有前一个任务执行完毕，才能执行后一个任

务。

**异步**：不进入主线程、而进入"任务队列"（task queue）的任务，只有等主线程任务执行完毕，"任务队

列"开始通知主线程，请求执行任务，该任务才会进入主线程执行。

#### 2 简单描述下同步和异步的区别，并举例实际应用中哪些是同步，哪些是异步？

同步的概念应该是来自于OS中关于同步的概念:不同进程为协同完成某项工作而在先后次序上调整(通过阻塞,唤醒等方式).同步强调的是顺序性.谁先谁后.异步则不存在这种顺序性.

同步：浏览器访问服务器请求，用户看得到页面刷新，重新发请求,等请求完，页面刷新，新内容出现，用户看到新内容,进行下一步操作。

异步：浏览器访问服务器请求，用户正常操作，浏览器后端进行请求。等请求完，页面不刷新，新内容也会出现，用户看到新内容。

#### 3 简述AJAX的工作原理

Ajax的原理简单来说通过XmlHttpRequest对象来向服务器发异步请求，从服务器获得数据，然后用javascript来操作DOM而更新页面。这其中最关键的一步就是从服务器获得请求数据。

XMLHttpRequest是ajax的核心机制，它是在IE5中首先引入的，是一种支持异步请求的技术。简单的说，也就是javascript可以及时向服务器提出请求和处理响应，而不阻塞用户。达到无刷新的效果。 Request 睿块四特

**六 HTTP**

**1 请写出至少 5 种常见的 http 状态码以及代表的意义**

100 => 正在初始化（一般是看不到的）

101 => 正在切换协议（websocket浏览器提供的）

200或者以2开头的两位数 => 都是代表响应主体的内容已经成功返回了

202 => 表示接受

301 => 永久重定向/永久转移

302 => 临时重定向/临时转移（一般用来做服务器负载均衡）

304 => 本次获取的内容是读取缓存中的数据，会每次去服务器校验

400 => 参数出现错误（客户端传递给服务器端的参数出现错误）

401 => 未认证，没有登录网站

403 => 禁止访问，没有权限

404 => 客户端访问的地址不存在

500 => 未知的服务器错误

503 => 服务器超负荷（假设一台服务器只能承受10000人，当第10001人访问的时候，如果服务器没有做负载均衡，那么这个人的网络状态码就是503）

#### 2 你知道的HTTP 请求方式有几种？

**HTTPRequestMethod共计17种**

1. GET 请求指定的页面信息，并返回实体主体。

2. HEAD 类似于get请求，只不过返回的响应中没有具体的内容，用于获取报头

3. POST 向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST请求可能会导致新的资源的建立和/或已有资源的修改。

4. PUT 从客户端向服务器传送的数据取代指定的文档的内容。

5. DELETE 请求服务器删除指定的页面。

6. CONNECT HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。

7. OPTIONS 允许客户端查看服务器的性能。

8. TRACE 回显服务器收到的请求，主要用于测试或诊断。

9. PATCH 实体中包含一个表，表中说明与该URI所表示的原内容的区别。

10. MOVE 请求服务器将指定的页面移至另一个网络地址。

11. COPY 请求服务器将指定的页面拷贝至另一个网络地址。

12. LINK 请求服务器建立链接关系。

13. UNLINK 断开链接关系。

14. WRAPPED 允许客户端发送经过封装的请求。

15. LOCK 允许用户锁定资源，比如可以再编辑某个资源时将其锁定，以防别人同时对其进行编辑。

16. MKCOL 允许用户创建资源

17. Extension-mothed 在不改动协议的前提下，可增加另外的方法。

***3 Http与Https的区别***

HTTP 的URL 以http:// 开头，而HTTPS 的URL 以https:// 开头

HTTP 是不安全的，而 HTTPS 是安全的

HTTP 标准端口是80 ，而 HTTPS 的标准端口是443

在OSI 网络模型中，HTTP工作于应用层，而HTTPS 的安全传输机制工作在传输层

HTTP 无法加密，而HTTPS 对传输的数据进行加密

HTTP无需证书，而HTTPS 需要CA机构wosign的颁发的

**七 ES6**

1. **ES6新特性**

1、**Let**：声明的变量，只作用于使用了 let 命令的代码块 莱特

2、**const :**声明一个常量，声明过后，就不可修改其值(会报错) 康死特

3、解构性赋值

3、**箭头函数**：用法的两个明显的优点： \* 函数的写法更加简洁 \*

箭头函数内部没有自己的 this 对象，而是全部继承外面的，所以内部的 this 就是外层代码块的 this。

4、**字符串模板**：ES6 中允许使用反引号 ` 来创建字符串，此种方法创建的字符串里面可以包含由

美元符号加花括号包裹的变量${vraible}

5、**Proxy**：Proxy 可以监听对象身上发生了什么事情，并在这些事情发生后执行一些相应的操作 普rua SEI

6、**循环**：for of 实现数组的遍历（以及 keys(),value(),entries()等方法）

***2 箭头函数和普通函数有什么区别***

* 函数体内的this对象，就是定义时所在的对象，而不是使用时所在的对象，用call applybind也不能改变this指向
* 不可以当作构造函数，也就是说，不可以使用new命令，否则会抛出一个错误。
* 不可以使用arguments对象，该对象在函数体内不存在。如果要用，可以用 rest 参数代替。
* 不可以使用yield命令，因此箭头函数不能用作 Generator 函数。
* 箭头函数没有原型对象prototype

**八 VUE**

**1 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：

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结构依然存在复制代码

beforeCreate: 组件实例化之前执行的函数。

created:组件实例化完毕，但页面未显示。

beforeMount:组件挂载前，页面仍未显示，但虚拟Dom已经配置。

mounted:组件挂载后，此方法执行后，页面显示。

beforeUpdate:组件更新前，页面仍未更新，但虚拟Dom已经配置。

updated:组件更新，此方法执行后，页面显示。

beforeDestroy:组件销毁前。

destroyedl:组件销毁。

beforeCreate：在new一个vue实例后，只有一些默认的生命周期钩子和默认事件，其他的东西都还没创建。在beforeCreate生命周期执行的时候，data和methods中的数据都还没有初始化。不能在这个阶段使用data中的数据和methods中的方法

create：data 和 methods都已经被初始化好了，如果要调用 methods 中的方法，或者操作 data 中的数据，最早可以在这个阶段中操作

beforeMount：执行到这个钩子的时候，在内存中已经编译好了模板了，但是还没有挂载到页面中，此时，页面还是旧的

mounted：执行到这个钩子的时候，就表示Vue实例已经初始化完成了。此时组件脱离了创建阶段，进入到了运行阶段。 如果我们想要通过插件操作页面上的DOM节点，最早可以在和这个阶段中进行

beforeUpdate： 当执行这个钩子时，页面中的显示的数据还是旧的，data中的数据是更新后的， 页面还没有和最新的数据保持同步

updated：页面显示的数据和data中的数据已经保持同步了，都是最新的

beforeDestory：Vue实例从运行阶段进入到了销毁阶段，这个时候上所有的 data 和 methods ， 指令， 过滤器 ……都是处于可用状态。还没有真正被销毁

destroyed： 这个时候上所有的 data 和 methods ， 指令， 过滤器 ……都是处于不可用状态。组件已经被销毁了。

**2、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； 爱色斯

components 是放组件； ten剖嫩斯

router 是定义路由相关的配置;

view 视图； 位柔

app.vue 是一个应用主组件；

main.js 是入口文件

**3 什么是MVVM？**  
答：MVVM是是Model-View-ViewModel的缩写，Model代表数据模型，定义数据操作的业务逻辑，View代表视图层，负责将数据模型渲染到页面上，ViewModel通过双向绑定把View和Model进行同步交互，不需要手动操作DOM的一种设计思想

**4 说下你对 mvvm 的理解？双向绑定的理解？**

mvvm 就是 vm 框架视图、m 模型就是用来定义驱动的数据、v 经过数据改变后的 html、vm 就是用来实现双 向绑定

双向绑定:一个变了另外一个跟着变了，例如：视图一个绑定了模型的节点有变化，模型对应的值会跟着变

**5 你是怎么认识 vuex 的？**

1、vuex 可以理解为一种开发模式或框架。比如 PHP 有 thinkphp，java 有 spring 等。

2、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）。

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

***6 vuex的属性***

state：声明定义数据

getters：存放有依赖关系的数据，类似于computed计算属性

mutation：同步的修改state中的数据

actions：异步修改数据

modules：让每一个模块都有自己的state，getters，mutation，actions

***7、vuex原理***

vuex的原理其实非常简单，它为什么能实现所有的组件共享同一份数据？

因为vuex生成了一个store实例，并且把这个实例挂在了所有的组件上，所有的组件引用的都是同一个store实例。

store实例上有数据，有方法，方法改变的都是store实例上的数据。由于其他组件引用的是同样的实例，所以一个组件改变了store上的数据，导致另一个组件上的数据也会改变，就像是一个对象的引用。

**8 Vue请求数据的方式:**

Vue-resource,Axios,fetch三种方式.

Vue-resource方式:vue自身不带处理http请求,如果要使用http请求,必须先要引入vue-resource.js库,它可以通过XMLHttpRequest发起请求并处理响应.类似于jQuery

Axios的方式:Axios是第三方插件,不仅能在vue中使用,还能在其他第三方库中使用,例如react

**9 MVVM思想:Vue.js是一套构建用户界面的MVVM框架**

MVVM分为三部分:分别是M(model,模型层),V(View,视图层),VM(ViewModel,V与M连接的桥梁,也可以看做为控制器MVC的C层)

1. M:模型层,主要负责业务数据相关
2. V:视图层,负责视图相关,主要是HTML+CSS
3. VM:V与M沟通的桥梁,负责监听M或V的修改,是实现MVVM双向绑定的要点;因此开发者只需要关注业务逻辑,不需要手动操作DOM,不需要关注数据状态的同步问题,复杂的数据状态维护完全由MVVM来统一管理.

MVP思想:MVP的全称为Model-View-Presenter,Model提供数据,View负责显示,Presenter负责逻辑的处理

MVP与MVC的最大的区别:在MVP中View并不直接使用Model,它们之间的通信数据是通过Presenter来进行的,而在MVc中View会直接从Model中读取数据而不是通过Controller

Vue数据驱动:通过控制数据的变化来显示Vue的数据驱动，是视图的内容随着数据的改变而改变

**10、使用axios？**

vue是虚拟DOM操作的，JQuery.ajax和都需要操作DOM，axios本身就可以解决回调地狱的问题

**11、真实DOM和虚拟DOM的区别**

虚拟DOM不会进行排版与重绘操作

真实DOM频繁排版与重绘的效率是相当低

虚拟DOM进行频繁修改，然后一次性比较并修改真实DOM中需要改的部分，最后并在真实DOM中进行排版与重绘，减少过多DOM节点排版与重绘损耗

虚拟DOM有效降低的重绘与排版，因为最终与真实DOM比较差异，可以只渲染局部

**12 什么是 Vant**

[Vant ( ˈvænt )](https://github.com/youzan/vant" \t "https://segmentfault.com/a/_blank) 是有赞前端团队基于有赞统一的规范实现的 Vue 组件库，提供了一整套 UI 基础组件和业务组件。通过 Vant，可以快速搭建出风格统一的页面，提升开发效率。目前已有近50个组件，这些组件被广泛使用于有赞的各个移动端业务中。我们会在此基础上不断完善，开发更多实用的组件。

**13 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。

**14 css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

**15 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，<smith-button> </smith-button>

问题有：smithButton命名，使用的时候则smith-button。

## **16 Vue与Angular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： 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；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。

1. 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Virtual DOM。

**17 vue常用的修饰符？**

答：.prevent: 提交事件不再重载页面；.stop: 阻止单击事件冒泡；.self: 当事件发生在该元素本身而不是子元素的时候会触发；.capture: 事件侦听，事件发生的时候会调用

**九 Webpack**

**1 你对 Webpack 的认识？**

webpack 是收把项目当作一个整体，通过一个给定的的主文件，webpack 将从这个文件开始找到你 的项目的所有依赖文件，使用 loaders 处理它们，最后打包成一个或多个浏览器可识别的 js 文件

#### **2** webpack是怎样配置的，简要描述一下过程

第一在项目文件创建一个webpack.config.js文件，配置文件创建好了，第二开始正式配置webpack了，首先下载好node.js，因为webpack基于node.js,装好node.js后，通过命令行窗口找到项目文件，在项目文件webpack.config.js文件所在的目录下输入 npm install webpack -save dev 下载webpack依赖文件到本地项目中，下载好后会在webpack.config.js文件下自动创建node\_modules文件夹，文件夹里就是所有项目中用到的依赖插件，现在只有一个webpack，项目中用到在下载！

**十 算法**

**1 经典编程算法**

1、快速排序算法

2、堆排序算法

3、归并排序

4、二分查找算法

5、BFPRT（线性查找算法）

6、DFS（深度优先搜索）

7、BFS（广度优先搜索）

8、Floyd-Warshall all-pairs 最短路径算法

**2、经典排序算法**

1、插入排序—直接插入排序(Straight Insertion Sort)

2、插入排序—希尔排序（Shell`s Sort）

3、选择排序—简单选择排序（Simple Selection Sort）

4、选择排序—堆排序（Heap Sort）

5、交换排序—冒泡排序（Bubble Sort）

6、快速排序（Quick Sort）

**十一 Bootstrap**

**1 响应式开发：**

那么 Ethan Marcotte 在 2010 年 5 月份提出的一个概念，简而言之，就是一个网站能够兼容多 个终端。越来越多的设计师也采用了这种设计。

CSS3 中的 Media Query（媒介查询），通过查询 screen 的宽度来指定某个宽度区间的网页布局。

² 超小屏幕（移动设备） 768px 以下

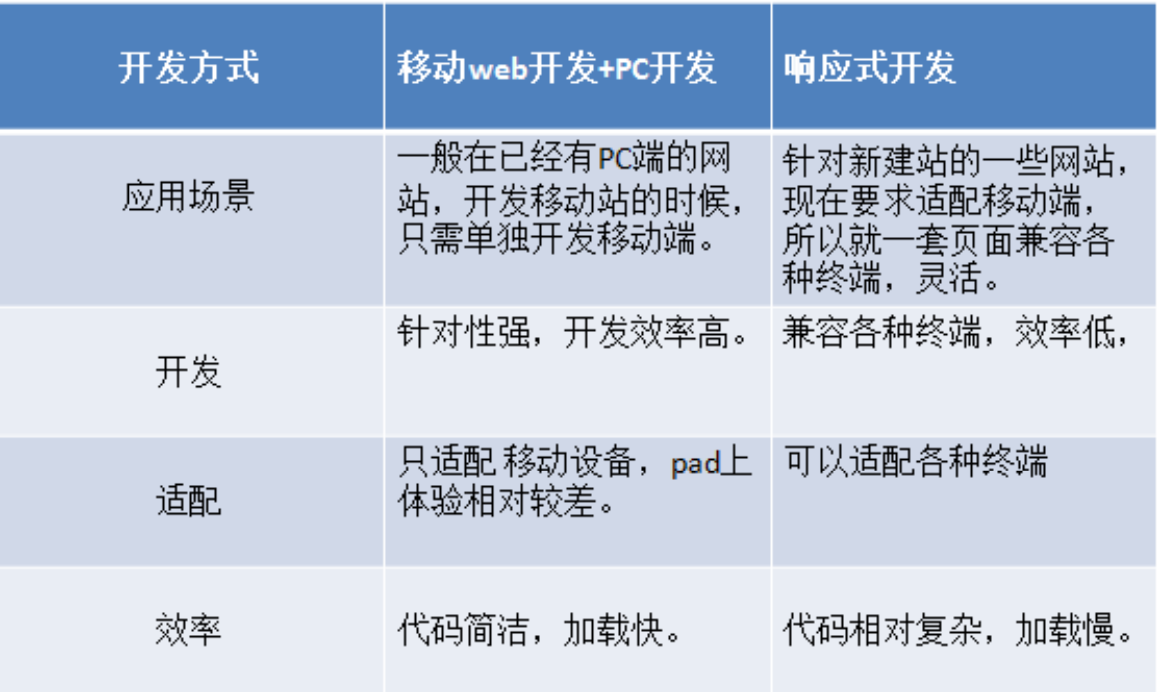
² 小屏设备 768px-992px

² 中等屏幕 992px-1200px

² 宽屏设备 1200px 以上

由于响应式开发显得繁琐些，一般使用第三方响应式框架来完成，比如 bootstrap 来完成一部分工作，当然也可以自己写响应式。

阐述下移动 web 和响应式的区别：



**2 Bootstrap 有什么好处，为什么要用 bootstrap，什么情况用比较合适**

Bootstrap 是一个用于快速开发 Web 应用程序和网站的前端框架。Bootstrap 是基于 HTML、CSS、

JAVASCRIPT 的。Bootstrap 是由 Twitter 的 Mark Otto 和 Jacob Thornton 开发的。Bootstrap 是

2011 年八月在 GitHub 上发布的开源产品。

为什么使用 Bootstrap？

移动设备优先：自 Bootstrap 3 起，框架包含了贯穿于整个库的移动设备优先的样式。

浏览器支持：所有的主流浏览器都支持 Bootstrap。

容易上手：只要您具备 HTML 和 CSS 的基础知识，您就可以开始学习 Bootstrap。

响应式设计：Bootstrap 的响应式 CSS 能够自适应于台式机、平板电脑和手机。它为开发人员创

建接口ᨀ供了一个简洁统一的解决方案。

它包含了功能强大的内置组件，易于定制。

它还ᨀ供了基于 Web 的定制。

它是开源的。

**3、什么是 Bootstrap 网格系统（Grid System）？**

Bootstrap 包含了一个响应式的、移动设备优先的、不固定的网格系统，可以随着设备或视口大小

的增加而适当地扩展到 12 列。它包含了用于简单的布局选项的预定义类，也包含了用于生成更

多语义布局的功能强大的混合类。

响应式网格系统随着屏幕或视口（viewport）尺寸的增加，系统会自动分为最多 12 列。

**4、Bootstrap 网格系统（Grid System）的工作原理？**

1、行必须放置在 .container class 内，以便获得适当的对齐（alignment）和内边距（padding）。

2、使用行来创建列的水平组。

3、内容应该放置在列内，且唯有列可以是行的直接子元素。

4、预定义的网格类，比如 .row 和 .col-xs-4，可用于快速创建网格布局。LESS 混合类可用于更多语

义布局。

5、列通过内边距（padding）来创建列内容之间的间隙。该内边距是通过 .rows 上的外边距（margin）

取负，表示第一列和最后一列的行偏移。

6、网格系统是通过指定您想要横跨的十二个可用的列来创建的。例如，要创建三个相等的列，则

使用三个 .col-xs-4

**十二 GIT与SVN**

#### 1 git的作用是什么，有哪些命令，这些命令的作用是什么？在pull时，如果只想要文件的一部分而不是工程文件的整个，使用什么命令？git的冲突如何解决？

git是一个开源的分布式版本控制系统，用以有效、高速处理很小到非常大的项目版本管理。

常用的命令：

git init 初始化仓库 一内特

git config --globaluser.name 配置用户名

git config --globaluser.email 配置邮箱

git add 文件添加到暂存区 add

git commit 文件添加到仓库 可美特

git branch 列出所有的分支 不润吃

git status 显示有变更的文件

git log 查看当前分支的版本历史

git push 提交本地代码到远程

git pull 从远程仓库下载到本地

git checkout 创建切换分支

解决git分支冲突： 将本地dev删除,在重新checkout一个dev分支(保证了此时我们的本地dev分支是最新的),在进行pull服务器分支,就这样解决了.

**十三 程序题**

**1 如何将字符串转化为数字，例如’12.3b’?**

\* parseFloat('12.3b');

\* 正则表达式，'12.3b'.match(/(\d)+(\.)?(\d)+/g)[0] \* 1, 但是这个不太靠谱，提供一种思路而已。

**2 如何将浮点数点左边的数每三位添加一个逗号，如12000000.11转化为『12,000,000.11』?**

function commafy(num){

return num && num

.toString()

.replace(/(\d)(?=(\d{3})+\.)/g, function($1, $2){

return $2 + ',';

});

}

**3 如何实现数组的随机排序？**

方法一：

var arr = [1,2,3,4,5,6,7,8,9,10];

function randSort1(arr){

for(var i = 0,len = arr.length;i < len; i++ ){

var rand = parseInt(Math.random()\*len);

var temp = arr[rand];

arr[rand] = arr;

arr = temp;

}

return arr;

}

console.log(randSort1(arr));

方法二：

var arr = [1,2,3,4,5,6,7,8,9,10];

function randSort2(arr){

var mixedArray = [];

while(arr.length > 0){

var randomIndex = parseInt(Math.random()\*arr.length);

mixedArray.push(arr[randomIndex]);

arr.splice(randomIndex, 1);

}

return mixedArray;

}

console.log(randSort2(arr));

方法三：

var arr = [1,2,3,4,5,6,7,8,9,10];

arr.sort(function(){

return Math.random() - 0.5;

})

console.log(arr);

### **4  js实现数组去重怎么实现?**

方法1. 创建一个新的临时数组来保存数组中已有的元素

var a = new Array(1,2,2,2,2,5,3,2,9,5,6,3);

Array.prototype.unique1 = function(){

var n = []; //一个新的临时数组

for(var i=0; i<this.length; i++){

//如果把当前数组的第i已经保存进了临时数组, 那么跳过

if(n.indexOf(this[i]) == -1){

n.push(this[i]);

}

}

return n;

}

console.log(a.unique1());

方法2. 使用哈希表存储已有的元素

Array.prototype.unique2 = function(){

var hash = {},

n = []; //hash 作为哈希表, n为临时数组

for(var i=0; i<this.length; i++){

if(!hash[this[i]]){ //如果hash表中没有当前项

hash[this[i]] = true; //存入hash表

n.push(this[i]); //当前元素push到临时数组中

}

}

return n;

}

**5 js实现简单的双向绑定**

**<**body>

<div id="app">

<input type="text" id="txt">

<p id="show"></p>

</div>

</body>

<script type="text/javascript">

var obj = {}

Object.defineProperty(obj, 'txt', {

get: function () {

return obj

},

set: function (newValue) {

document.getElementById('txt').value = newValue

document.getElementById('show').innerHTML = newValue

}

})

document.addEventListener('keyup', function (e) {

obj.txt = e.target.value

})

</script>

**十四 其他框架**

**1 什么是RequireJS**

RequireJS是一个JavaScript文件和模块加载器，可视为模块管理工具。