前端面试题整理

阅读 6012 收藏 377 2017-03-13 原文链接: www.jianshu.com

腾讯云助力开发者快速搭建小程序,一站式配置主机和域名,共享10亿客户,开发套餐11元/月起。cloud.tencent.com

原文链接 blog.poetries.top/2017/03/12/...

目录

\$HTML, HTTP, web综合问题

- 1、前端需要注意哪些 SEO
- 2、 的 title 和 alt 有什么区别
- 3、 HTTP 的几种请求方法用途
- 4、从浏览器地址栏输入 url 到显示页面的步骤
- 5、如何进行网站性能优化
- 6、HTTP状态码及其含义
- 7、语义化的理解
- 8、介绍一下你对浏览器内核的理解
- 9、 html5 有哪些新特性、移除了那些元素?
- 10、 HTML5 的离线储存怎么使用,工作原理能不能解释一下?
- 11、浏览器是怎么对 HTML5 的离线储存资源进行管理和加载的呢
- 12、请描述一下 cookies , sessionStorage 和 localStorage 的区别
- 13、 **iframe** 有那些缺点?
- 14、WEB 标准以及W3C标准是什么?
- 15、 xhtml 和 html 有什么区别?
- 16、 Doctype 作用? 严格模式与混杂模式如何区分? 它们有何意义?
- 17、行内元素有哪些? 块级元素有哪些? 空(void)元素有那些? 行内元素和块级元素有什么区别?
- 18、HTML 全局属性(global attribute)有哪些







- 2、display: none; 与 visibility: hidden;的区别
- 3、link与@import的区别
- 4、什么是 FOUC? 如何避免
- 5、如何创建块级格式化上下文(block formatting context), BFC 有什么用
- 7、清除浮动的几种方式,各自的优缺点
- 8、为什么要初始化 css 样式?
- 9、 css3 有哪些新特性
- 10、 display 有哪些值? 说明他们的作用
- 11、介绍一下标准的CSS的盒子模型? 低版本IE的盒子模型有什么不同的?
- 12、css 优先级算法如何计算?
- 13、对 BFC 规范的理解?
- 14、谈谈浮动和清除浮动
- 15、 position 的值, relative 和absolute 定位原点是
- 16、 display:inline-block 什么时候不会显示间隙? (携程)
- 17、 PNG, GIF, JPG 的区别及如何选

\$JavaScript

- 1、闭包
- 2、说说你对作用域链的理解
- 3、JavaScript 原型,原型链?有什么特点?
- 4、请解释什么是事件代理
- 5、 Javascript 如何实现继承?
- 6、谈谈 This 对象的理解
- 7、事件模型
- 8、 new 操作符具体干了什么呢?
- 9、Ajax 原理
- 11、模块化开发怎么做?
- 12、异步加载 JS 的方式有哪些?
- 13、那些操作会造成内存泄漏?
- 14、 XML 和 JSON 的区别?
- 15、谈谈你对 webpack 的看法
- 17、常见 web 安全及防护原理
- 18、用过哪些设计模式?
- 19、为什么要有同源限制?
- 20、 offsetWidth/offsetHeight, clientWidth/clientHeight与 scrollWidth/scrollHeight 的区
- 😻 掘金 一个帮助开发者成长的社区



※ 掘金 前▼

登录·注册

- 23、你觉得 jQuery 源码有哪些与的好的地方
- 25、 Node 的应用场景
- 26、谈谈你对 AMD 、 CMD 的理解
- 27、那些操作会造成内存泄漏?
- 28、web 开发中会话跟踪的方法有哪些
- 29、介绍 js 的基本数据类型
- 30、介绍 js 有哪些内置对象?
- 31、说几条写 JavaScript 的基本规范?
- 32、 JavaScript 有几种类型的值? , 你能画一下他们的内存图吗?
- 33、 javascript 创建对象的几种方式?
- 34、 eval 是做什么的?
- 35、null, undefined 的区别?
- 36、["1", "2", "3"].map(parseInt) 答案是多少?
- 37、 javascript 代码中的 "use strict";是什么意思?使用它区别是什么?
- 38、JSON 的了解?
- 39、js延迟加载的方式有哪些?
- 40、同步和异步的区别?
- 41、渐进增强和优雅降级
- 42、 defer 和 async
- 43、说说严格模式的限制
- 44、 attribute 和 property 的区别是什么?
- 45、 谈谈你对 ES6 的理解

\$编程题

- 1、写一个通用的事件侦听器函数
- 2、如何判断一个对象是否为数组
- 3、冒泡排序
- 4、快速排序
- 5、编写一个方法 求一个字符串的字节长度

\$其他

- 1、谈谈你对重构的理解
- 2、什么样的前端代码是好的
- 3、对前端工程师这个职位是怎么样理解的?它的前景会怎么样?
- . 1 次举组益澳工组的总法从现在哪
- 掃金 一个帮助开发者成长的社区



- 你有什么爱好?
- 你最大的优点和缺点是什么?
- 你为什么会选择这个行业, 职位?
- 你觉得你适合从事这个岗位吗?
- 你有什么职业规划?
- 你对工资有什么要求?
- 如何看待前端开发?
- 未来三到五年的规划是怎样的?

一些问题

- 你的项目中技术难点是什么?
- 遇到了什么问题?
- 你是怎么解决的?
- 最近在看哪些前端方面的书?
- 平时是如何学习前端开发的?
- 为什么大型应用需要从多个域名请求资源?
- 什么样的页面具有良好的用户体验?
- 是否了解 Web 注入攻击,说下原理,最-常见的两种攻击(XSS 和 CSRF)了解到什么程度

\$HTML, HTTP, web综合问题

1、前端需要注意哪些SEO

- 合理的 title 、 description 、 keywords : 搜索对着三项的权重逐个减小, title值强调重点即可, 重要关键词出现不要超过2次, 而且要靠前, 不同页面 title 要有所不同; description 把页面内容高度概括, 长度合适, 不可过分堆砌关键词, 不同页面 description 有所不同; keywords 列举出重要关键词即可
- 语义化的 HTML 代码,符合W3C规范:语义化代码让搜索引擎容易理解网页
- 重要内容 HTML 代码放在最前:搜索引擎抓取 HTML 顺序是从上到下,有的搜索引擎对抓取长度有限制,保证重要内容一定会被抓取
- 重要内容不要用 js 输出: 爬虫不会执行js获取内容
- 少用 iframe: 搜索引擎不会抓取 iframe 中的内容
- 非装饰性图片必须加 alt
- 提高网站速度: 网站速度是搜索引擎排序的一个重要指标







• alt 是 的特有属性,是图片内容的等价描述,用于图片无法加载时显示、读屏器阅读图片。可提图片高可访问性,除了纯装饰图片外都必须设置有意义的值,搜索引擎会重点分析。

3、HTTP的几种请求方法用途

- 1、GET 方法
 - 。 发送一个请求来取得服务器上的某一资源
- 2、POST 方法
 - 。 向 URL 指定的资源提交数据或附加新的数据
- 3、PUT 方法
 - 。 跟 POST 方法很像,也是想服务器提交数据。但是,它们之间有不同。PUT指定了资源在服务器上的位置,而 POST 没有
- 4、 HEAD 方法
 - 。 只请求页面的首部
- 5、DELETE 方法
 - 。 删除服务器上的某资源
- 6、OPTIONS 方法
 - 。它用于获取当前 URL 所支持的方法。如果请求成功,会有一个 Allow 的头包含类似 "GET, POST" 这样的信息
- 7、TRACE 方法
 - 。 TRACE 方法被用于激发一个远程的,应用层的请求消息回路
- 8、CONNECT 方法
 - 。 把请求连接转换到诱明的 TCP/IP 诵道
- 🧼 掘金 一个帮助开发者成长的社区



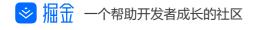
※ 掘金 首页▼

登录·注册

- 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);
- 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);
- 载入解析到的资源文件, 渲染页面, 完成。

5、如何进行网站性能优化

- content 方面
 - 1. 减少 HTTP 请求: 合并文件、 CSS 精灵、 inline Image
 - 2. 减少 DNS 查询: DNS 缓存、将资源分布到恰当数量的主机名
 - 3. 减少 DOM 元素数量
- Server 方面
 - 1. 使用 CDN
 - 2. 配置 ETag
 - 3. 对组件使用 Gzip 压缩
- Cookie 方面
 - 1. 减小 cookie 大小
- css 方面
 - 1. 将样式表放到页面顶部
 - 2. 不使用 CSS 表达式
 - 3. 使用 不使用 @import
- Javascript 方面
 - 1. 将脚本放到页面底部
 - 2. 将 javascript 和 css 从外部引入
 - 3. 压缩 javascript 和 css
 - 4. 删除不需要的脚本
 - 5. 减少 DOM 访问









3. 小要仕 HTML 中拉伸图片

6、HTTP状态码及其含义

- **1XX** : 信息状态码
 - 。 100 Continue 继续,一般在发送 post 请求时,已发送了 http header 之后服务端将返回此 信息,表示确认,之后发送具体参数信息
- 2XX: 成功状态码
 - 200 OK 正常返回信息
 - 。 201 Created 请求成功并且服务器创建了新的资源
 - 。 202 Accepted 服务器已接受请求,但尚未处理
- 3XX: 重定向
 - 。 301 Moved Permanently 请求的网页已永久移动到新位置。
 - o 302 Found 临时性重定向。
 - o 303 See Other 临时性重定向,且总是使用 GET 请求新的 URI。
 - 。 304 Not Modified 自从上次请求后,请求的网页未修改过。
- 4XX: 客户端错误
 - 。 400 Bad Request 服务器无法理解请求的格式,客户端不应当尝试再次使用相同的内容发起 请求。
 - 401 Unauthorized 请求未授权。
 - o 403 Forbidden 禁止访问。
 - 404 Not Found 找不到如何与 URI 相匹配的资源。
- 5XX: 服务器错误
 - 500 Internal Server Error 最常见的服务器端错误。
 - 。 503 Service Unavailable 服务器端暂时无法处理请求(可能是过载或维护)。

7、语义化的理解

- 用正确的标签做正确的事情!
- html 语义化就是让页面的内容结构化,便于对浏览器、搜索引擎解析;
- 在没有样式 CSS 情况下也以一种文档格式显示,并且是容易阅读的。
- 搜索引擎的爬虫依赖于标记来确定上下文和各个关键字的权重, 利于 SEO 。
- 使阅读源代码的人对网站更容易将网站分块,便于阅读维护理解

8、介绍一下你对浏览器内核的理解?



😺 掘缶 一个帮助开发者成长的社区



解释会有个问,所以渲染的效果也个相问。所有网页浏览器、电子邮件各户端以及其它需要编辑、显示网络内容的应用程序都需要内核

- JS 引擎则:解析和执行 javascript 来实现网页的动态效果
- 最开始渲染引擎和 JS 引擎并没有区分的很明确,后来JS引擎越来越独立,内核就倾向于只指渲染引擎

9、html5有哪些新特性、移除了那些元素?

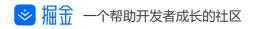
- HTML5 现在已经不是 SGML 的子集,主要是关于图像,位置,存储,多任务等功能的增加
 - o 绘画 canvas
 - 。 用于媒介回放的 video 和 audio 元素
 - 本地离线存储 localStorage 长期存储数据,浏览器关闭后数据不丢失
 - 。 sessionStorage 的数据在浏览器关闭后自动删除
 - 。 语意化更好的内容元素,比如 article 、 footer 、 header 、 nav 、 section
 - 。 表单控件, calendar 、 date 、 time 、 email 、 url 、 search
 - 。 新的技术 webworker, websocket, Geolocation

• 移除的元素:

- 。 纯表现的元素: basefont , big , center , font , s , strike, tt, u`
- 。 对可用性产生负面影响的元素: frame , frameset , noframes
- 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
 - 。 浏览器支持新标签后,还需要添加标签默认的样式
- 当然也可以直接使用成熟的框架、比如 html5shim

10、 HTML5 的离线储存怎么使用,工作原理能不能解释一下?

在用户没有与因特网连接时,可以正常访问站点或应用,在用户与因特网连接时,更新用户机器 上的缓存文件







- י און ערויין ער יי
 - 。 页面头部像下面一样加入一个 manifest 的属性;
 - 。 在 cache.manifest 文件的编写离线存储的资源
 - 。 在离线状态时,操作 window.applicationCache 进行需求实现

CACHE MANIFEST

#v0.11

CACHE:

js/app.js

css/style.css

NETWORK:

resourse/logo.png

FALLBACK:

/ /offline.html

11、浏览器是怎么对 HTML5 的离线储存资源进行管理和加载的呢

- 在线的情况下,浏览器发现 html 头部有 manifest 属性,它会请求 manifest 文件,如果是第一次访问 app,那么浏览器就会根据manifest文件的内容下载相应的资源并且进行离线存储。如果已经访问过 app 并且资源已经离线存储了,那么浏览器就会使用离线的资源加载页面,然后浏览器会对比新的 manifest 文件与旧的 manifes t文件,如果文件没有发生改变,就不做任何操作,如果文件改变了,那么就会重新下载文件中的资源并进行离线存储。
- 离线的情况下,浏览器就直接使用离线存储的资源。

12、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









o cookie 设置的 cookie 过期时间之前一直有效,即使窗口或浏览器天材

13、iframe有那些缺点?

- iframe 会阻塞主页面的 Onload 事件
- 搜索引擎的检索程序无法解读这种页面,不利于 SEO
- 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 iframe 添加 src 属性值,这样可以绕开以上两个问题

14、WEB标准以及W3C标准是什么?

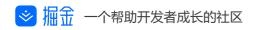
• 标签闭合、标签小写、不乱嵌套、使用外链 css 和 js 、结构行为表现的分离

15、xhtml和html有什么区别?

- 一个是功能上的差别
 - 主要是 XHTML 可兼容各大浏览器、手机以及 PDA , 并且浏览器也能快速正确地编译网页
- 另外是书写习惯的差别
 - o XHTML 元素必须被正确地嵌套,闭合,区分大小写,文档必须拥有根元素

16、Doctype作用? 严格模式与混杂模式如何区分? 它们有何意义?

- 页面被加载的时, link 会同时被加载,而@imort 页面被加载的时, link 会同时被加载,而@import 引用的 CSS 会等到页面被加载完再加载
 - import 只在 IE5 以上才能识别,而 link 是 XHTML 标签,无兼容问题 link 方式的样式的权重 高于 @import 的权重
- <!DOCTYPE> 声明位于文档中的最前面,处于 <html> 标签之前。告知浏览器的解析器,用什么 文档类型 规范来解析这个文档
- 严格模式的排版和 JS 运作模式是以该浏览器支持的最高标准运行
- 在混杂模式中,页面以宽松的向后兼容的方式显示。模拟老式浏览器的行为以防止站点无法工作。 DOCTYPE 不存在或格式不正确会导致文档以混杂模式呈现



登录・注册

- 空兀系:
 <hr> <input> <link> <meta>
- 行内元素不可以设置宽高,不独占一行
- 块级元素可以设置宽高,独占一行

18、HTML全局属性(global attribute)有哪些

• class:为元素设置类标识

• data-*: 为元素增加自定义属性

• draggable:设置元素是否可拖拽

• id:元素 id, 文档内唯一

• lang:元素内容的的语言

• style: 行内 css 样式

• title:元素相关的建议信息

\$CSS部分

1、css sprite是什么,有什么优缺点

- 概念:将多个小图片拼接到一个图片中。通过 background-position 和元素尺寸调节需要显示的 背景图案。
- 优点:
 - 。 减少 HTTP 请求数,极大地提高页面加载速度
 - 。 增加图片信息重复度, 提高压缩比, 减少图片大小
 - 。 更换风格方便, 只需在一张或几张图片上修改颜色或样式即可实现
- 缺点:
 - 。 图片合并麻烦
 - 。 维护麻烦, 修改一个图片可能需要从新布局整个图片, 样式
- 2、 display: none; 与 visibility: hidden; 的区别
 - 联系:它们都能让元素不可见
- 掃金 一个帮助开发者成长的社区





- 。 display: none;是非继承属性, 子孙节点消失田士元素从渲染树消失造成, 通过修改子孙节点属性无法显示; visibility: hidden; 是继承属性, 子孙节点消失由于继承了 hidden, 通过设置 visibility: visible; 可以让子孙节点显式
- 。 修改常规流中元素的 display 通常会造成文档重排。修改 visibility 属性只会造成本元素的 重绘。
- 。 读屏器不会读取 display: none ;元素内容; 会读取 visibility: hidden; 元素内容

3、 link 与 @import 的区别

- 1. link 是 HTML 方式, @import 是CSS方式
- 2. link 最大限度支持并行下载, @import 过多嵌套导致串行下载, 出现 FOUC
- 3. link 可以通过 rel="alternate stylesheet" 指定候选样式
- 4. 浏览器对 link 支持早于 @import , 可以使用 @import 对老浏览器隐藏样式
- 5. @import 必须在样式规则之前,可以在css文件中引用其他文件
- 6. 总体来说: link 优于 @import

4、什么是FOUC?如何避免

- Flash Of Unstyled Content: 用户定义样式表加载之前浏览器使用默认样式显示文档,用户样式加载渲染之后再从新显示文档,造成页面闪烁。
- 解决方法: 把样式表放到文档的 head

5、如何创建块级格式化上下文(block formatting context),BFC有什么用

- 创建规则:
 - 。根元素
 - 浮动元素 (float 不是 none)
 - 。 绝对定位元素(position 取值为 absolute 或 fixed)
 - o display 取值为 inline-block , table-cell , table-caption , flex , inline-flex 之一的元素
 - o overflow 不是 visible 的元素
- 作用:
 - 。 可以包含浮动元素



一个帮助开发者成长的社区



※ 掘缶 前▼

登录·注册

- 否则,如果 position 值为 absolute 或者 fixed ,框就是绝对定位的, float 的计算值为 none , display 根据下面的表格进行调整。
- 否则,如果 float 不是 none,框是浮动的, display 根据下表进行调整
- 否则,如果元素是根元素, display 根据下表进行调整
- 其他情况下 display 的值为指定值
- 总结起来: 绝对定位、浮动、根元素都需要调整 display

7、清除浮动的几种方式, 各自的优缺点

- 使用空标签清除浮动 clear:both (缺点,增加无意义的标签)
- 使用 overflow:auto (使用 zoom:1 用于兼容 IE , 缺点:内部宽高超过父级 div 时,会出现滚动条)
- 用 afert 伪元素清除浮动(IE8 以上和非 IE 浏览器才支持, 目前: 大型网站都有使用

8、为什么要初始化CSS样式?

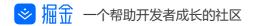
- 因为浏览器的兼容问题,不同浏览器对有些标签的默认值是不同的,如果没对 css 初始化往往会 出现浏览器之间的页面显示差异。
- 当然,初始化样式会对 SEO 有一定的影响,但鱼和熊掌不可兼得,但力求影响最小的情况下初始
 化

9、css3有哪些新特性

- 新增各种 css 选择器
- 圆角 border-radius
- 多列布局
- 阴影和反射
- 文字特效 text-shadow
- 线性渐变
- 旋转 transform

CSS3新增伪类有那些?

- p:first-of-type 选择属于其父元素的首个 元素的每个 元素。
- p:last-of-type 选择属于其父元素的最后 元素的每个 元素。
- p:only-of-type 选择属于其父元素唯一的 元素的每个 元素。
- n·only-child 选择屋干甘心元素的唯一子元素的每个 /n> 元妻







- :enabled
- :disabled 控制表单控件的禁用状态。
- :checked 单选框或复选框被选中

10、display有哪些值?说明他们的作用

- block 象块类型元素一样显示。
- none 缺省值。象行内元素类型一样显示。
- inline-block 象行内元素一样显示,但其内容象块类型元素一样显示。
- list-item 象块类型元素一样显示,并添加样式列表标记。
- table 此元素会作为块级表格来显示
- inherit 规定应该从父元素继承 display 属性的值

11、介绍一下标准的CSS的盒子模型?低版本IE的盒子模型有什么不同的?

- 有两种, IE 盒子模型、 W3C 盒子模型;
- 盒模型: 内容(content)、填充(padding)、边界(margin)、边框(border);
- 区别: IE 的C ontent 部分把 border 和 padding 计算了进去;

12、CSS优先级算法如何计算?

- 优先级就近原则,同权重情况下样式定义最近者为准
- 载入样式以最后载入的定位为准
- 优先级为: !important > id > class > tag important 比 内联优先级高

13、对BFC规范的理解?

• 它决定了元素如何对其内容进行定位,以及与其他元素的关系和相互作用

14、谈谈浮动和清除浮动

• 浮动的框可以向左或向右移动,直到他的外边缘碰到包含框或另一个浮动框的边框为止。由于浮 动框不在文档的普通流中,所以文档的普通流的块框表现得就像浮动框不存在一样。浮动的块框 会漂浮在文档普通流的块框上

15、position的值, relative和absolute定位原点是



😺 掘缶 一个帮助开发者成长的社区



inherit 规定从父元素继承 position 属性的值

16、display:inline-block 什么时候不会显示间隙? (携程)

- 移除空格
- 使用 margin 负值
- 使用 font-size:0
- letter-spacing
- word-spacing

17、PNG,GIF,JPG的区别及如何选

- GIF
 - 8位像素, 256色
 - 。 无损压缩
 - 。 支持简单动画
 - 。 支持 boolean 透明
 - 。 适合简单动画
- JPEG
 - 。 颜色限于 256
 - 。 有损压缩
 - 。 可控制压缩质量
 - 。 不支持透明
 - 。适合照片
- PNG
 - 有 PNG8 和 truecolor PNG
 - 。 PNG8 类似 GIF 颜色上限为 256 , 文件小, 支持 alpha 透明度, 无动画
 - 。 适合图标、背景、按钮

\$JavaScript

1、闭包



※ 掘金 一个帮助开发者成长的社区



- 闭包的特性:
 - 。 函数内再嵌套函数
 - 。 内部函数可以引用外层的参数和变量
 - 。 参数和变量不会被垃圾回收机制回收

说说你对闭包的理解

• 使用闭包主要是为了设计私有的方法和变量。闭包的优点是可以避免全局变量的污染,缺点是闭 包会常驻内存,会增大内存使用量,使用不当很容易造成内存泄露。在js中,函数即闭包,只有函 数才会产生作用域的概念

2、说说你对作用域链的理解

- 作用域链的作用是保证执行环境里有权访问的变量和函数是有序的,作用域链的变量只能向上访 问,变量访问到 window 对象即被终止,作用域链向下访问变量是不被允许的
- 简单的说,作用域就是变量与函数的可访问范围,即作用域控制着变量与函数的可见性和生命周 期

3、JavaScript原型,原型链?有什么特点?

- 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
- 关系: instance.constructor.prototype = instance.__proto__
- 特点:
 - 。 JavaScript 对象是通过引用来传递的,我们创建的每个新对象实体中并没有一份属于自己的 原型副本。当我们修改原型时,与之相关的对象也会继承这一改变
- 当我们需要一个属性的时, Javascript 引擎会先看当前对象中是否有这个属性, 如果没有的
- 就会查找他的 Prototype 对象是否有这个属性,如此递推下去,一直检索到 Object 内建对象

4、请解释什么是事件代理



😺 掘缶 一个帮助开发者成长的社区



可以实现当新增于对家时无需再次对具绑定

5、Javascript如何实现继承?

- 构造继承
- 原型继承
- 实例继承
- 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

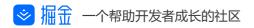
var demo = new Child();
alert(demo.age);
alert(demo.name);//得到被继承的属性
}
```

6、谈谈This对象的理解

- this 总是指向函数的直接调用者 (而非间接调用者)
- 如果有 new 关键字, this 指向 new 出来的那个对象
- 在事件中, this 指向触发这个事件的对象,特殊的是, IE 中的 attachEvent 中的 this 总是指向全局对象 Window

7、事件模型

- 冒泡型事件: 当你使用事件冒泡时, 子级元素先触发, 父级元素后触发
- 捕获型事件: 当你使用事件捕获时, 父级元素先触发, 子级元素后触发
- DOM 事件流:同时支持两种事件模型:捕获型事件和冒泡型事件
- 阳止国沟・左 Line th 信田 ctonDonorgation () 古注・左に下沿署 cancel Bubble -







8、new操作符具体干了什么呢?

- 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
- 属性和方法被加入到 this 引用的对象中
- 新创建的对象由 this 所引用,并且最后隐式的返回 this

9、Ajax原理

- Ajax 的原理简单来说是在用户和服务器之间加了一个中间层(AJAX 引擎),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 javascrip t来操作 DOM 而更新页面。使用户操作与服务器响应异步化。这其中最关键的一步就是从服务器获得请求数据
- Ajax 的过程只涉及 JavaScript 、 XMLHttpRequest 和 DOM 。 XMLHttpRequest 是 aja X的核心机制

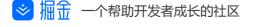
```
// 1. 创建连接
  var xhr = null;
  xhr = new XMLHttpRequest()
  // 2. 连接服务器
  xhr.open('get', url, true)
  // 3. 发送请求
  xhr.send(null);
  // 4. 接受请求
  xhr.onreadystatechange = function(){
       if(xhr.readyState == 4){
          if(xhr.status == 200){
              success(xhr.responseText);
          } else { // fail
              fail && fail(xhr.status);
      }
  }
```

10、如何解决跨域问题?

• jsonp 、 iframe 、 window.name 、 window.postMessage 、服务器上设置代理页面

11、模块化开发怎么做?

• 立即执行函数,不暴露私有成员







12、异步加载JS的方式有哪些?

- defer, 只支持 IE
- async:
- 创建 script , 插入到 DOM 中, 加载完毕后 callBack

13、那些操作会造成内存泄漏?

- 内存泄漏指任何对象在您不再拥有或需要它之后仍然存在
- setTimeout 的第一个参数使用字符串而非函数的话,会引发内存泄漏
- 闭包使用不当

14、XML和JSON的区别?

- 数据体积方面
 - JSON 相对于XML来讲,数据的体积小,传递的速度更快些。
- 数据交互方面
 - 。 JSON 与 JavaScript 的交互更加方便,更容易解析处理,更好的数据交互
- 数据描述方面
 - JSON 对数据的描述性比 XML 较差
- **上上**公古市 古石



↗ 掘金

一个帮助开发者成长的社区



15、谈谈你对webpack的看法

WebPack 是一个模块打包工具,你可以使用 WebPack 管理你的模块依赖,并编绎输出模块们所需的静态文件。它能够很好地管理、打包 Web 开发中所用到的 HTML 、 Javascript 、 CSS 以及各种静态文件(图片、字体等),让开发过程更加高效。对于不同类型的资源, webpack 有对应的模块加载器。 Webpack 模块打包器会分析模块间的依赖关系,最后 生成了优化且合并后的静态资源

16、说说你对AMD和Commonjs的理解

- CommonJS 是服务器端模块的规范, Node.js 采用了这个规范。 CommonJS 规范加载模块是同步的,也就是说,只有加载完成,才能执行后面的操作。 AMD 规范则是非同步加载模块,允许指定回调函数
- AMD 推荐的风格通过返回一个对象做为模块对象, CommonJS 的风格通过对 module.exports 或 exports 的属性赋值来达到暴露模块对象的目的

17、常见web安全及防护原理

- sql 注入原理
 - 。 就是通过把 SQL 命令插入到 Web 表单递交或输入域名或页面请求的查询字符串,最终达到欺骗服务器执行恶意的SQL命令
- 总的来说有以下几点
 - 。 永远不要信任用户的输入,要对用户的输入进行校验,可以通过正则表达式,或限制长度, 对单引号和双 "-" 进行转换等
 - 。 永远不要使用动态拼装SQL,可以使用参数化的 SQL 或者直接使用存储过程进行数据查询存取
 - 。 永远不要使用管理员权限的数据库连接, 为每个应用使用单独的权限有限的数据库连接
 - 。 不要把机密信息明文存放,请加密或者 hash 掉密码和敏感的信息

XSS原理及防范

Xss(cross-site scripting) 攻击指的是攻击者往 Web 页面里插入恶意 html 标签或者 javascript 代码。比如:攻击者在论坛中放一个看似安全的链接,骗取用户点击后,窃取 cookie 中的用户私密信息;或者攻击者在论坛中加一个恶意表单,当用户提交表单的时候,却把信息传送到攻击者的服务器中,而不是用户原本以为的信任站点





好, 至少可以堵任超过一半的XSS 攻击

XSS与CSRF有什么区别吗?

- XSS 是获取信息,不需要提前知道其他用户页面的代码和数据包。 CSRF 是代替用户完成指定的 动作,需要知道其他用户页面的代码和数据包。要完成一次 CSRF 攻击,受害者必须依次完成两个 步骤
- 登录受信任网站 A , 并在本地生成 Cookie
- 在不登出 A 的情况下, 访问危险网站 B

CSRF的防御

- 服务端的 CSRF 方式方法很多样,但总的思想都是一致的,就是在客户端页面增加伪随机数
- 通过验证码的方法

18、用过哪些设计模式?

- 工厂模式:
 - 。 工厂模式解决了重复实例化的问题, 但还有一个问题,那就是识别问题, 因为根本无法
 - 。 主要好处就是可以消除对象间的耦合,通过使用工程方法而不是 new 关键字
- 构造函数模式
 - 。 使用构造函数的方法,即解决了重复实例化的问题,又解决了对象识别的问题,该模式与工厂模式的不同之处在于
 - 。 直接将属性和方法赋值给 this 对象;

19、为什么要有同源限制?

- 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
- 举例说明:比如一个黑客程序,他利用 Iframe 把真正的银行登录页面嵌到他的页面上,当你使用真实的用户名,密码登录时,他的页面就可以通过 Javascript 读取到你的表单中 input 中的内容,这样用户名,密码就轻松到手了。







- clientWidth/clientHeight 返回值只包含content + padding, 如果有滚动条, 也不包含滚动条
- scrollWidth/scrollHeight 返回值包含content + padding + 溢出内容的尺寸

21、javascript有哪些方法定义对象

```
对象字面量: var obj = {};
构造函数: var obj = new Object();
Object.create(): var obj = Object.create(Object.prototype);
```

22、常见兼容性问题?

- png24 位的图片在iE6浏览器上出现背景,解决方案是做成 PNG8
- 浏览器默认的 margin 和 padding 不同。解决方案是加一个全局的 *{margin:0;padding:0;} 来统一,, 但是全局效率很低, 一般是如下这样解决:

```
body,ul,li,ol,dl,dt,dd,form,input,h1,h2,h3,h4,h5,h6,p{
margin:0;
padding:0;
}
```

- 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.

22、说说你对promise的了解

- 依照 Promise/A+ 的定义, Promise 有四种状态:
 - o pending: 初始状态, 非 fulfilled 或 rejected.
 - o fulfilled: 成功的操作.
 - o rejected: 失败的操作.
 - settled: Promise 已被 fulfilled 或 rejected , 且不是 pending
- 另外, fulfilled 与 rejected 一起合称 settled
- Promise 对象用来讲行延识(deferred)和显光(asynchronous)计算







```
if (...) { // succeed
    resolve(result);
} else { // fails
    reject(Error(errMessage));
}
```

var promise = new Promise(function(resolve, reject) {

• Promise 实例拥有 then 方法(具有 then 方法的对象,通常被称为 thenable)。它的使用方法如下:

```
promise.then(onFulfilled, onRejected)
```

• 接收两个函数作为参数,一个在 fulfilled 的时候被调用,一个在 rejected 的时候被调用,接收参数就是 future , onFulfilled 对应 resolve , onRejected 对应 reject

23、你觉得jQuery源码有哪些写的好的地方

- jquery 源码封装在一个匿名函数的自执行环境中,有助于防止变量的全局污染,然后通过传入 window 对象参数,可以使 window 对象作为局部变量使用,好处是当 jquery 中访问 window 对象 的时候,就不用将作用域链退回到顶层作用域了,从而可以更快的访问window对象。同样,传入 undefined 参数,可以缩短查找 undefined 时的作用域链
- jquery 将一些原型属性和方法封装在了 jquery.prototype 中,为了缩短名称,又赋值给了 jquery.fn ,这是很形象的写法
- 有一些数组或对象的方法经常能使用到, jQuery 将其保存为局部变量以提高访问速度
- jquery 实现的链式调用可以节约代码,所返回的都是同一个对象,可以提高代码效率

24、vue、react、angular

• Vue.js

一个用于创建 web 交互界面的库,是一个精简的 MVVM。它通过双向数据绑定把 View 层和 Model 层连接了起来。实际的 DOM 封装和输出格式都被抽象为了 Directives 和 Filters







所有切能,模板切能強大丰富,目带了丰富的 Angular 指令

• react

React 仅仅是 VIEW 层是 facebook 公司。推出的一个用于构建 UI 的一个库,能够实现服务器端的渲染。用了 virtual dom ,所以性能很好。

25、Node的应用场景

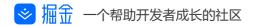
- 特点:
 - 1、它是一个 Javascript 运行环境
 - 2、依赖于 Chrome V8 引擎进行代码解释
 - 。 3、事件驱动
 - 4、非阻塞 I/0
 - 。 5、单进程, 单线程
- 优点:
 - 。 高并发 (最重要的优点)
- 缺点:
 - 。 1、只支持单 核CPU ,不能充分利用 CPU
 - 。 2、可靠性低, 一旦代码某个环节崩溃, 整个系统都崩溃

26、谈谈你对AMD、CMD的理解

- CommonJS 是服务器端模块的规范, Node.js 采用了这个规范。 CommonJS 规范加载模块是同步的,也就是说,只有加载完成,才能执行后面的操作。 AMD 规范则是非同步加载模块,允许指定回调函数
- AMD 推荐的风格通过返回一个对象做为模块对象, CommonJS 的风格通过对 module.exports 或 exports 的属性赋值来达到暴露模块对象的目的

27、那些操作会造成内存泄漏?

• 内存泄漏指任何对象在您不再拥有或需要它之后仍然存在







- session
- url 重写
- 隐藏 input
- ip 地址

29、介绍js的基本数据类型

• Undefined Null Boolean Number String

30、介绍js有哪些内置对象?

- Object 是 JavaScript 中所有对象的父对象
- 数据封装类对象: Object 、 Array 、 Boolean 、 Number 和 String
- 其他对象: Function 、 Arguments 、 Math 、 Date 、 RegExp 、 Error

31、说几条写JavaScript的基本规范?

- 不要在同一行声明多个变量
- 请使用 ===/!== 来比较 true/false 或者数值
- 使用对象字面量替代 new Array 这种形式
- 不要使用全局函数
- Switch 语句必须带有 default 分支
- If 语句必须使用大括号
- for-in 循环中的变量 应该使用 var 关键字明确限定作用域,从而避免作用域污

32、JavaScript有几种类型的值? , 你能画一下他们的内存图吗?

- 栈:原始数据类型 (Undefined, Null, Boolean, Number、String)
- 堆:引用数据类型(对象、数组和函数)
- 两种类型的区别是: 存储位置不同;
- 原始数据类型直接存储在栈(stack)中的简单数据段,占据空间小、大小固定,属于被频繁使用数据,所以放入栈中存储;
- 引用数据类型存储在堆(heap)中的对象,占据空间大、大小不固定,如果存储在栈中,将会影响程序运行的性能;引用数据类型在栈中存储了指针,该指针指向堆中该实体的起始地址。当解释器寻找引用值时,会首先检索其
- 在栈中的地址, 取得地址后从堆中获得实体







33、javascript创建对象的几种方式?

javascript 创建对象简单的说,无非就是使用内置对象或各种自定义对象,当然还可以用 JSON;但写法有很多种,也能混合使用

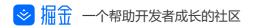
• 对象字面量的方式

```
person={firstname:"Mark",lastname:"Yun",age:25,eyecolor:"black"};
```

• 用 function 来模拟无参的构造函数

```
function Person(){}

var person=new Person();//定义一个function, 如果使用new"实例化",该function可以看作是一个Class
    person.name="Mark";
    person.age="25";
    person.work=function(){
        alert(person.name+" hello...");
```







```
function Pet(name,age,hobby){
    this.name=name;//this作用域: 当前对象
    this.age=age;
    this.hobby=hobby;
    this.eat=function(){
        alert("我叫"+this.name+",我喜欢"+this.hobby+",是个程序员");
    }
}
var maidou =new Pet("麦兜",25,"coding");//实例化、创建对象
maidou.eat();//调用eat方法
```

• 用工厂方式来创建(内置对象)

```
var wcDog =new Object();
    wcDog.name="旺财";
    wcDog.age=3;
    wcDog.work=function(){
        alert("我是"+wcDog.name+",汪汪汪....");
    }
    wcDog.work();
```

• 用原型方式来创建

```
function Dog(){

}

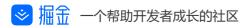
Dog.prototype.name="旺财";

Dog.prototype.eat=function(){
    alert(this.name+"是个吃货");
    }

var wangcai = new Dog();
    wangcai.eat();
```

• 用混合方式来创建

```
function Car(name,price){
    this.name=name;
    this.price=price;
}
Car.prototype.sell=function(){
    alert("我是"+this.name+", 我现在卖"+this.price+"万元");
```







34、eval是做什么的?

- 它的功能是把对应的字符串解析成 JS 代码并运行
- 应该避免使用 eval , 不安全, 非常耗性能 (2 次, 一次解析成 js 语句, 一次执行)
- 由 JSON 字符串转换为JSON对象的时候可以用 eval, var obj =eval('('+ str +')')

35、null, undefined 的区别?

- undefined 表示不存在这个值。
- undefined:是一个表示"无"的原始值或者说表示"缺少值",就是此处应该有一个值,但是还没有定义。当尝试读取时会返回 undefined
- 例如变量被声明了,但没有赋值时,就等于 undefined
- null 表示一个对象被定义了, 值为 "空值"
- null:是一个对象(空对象,没有任何属性和方法)
- 例如作为函数的参数,表示该函数的参数不是对象;
- 在验证 null 时,一定要使用 === ,因为 == 无法分别 null 和 undefined

36、["1", "2", "3"].map(parseInt) 答案是多少?

- [1, NaN, NaN] 因为 parseInt 需要两个参数 (val, radix), 其中 radix 表示解析时用的基数。
- map 传了 3 个 (element, index, array), 对应的 radix 不合法导致解析失败。

37、javascript 代码中的"use strict";是什么意思?使用它区别是什么?

• use strict 是一种 ECMAscript 5 添加的 (严格) 运行模式,这种模式使得 Javascript 在更严格的条件下运行,使 JS 编码更加规范化的模式,消除 Javascript 语法的一些不合理、不严谨之处,减少一些怪异行为

38、JSON 的了解? **

• JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式







```
var obj =eval('('+ str +')');
var obj = str.parseJSON();
var obj = JSON.parse(str);
```

JSON 对象转换为JSON字符串:

```
var last=obj.toJSONString();
var last=JSON.stringify(obj);
```

39、js延迟加载的方式有哪些?

• defer 和 async 、动态创建 DOM 方式 (用得最多) 、按需异步载入 js

40、同步和异步的区别?

- 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

41、渐进增强和优雅降级

- 渐进增强: 针对低版本浏览器进行构建页面, 保证最基本的功能, 然后再针对高级浏览器进行效 果、交互等改进和追加功能达到更好的用户体验。
- 优雅降级: 一开始就构建完整的功能, 然后再针对低版本浏览器进行兼容

42、defer和async

- defer 并行加载 js 文件, 会按照页面上 script 标签的顺序执行
- async 并行加载 is 文件,下载完成立即执行,不会按照页面上 script 标签的顺序执行

43、说说严格模式的限制

- 变量必须声明后再使用
- 函数的参数不能有同名属性, 否则报错
- ▲ 不能估田 运行



🤝 掘缶 一个帮助开发者成长的社区



- property 就是 dom 元素在 js 中作为对象拥有的属性。
- 对于 html 的标准属性来说, attribute 和 property 是同步的, 是会自动更新的
- 但是对于自定义的属性来说,他们是不同步的

45、谈谈你对ES6的理解

- 新增模板字符串(为 JavaScript 提供了简单的字符串插值功能)
- 箭头函数
- for-of (用来遍历数据—例如数组中的值。)
- arguments 对象可被不定参数和默认参数完美代替。
- ES6 将p romise 对象纳入规范,提供了原生的 Promise 对象。
- 增加了 let 和 const 命令, 用来声明变量。
- 增加了块级作用域。
- let 命令实际上就增加了块级作用域。
- 还有就是引入 module 模块的概念

\$编程题

1、写一个通用的事件侦听器函数

```
// event(事件)工具集,来源: github.com/markyun
  markyun.Event = {
      // 视能力分别使用dom0||dom2||IE方式 来绑定事件
      // 参数: 操作的元素,事件名称 ,事件处理程序
      addEvent : function(element, type, handler) {
          if (element.addEventListener) {
             //事件类型、需要执行的函数、是否捕捉
             element.addEventListener(type, handler, false);
          } else if (element.attachEvent) {
             element.attachEvent('on' + type, function() {
                 handler.call(element);
             });
          } else {
             element['on' + type] = handler;
          }
      },
      // 移除事件
```



首页▼

登录・注册

```
} else {
       element['on' + type] = null;
   }
},
// 阻止事件 (主要是事件冒泡,因为IE不支持事件捕获)
stopPropagation : function(ev) {
   if (ev.stopPropagation) {
       ev.stopPropagation();
   } else {
       ev.cancelBubble = true;
   }
},
// 取消事件的默认行为
preventDefault : function(event) {
   if (event.preventDefault) {
       event.preventDefault();
   } else {
       event.returnValue = false;
   }
},
// 获取事件目标
getTarget : function(event) {
   return event.target || event.srcElement;
}
```

2、如何判断一个对象是否为数组

```
function isArray(arg) {
   if (typeof arg === 'object') {
      return Object.prototype.toString.call(arg) === '[object Array]';
   }
   return false;
}
```

3、冒泡排序

• 每次比较相邻的两个数,如果后一个比前一个小,换位置

```
var arr = [3, 1, 4, 6, 5, 7, 2];
```





```
temp = arr[j];
    arr[j] = arr[j + 1];
    arr[j + 1] = temp;
}

}
return arr;
}

console.log(bubbleSort(arr));
```

4、快速排序

• 采用二分法,取出中间数,数组每次和中间数比较,小的放到左边,大的放到右边

```
var arr = [3, 1, 4, 6, 5, 7, 2];
function quickSort(arr) {
    if(arr.length == 0) {
        return []; // 返回空数组
    }
    var cIndex = Math.floor(arr.length / 2);
    var c = arr.splice(cIndex, 1);
    var 1 = [];
    var r = [];
    for (var i = 0; i < arr.length; i++) {</pre>
        if(arr[i] < c) {
            1.push(arr[i]);
        } else {
            r.push(arr[i]);
    }
    return quickSort(1).concat(c, quickSort(r));
}
console.log(quickSort(arr));
```

5. 编写一个方法 求一个字符串的字节长度







```
var len = str.length;

var bytes = len;

for(var i=0; i<len; i++){

    if (str.charCodeAt(i) > 255) bytes++;
}

return bytes;
}

alert(GetBytes("你好,as"));
```

\$其他

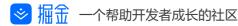
1、谈谈你对重构的理解

- 网站重构:在不改变外部行为的前提下,简化结构、添加可读性,而在网站前端保持一致的行为。也就是说是在不改变UI的情况下,对网站进行优化,在扩展的同时保持一致的UI
- 对于传统的网站来说重构通常是:
 - 。 表格(table)布局改为 DIV+CSS
 - 。 使网站前端兼容于现代浏览器(针对于不合规范的 css 、如对IE6有效的)
 - 。 对于移动平台的优化
 - 。 针对于 SEO 进行优化

2、什么样的前端代码是好的

• 高复用低耦合,这样文件小,好维护,而且好扩展。

3、对前端工程师这个职位是怎么样理解的?它的前景会怎么样?







- 有了Node.js, 丽端可以实现服务端的一些事情
- 前端是最贴近用户的程序员,前端的能力就是能让产品从 90分进化到 100 分,甚至更好,
- 与团队成员, **UI** 设计, 产品经理的沟通;
- 做好的页面结构,页面重构和用户体验;

4、你觉得前端工程的价值体现在哪

- 为简化用户使用提供技术支持(交互部分)
- 为多个浏览器兼容性提供支持
- 为提高用户浏览速度(浏览器性能)提供支持
- 为跨平台或者其他基于webkit或其他渲染引擎的应用提供支持
- 为展示数据提供支持(数据接口)

5、平时如何管理你的项目?

- 先期团队必须确定好全局样式 (globe.css) , 编码模式 (utf-8) 等;
- 编写习惯必须一致(例如都是采用继承式的写法,单样式都写成一行);
- 标注样式编写人,各模块都及时标注(标注关键样式调用的地方);
- 页面进行标注(例如页面模块开始和结束);
- CSS 跟 HTML 分文件夹并行存放,命名都得统一(例如 style.css);
- JS 分文件夹存放 命名以该 JS 功能为准的英文翻译。
- 图片采用整合的 images.png png8 格式文件使用 尽量整合在一起使用方便将来的管理

人事面



这 掘缶 一个帮助开发者成长的社区



- 你为什么会选择这个行业,职位?
- 你觉得你适合从事这个岗位吗?
- 你有什么职业规划?
- 你对工资有什么要求?
- 如何看待前端开发?
- 未来三到五年的规划是怎样的?

一些问题

- 你的项目中技术难点是什么?
- 遇到了什么问题?
- 你是怎么解决的?
- 最近在看哪些前端方面的书?
- 平时是如何学习前端开发的?
- 为什么大型应用需要从多个域名请求资源?
- 什么样的页面具有良好的用户体验?
- 是否了解 Web 注入攻击,说下原理,最常见的两种攻击 (XSS 和 CSRF) 了解到什么程度

本文对你有帮助?欢迎扫码加入前端学习小组微信群:



面试

披术圈子



微信交流群



相关热门文章

不要再去背面试的知识点了

Lijie0981

有趣的前端编程题: 携程 2018 春招

RavJune 7



😻 掘金 一个帮助开发者成长的社区





狸寻寻君

实习都如此艰难 | 掘金技术征文

李醇罡 1

评论

登录

说说你的看法

FHY爱学习

能拉我进群吗

▲ 0 评论 9天前

张绪耀重名了

很好的 js 知识点

▲ 0 评论 1年前

Poetries 添加我的职位@公司名

PDF 版下载: http://pan.baidu.com/s/1gfiNfHL

Md 原文: https://github.com/poetries/poetries.github.io/blob/dev/source/_posts/front-end-interview-summary.md

,

▲ 0 评论 1年前