#前端开发面试题

前言

只看问题点这里

看全部问题和答案点这里

本文由我收集总结了一些前端面试题，初学者阅后也要用心钻研其中的原理，重要知识需要系统学习、透彻学习，形成自己的知识链。万不可投机取巧，临时抱佛脚只求面试侥幸混过关是错误的！也是不可能的！不可能的！不可能的！

前端还是一个年轻的行业，新的行业标准， 框架， 库都不断在更新和新增，正如赫门在 2015 深 JS 大会上的《前端服务化之路》主题演讲中说的一句话：“每 18 至 24 个月，前端都会难一倍”，这些变化使前端的能力更加丰富、创造的应用也会更加完美。所以关注各种前端技术，跟上快速变化的节奏，也是身为一个前端程序员必备的技能之一。

最近也收到许多微博私信的鼓励和更正题目信息，后面会经常更新题目和答案到 github 博客。希望前端 er 达到既能使用也会表达，对理论知识有自己的理解。可根据下面的知识点一个一个去进阶学习，形成自己的职业技能链。

面试有几点需注意：(来源寒冬 winter 老师，github:@wintercn)

1. 面试题目： 根据你的等级和职位的变化，入门级到专家级，广度和深度都会有所增加。

2. 题目类型： 理论知识、算法、项目细节、技术视野、开放性题、工作案例。

3. 细节追问： 可以确保问到你开始不懂或面试官开始不懂为止，这样可以大大延展题目的区分度和深度，知道你的实际能力。因为这种知识关联是长时期的学习，临时抱佛脚绝对是记不住的。

4. 回答问题再棒，面试官（可能是你面试职位的直接领导），会考虑我要不要这个人做我的同事？所以态度很重要、除了能做事，还要会做人。（感觉更像是相亲(*:ω:*)）

5. 资深的前端开发能把 absolute 和 relative 弄混，这样的人不要也罢，因为团队需要的是：你这个人具有可以依靠的才能（靠谱）。

前端开发知识点：

```
HTML&CSS：
```

对 Web 标准的理解、浏览器内核差异、兼容性、hack、CSS 基本功：布局、盒子模型、选择器优先级、
HTML5、CSS3、Flexbox

JavaScript：
数据类型、运算、对象、Function、继承、闭包、作用域、原型链、事件、RegExp、JSON、Ajax、
DOM、BOM、内存泄漏、跨域、异步装载、模板引擎、前端 MVC、路由、模块化、Canvas、ECMAScript 6、Nodejs

其他：
移动端、响应式、自动化构建、HTTP、离线存储、WEB 安全、优化、重构、团队协作、可维护、易用性、SEO、UED、架构、职业生涯、快速学习能力

作为一名前端工程师，**无论工作年头长短都应该掌握的知识点**：

此条由 王子墨 发表在 攻城师的实验室

1、DOM 结构 —— 两个节点之间可能存在哪些关系以及如何在节点之间任意移动。

2、DOM 操作 —— 如何添加、移除、移动、复制、创建和查找节点等。

3、事件 —— 如何使用事件，以及 IE 和标准 DOM 事件模型之间存在的差别。

4、XMLHttpRequest —— 这是什么、怎样完整地执行一次 GET 请求、怎样检测错误。

5、严格模式与混杂模式 —— 如何触发这两种模式，区分它们有何意义。

6、盒模型 —— 外边距、内边距和边框之间的关系，及 IE8 以下版本的浏览器中的盒模型

7、块级元素与行内元素 —— 怎么用 CSS 控制它们、以及如何合理的使用它们

8、浮动元素 —— 怎么使用它们、它们有什么问题以及怎么解决这些问题。

9、HTML 与 XHTML —— 二者有什么区别，你觉得应该使用哪一个并说出理由。

10、JSON —— 作用、用途、设计结构。

备注：

根据自己需要选择性阅读，面试题是对理论知识的总结，让自己学会应该如何表达。

资料答案不够正确和全面，欢迎欢迎 Star 和提交 issues。

格式不断修改更新中。

更新记录：
2016 年 3 月 25 日：新增 ECMAScript6 相关问题
###更新时间: 2016-3-25

HTML

- Doctype 作用？标准模式与兼容模式各有什么区别?

（1）、<!DOCTYPE>声明位于位于 HTML 文档中的第一行，处于 <html> 标签之前。告知浏览器的解析器用什么文档标准解析这个文档。DOCTYPE 不存在或格式不正确会导致文档以兼容模式呈现。

（2）、标准模式的排版 和 JS 运作模式都是以该浏览器支持的最高标准运行。在兼容模式中，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。

- HTML5 为什么只需要写 ？

HTML5 不基于 SGML，因此不需要对 DTD 进行引用，但是需要 doctype 来规范浏览器的行为（让浏览器按照它们应该的方式来运行）；

而 HTML4.01 基于 SGML,所以需要对 DTD 进行引用，才能告知浏览器文档所使用的文档类型。

- 行内元素有哪些？块级元素有哪些？ 空(void)元素有那些？

首先：CSS 规范规定，每个元素都有 display 属性，确定该元素的类型，每个元素都有默认的 display 值，如 div 的 display 默认值为“block”，则为“块级”元素；span 默认 display 属性值为“inline”，是“行内”元素。

（1）行内元素有：a b span img input select strong（强调的语气）

（2）块级元素有：div ul ol li dl dt dd h1 h2 h3 h4…p

（3）常见的空元素：

 <hr> <img> <input> <link> <meta>

鲜为人知的是：

<area> <base> <col> <command> <embed> <keygen> <param> <source> <track> <wbr>

不同浏览器（版本）、HTML4（5）、CSS2 等实际略有差异

参考: http://stackoverflow.com/questions/6867254/browsers-default-css-for-html-elements

- 页面导入样式时，使用 link 和@import 有什么区别？

（1）link 属于 XHTML 标签，除了加载 CSS 外，还能用于定义 RSS，定义 rel 连接属性等作用；而@import 是 CSS 提供的，只能用于加载 CSS;

（2）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;

（3）import 是 CSS2.1 提出的，只在 IE5 以上才能被识别，而 link 是 XHTML 标签，无兼容问题;

- 介绍一下你对浏览器内核的理解？

主要分成两部分：渲染引擎(layout engineer 或 Rendering Engine)和 JS 引擎。
渲染引擎：负责取得网页的内容（HTML、XML、图像等等）、整理讯息（例如加入 CSS 等），以及计算网页的显示方式，然后会输出至显示器或打印机。浏览器的内核的不同对于网页的语法解释会有不同，所以渲染的效果也不相同。所有网页浏览器、电子邮件客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。

- 常见的浏览器内核有哪些？

Trident 内核：IE,MaxThon,TT,The World,360,搜狗浏览器等。[又称 MSHTML]
Gecko 内核：Netscape6 及以上版本，FF,MozillaSuite/SeaMonkey 等
Presto 内核：Opera7 及以上。 [Opera 内核原为：Presto，现为：Blink;]
Webkit 内核：Safari,Chrome 等。 [Chrome 的：Blink（WebKit 的分支）]

详细文章：[浏览器内核的解析和对比](http://www.cnblogs.com/fullhouse/archive/2011/12/19/2293455.html)

- html5 有哪些新特性、移除了那些元素？如何处理 HTML5 新标签的浏览器兼容问题？如何区分 HTML 和 HTML5？

```
* 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。
        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;
        <!--[if lt IE 9]>
```

- <script> src="http://html5shim.googlecode.com/svn/trunk/html5.js"</script>
- <![endif]-->
-
- * 如何区分 HTML5： DOCTYPE 声明\新增的结构元素\功能元素

- 简述一下你对 HTML 语义化的理解？

- 用正确的标签做正确的事情。
- html 语义化让页面的内容结构化，结构更清晰，便于对浏览器、搜索引擎解析;
- 即使在没有样式 CSS 情况下也以一种文档格式显示，并且是容易阅读的;
- 搜索引擎的爬虫也依赖于 HTML 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。

- 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。
- 原理：HTML5 的离线存储是基于一个新建的.appcache 文件的缓存机制(不是存储技术)，通过这个文件上的解析清单离线存储资源，这些资源就会像 cookie 一样被存储了下来。之后当网络在处于离线状态下时，浏览器会通过被离线存储的数据进行页面展示。
-
-
- 如何使用：
- 1、页面头部像下面一样加入一个 manifest 的属性；
- 2、在 cache.manifest 文件的编写离线存储的资源；
- CACHE MANIFEST
- #v0.11
- CACHE:
- js/app.js
- css/style.css
- NETWORK:
- resourse/logo.png
- FALLBACK:
- / /offline.html
- 3、在离线状态时，操作 window.applicationCache 进行需求实现。

详细的使用请参考：

HTML5 离线缓存-manifest 简介

有趣的 HTML5：离线存储

- 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 manifest 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。

详细请参考：[有趣的 HTML5：离线存储]

- 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

- 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。

- Label 的作用是什么？是怎么用的？

- label 标签来定义表单控制间的关系,当用户选择该标签时，浏览器会自动将焦点转到和标签相关的表单控件上。
-
- <label for="Name">Number:</label>
- <input type=“text“name="Name" id="Name"/>
-
- <label>Date:<input type="text" name="B"/></label>

- HTML5 的 form 如何关闭自动完成功能？

- 给不想要提示的 form 或某个 input 设置为 autocomplete=off。

- 如何实现浏览器内多个标签页之间的通信? (阿里)

- WebSocket、SharedWorker；
- 也可以调用 localstorge、cookies 等本地存储方式；
-
- localstorge 另一个浏览上下文里被添加、修改或删除时，它都会触发一个事件，
- 我们通过监听事件，控制它的值来进行页面信息通信；
- 注意 quirks：Safari 在无痕模式下设置 localstorge 值时会抛出 QuotaExceededError 的异常；

- webSocket 如何兼容低浏览器？(阿里)

- Adobe Flash Socket 、
- ActiveX HTMLFile (IE) 、
- 基于 multipart 编码发送 XHR 、
- 基于长轮询的 XHR

- 页面可见性（Page Visibility API） 可以有哪些用途？

- 通过 visibilityState 的值检测页面当前是否可见，以及打开网页的时间等;
- 在页面被切换到其他后台进程的时候，自动暂停音乐或视频的播放；

- 如何在页面上实现一个圆形的可点击区域？

- 1、map+area 或者 svg
- 2、border-radius
- 3、纯 js 实现 需要求一个点在不在圆上简单算法、获取鼠标坐标等等

- 实现不使用 border 画出 1px 高的线，在不同浏览器的标准模式与怪异模式下都能保持一致的效果。

-

- 网页验证码是干嘛的，是为了解决什么安全问题。

- 区分用户是计算机还是人的公共全自动程序。可以防止恶意破解密码、刷票、论坛灌水；
- 有效防止黑客对某一个特定注册用户用特定程序暴力破解方式进行不断的登陆尝试。

- title 与 h1 的区别、b 与 strong 的区别、i 与 em 的区别？

- title 属性没有明确意义只表示是个标题，H1 则表示层次明确的标题，对页面信息的抓取也有很大的影响；
-
- strong 是标明重点内容，有语气加强的含义，使用阅读设备阅读网络时：会重读，而<B>是展示强调内容。

```
    i 内容展示为斜体，em 表示强调的文本；

    Physical Style Elements -- 自然样式标签
    b, i, u, s, pre
    Semantic Style Elements -- 语义样式标签
    strong, em, ins, del, code
    应该准确使用语义样式标签，但不能滥用，如果不能确定时首选使用自然样式标签。
```

CSS

- 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？

```
    （1）有两种， IE 盒子模型、W3C 盒子模型；
    （2）盒模型： 内容(content)、填充(padding)、边界(margin)、 边框(border)；
    （3）区　别： IE 的 content 部分把 border 和 padding 计算了进去;
```

- 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 ;
```

- 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:
            同权重: 内联样式表（标签内部）> 嵌入样式表（当前文件中）> 外部样式表（外部文件中）。
            !important >  id > class > tag
            important 比 内联优先级高
```

- CSS3 新增伪类有那些？

```
            举例：
            p:first-of-type  选择属于其父元素的首个 <p> 元素的每个 <p> 元素。
            p:last-of-type   选择属于其父元素的最后 <p> 元素的每个 <p> 元素。
        p:only-of-type       选择属于其父元素唯一的 <p> 元素的每个 <p> 元素。
            p:only-child             选择属于其父元素的唯一子元素的每个 <p> 元素。
            p:nth-child(2)   选择属于其父元素的第二个子元素的每个 <p> 元素。

            :after                   在元素之前添加内容,也可以用来做清除浮动。
            :before                  在元素之后添加内容
        :enabled
            :disabled                控制表单控件的禁用状态。
            :checked        单选框或复选框被选中。
```

- 如何居中 div？
    - 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;
  }
```

    - 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 */
 }
```

    - 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 */

```

```
    }
```

- 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 */

 }
```

- 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 */
 }
```

- display 有哪些值？说明他们的作用。

```
block          块类型。默认宽度为父元素宽度，可设置宽高，换行显示。
none           缺省值。象行内元素类型一样显示。
inline         行内元素类型。默认宽度为内容宽度，不可设置宽高，同行显示。
inline-block   默认宽度为内容宽度，可以设置宽高，同行显示。
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。
```

- 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。

- CSS3 有哪些新特性？

- 新增各种 CSS 选择器 （: not(.input)：所有 class 不是“input”的节点）
- 圆角 （border-radius:8px）
- 多列布局 （multi-column layout）
- 阴影和反射 （Shadow\Reflect）
- 文字特效 （text-shadow、）
- 文字渲染 （Text-decoration）
- 线性渐变 （gradient）
- 旋转 （transform）
- 缩放,定位,倾斜,动画,多背景
- 例如:transform:\scale(0.85,0.90)\ translate(0px,-30px)\ skew(-9deg,0deg)\Animation:

- 请解释一下 CSS3 的 Flexbox（弹性盒布局模型）,以及适用场景？

- 一个用于页面布局的全新 CSS3 功能，Flexbox 可以把列表放在同一个方向（从上到下排列，从左到右），并让列表能延伸到占用可用的空间。
- 较为复杂的布局还可以通过嵌套一个伸缩容器（flex container）来实现。
- 采用 Flex 布局的元素，称为 Flex 容器（flex container），简称"容器"。
- 它的所有子元素自动成为容器成员，称为 Flex 项目（flex item），简称"项目"。
- 常规布局是基于块和内联流方向，而 Flex 布局是基于 flex-flow 流可以很方便的用来做局中，能对不同屏幕大小自适应。
- 在布局上有了比以前更加灵活的空间。
-
- 具体：http://www.w3cplus.com/css3/flexbox-basics.html

- 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;
- }

- 一个满屏 品 字布局 如何设计?

- 简单的方式：
- 上面的 div 宽 100%,
- 下面的两个 div 分别宽 50%,
- 然后用 float 或者 inline 使其不换行即可

- css 多列等高如何实现？

- 利用 padding-bottom|margin-bottom 正负值相抵；
- 设置父容器设置超出隐藏（overflow:hidden），这样子父容器的高度就还是它里面的列没有设定 padding-bottom 时的高度，
- 当它里面的任 一列高度增加了，则父容器的高度被撑到里面最高那列的高度，
- 其他比这列矮的列会用它们的 padding-bottom 补偿这部分高度差。

- 经常遇到的浏览器的兼容性有哪些？原因，解决方法是什么，常用 hack 的技巧 ？

- * 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。
-
- * IE6 双边距 bug:块属性标签 float 后，又有横行的 margin 情况下，在 ie6 显示 margin 比设置的大。
-
- 浮动 ie 产生的双倍距离 #box{ float:left; width:10px; margin:0 0 0 100px;}
-
- 这种情况之下 IE 会产生 20px 的距离，解决方案是在 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)
-
- 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。
-
- css
- .bb{
- background-color:red;/*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*/
- }
-
-
- * 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。
-
- * IE 下,even 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。
-
- * Chrome 中文界面下默认会将小于 12px 的文本强制按照 12px 显示,
- 可通过加入 CSS 属性 -webkit-text-size-adjust: none; 解决。
-
- 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 了解决方法是改变 CSS 属性的排列顺序:
- L-V-H-A : a:link {} a:visited {} a:hover {} a:active {}

- li 与 li 之间有看不见的空白间隔是什么原因引起的？有什么解决办法？

- 行框的排列会受到中间空白（回车\空格）等的影响，因为空格也属于字符,这些空白也会被应用样式，占据空间，所以会有间隔，把字符大小设为 0，就没有空格了。

- 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。
-
- 最简单的初始化方法： * {padding: 0; margin: 0;} （强烈不建议）
-
- 淘宝的样式初始化代码：
- body, h1, h2, h3, h4, h5, h6, hr, p, blockquote, dl, dt, dd, ul, ol, li, pre, form, fieldset, legend, button, input, textarea, th, td { margin:0; padding:0; }
- body, button, input, select, textarea { font:12px/1.5tahoma, arial, \5b8b\4f53; }
- h1, h2, h3, h4, h5, h6{ font-size:100%; }
- address, cite, dfn, em, var { font-style:normal; }
- code, kbd, pre, samp { font-family:couriernew, courier, monospace; }
- small{ font-size:12px; }
- ul, ol { list-style:none; }
- a { text-decoration:none; }
- a:hover { text-decoration:underline; }
- sup { vertical-align:text-top; }
- sub{ vertical-align:text-bottom; }
- legend { color:#000; }
- fieldset, img { border:0; }
- button, input, select, textarea { font-size:100%; }
- table { border-collapse:collapse; border-spacing:0; }

- absolute 的 containing block(容器块)计算方式跟正常流有什么不同？

- 无论属于哪种，都要先找到其祖先元素中最近的 position 值不为 static 的元素，然后再判断：
- 1、若此元素为 inline 元素，则 containing block 为能够包含这个元素生成的第一个和最后一个 inline box 的 padding box (除 margin, border 外的区域) 的最小矩形；
- 2、否则,则由这个祖先元素的 padding box 构成。
- 如果都找不到，则为 initial containing block。
-
- 补充：
- 1. static(默认的)/relative：简单说就是它的父元素的内容框（即去掉 padding 的部分）
- 2. absolute: 向上找最近的定位为 absolute/relative 的元素
- 3. fixed: 它的 containing block 一律为根元素(html/body)，根元素也是 initial containing block

- CSS 里的 visibility 属性有个 collapse 属性值是干嘛用的？在不同浏览器下以后什么区别？

  对于普通元素 visibility:collapse;会将元素完全隐藏,不占据页面布局空间,与 display:none;表现相同. 如果目标元素为 table,visibility:collapse;将 table 隐藏,但是会占据页面布局空间. 仅在 Firefox 下起作用,IE 会显示元素,Chrome 会将元素隐藏,但是占据空间.

- position 跟 display、margin collapse、overflow、float 这些特性相互叠加后会怎么样？

  如果元素的 display 为 none,那么元素不被渲染,position,float 不起作用,如果元素拥有 position:absolute;或者 position:fixed;属性那么元素将为绝对定位,float 不起作用.如果元素 float 属性不是 none,元素会脱离文档流,根据 float 属性值来显示.有浮动,绝对定位,inline-block 属性的元素,margin 不会和垂直方向上的其他元素 margin 折叠.

- 对 BFC 规范(块级格式化上下文：block formatting context)的理解？

- （W3C CSS 2.1 规范中的一个概念,它是一个独立容器，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。）
- 一个页面是由很多个 Box 组成的,元素的类型和 display 属性,决定了这个 Box 的类型。
- 不同类型的 Box,会参与不同的 Formatting Context（决定如何渲染文档的容器）,因此 Box 内的元素会以不同的方式渲染,也就是说 BFC 内部的元素和外部的元素不会互相影响。

- css 定义的权重

- 以下是权重的规则：标签的权重为 1，class 的权重为 10，id 的权重为 100，以下例子是演示各种定义的权重值：
-
- /*权重为 1*/
- div{

```
    }
    /*权重为 10*/
    .class1{
    }
    /*权重为 100*/
    #id1{
    }
    /*权重为 100+1=101*/
    #id1 div{
    }
    /*权重为 10+1=11*/
    .class1 div{
    }
    /*权重为 10+10+1=21*/
    .class1 .class2 div{
    }

    如果权重相同，则最后定义的样式会起作用，但是应该避免这种情况出现
```

- 请解释一下为什么需要清除浮动？清除浮动的方式

  清除浮动是为了清除使用浮动元素产生的影响。浮动的元素，高度会塌陷，而高度的塌陷使我们页面后面的布局不能正常显示。

```
1、父级 div 定义 height；
2、父级 div 也一起浮动；
3、常规的使用一个 class；
        .clearfix:before, .clearfix:after {
            content: " ";
            display: table;
        }
        .clearfix:after {
            clear: both;
        }
        .clearfix {
            *zoom: 1;
        }

4、SASS 编译的时候，浮动元素的父级 div 定义伪类:after
        &:after,&:before{
            content: " ";
        visibility: hidden;
        display: block;
        height: 0;
        clear: both;
        }

解析原理：
1) display:block 使生成的元素以块级元素显示,占满剩余空间;
2) height:0 避免生成内容破坏原有布局的高度。
```

3) visibility:hidden 使生成的内容不可见，并允许可能被生成内容盖住的内容可以进行点击和交互;
4）通过 content:"."生成内容作为最后一个元素，至于 content 里面是点还是其他都是可以的，例如 oocss 里面就有经典的 content:".",有些版本可能 content 里面内容为空,一丝冰凉是不推荐这样做的,firefox 直到 7.0 content:”" 仍然会产生额外的空隙；
5）zoom：1 触发 IE hasLayout。

通过分析发现，除了 clear：both 用来闭合浮动的，其他代码无非都是为了隐藏掉 content 生成的内容，这也就是其他版本的闭合浮动为什么会有 font-size：0，line-height：0。

- 什么是外边距合并？

- 外边距合并指的是，当两个垂直外边距相遇时，它们将形成一个外边距。
- 合并后的外边距的高度等于两个发生合并的外边距的高度中的较大者。
- w3school 介绍网址： http://www.w3school.com.cn/css/css_margin_collapsing.asp

- zoom:1 的清除浮动原理?

- 清除浮动，触发 hasLayout；
- Zoom 属性是 IE 浏览器的专有属性，它可以设置或检索对象的缩放比例。解决 ie 下比较奇葩的 bug。
- 譬如外边距（margin）的重叠，浮动清除，触发 ie 的 haslayout 属性等。
-
- 来龙去脉大概如下：
- 当设置了 zoom 的值之后，所设置的元素就会就会扩大或者缩小，高度宽度就会重新计算了，这里一旦改变 zoom 值时其实也会发生重新渲染，运用这个原理，也就解决了 ie 下子元素浮动时候父元素不随着自动扩大的问题。
-
- Zoom 属是 IE 浏览器的专有属性，火狐和老版本的 webkit 核心的浏览器都不支持这个属性。然而，zoom 现在已经被逐步标准化，出现在 CSS 3.0 规范草案中。
-
- 目前非 ie 由于不支持这个属性，它们又是通过什么属性来实现元素的缩放呢？
- 可以通过 css3 里面的动画属性 scale 进行缩放。

- 移动端的布局用过媒体查询吗？

假设你现在正用一台显示设备来阅读这篇文章，同时你也想把它投影到屏幕上，或者打印出来， 而显示设备、屏幕投影和打印等这些媒介都有自己的特点，CSS 就是为文档提供在不同媒介上展示的适配方法

当媒体查询为真时，相关的样式表或样式规则会按照正常的级联规被应用。 当媒体查询返回假， 标签上带有媒体查询的样式表 仍将被下载 （只不过不会被应用）。

包含了一个媒体类型和至少一个使用 宽度、高度和颜色等媒体属性来限制样式表范围的表达式。 CSS3 加入的媒体查询使得无需修改内容便可以使样式应用于某些特定的设备范围。

<style> @media (min-width: 700px) and (orientation: landscape){ .sidebar { display: none; } } </style>

- 使用 CSS 预处理器吗？喜欢那个？

- SASS（SASS、LESS 没有本质区别，只因为团队前端都是用的 SASS）

- CSS 优化、提高性能的方法有哪些？

- 关键选择器（key selector）。选择器的最后面的部分为关键选择器（即用来匹配目标元素的部分）；
- 如果规则拥有 ID 选择器作为其关键选择器，则不要为规则增加标签。过滤掉无关的规则（这样样式系统就不会浪费时间去匹配它们了）；
- 提取项目的通用公有样式，增强可复用性，按模块编写组件；增强项目的协同开发性、可维护性和可扩展性;
- 使用预处理工具或构建工具（gulp 对 css 进行语法检查、自动补前缀、打包压缩、自动优雅降级）；

- 浏览器是怎样解析 CSS 选择器的？

- 样式系统从关键选择器开始匹配，然后左移查找规则选择器的祖先元素。
- 只要选择器的子树一直在工作，样式系统就会持续左移，直到和规则匹配，或者是因为不匹配而放弃该规则。

- 在网页中的应该使用奇数还是偶数的字体？为什么呢？

- margin 和 padding 分别适合什么场景使用？

- margin 是用来隔开元素与元素的间距；padding 是用来隔开元素与内容的间隔。
- margin 用于布局分开元素使元素与元素互不相干；
- padding 用于元素与内容之间的间隔，让内容（文字）与（包裹）元素之间有一段

- 抽离样式模块怎么写，说出思路，有无实践经验？[阿里航旅的面试题]

- 元素竖向的百分比设定是相对于容器的高度吗？

- 全屏滚动的原理是什么？用到了 CSS 的那些属性？

- 什么是响应式设计？响应式设计的基本原理是什么？如何兼容低版本的 IE？

- 视差滚动效果，如何给每页做不同的动画？（回到顶部，向下滑动要再次出现，和只出现一次分别怎么做？）

- ::before 和 :after 中双冒号和单冒号 有什么区别？解释一下这 2 个伪元素的作用。

- 单冒号(:)用于 CSS3 伪类，双冒号(::)用于 CSS3 伪元素。（伪元素由双冒号和伪元素名称组成）
- 双冒号是在当前规范中引入的，用于区分伪类和伪元素。不过浏览器需要同时支持旧的已经存在的伪元素写法，
- 比如:first-line、:first-letter、:before、:after 等，
- 而新的在 CSS3 中引入的伪元素则不允许再支持旧的单冒号的写法。
-
- 想让插入的内容出现在其它内容前，使用::before，否者，使用::after；
- 在代码顺序上，::after 生成的内容也比::before 生成的内容靠后。
- 如果按堆栈视角，::after 生成的内容会在::before 生成的内容之上

- 如何修改 chrome 记住密码后自动填充表单的黄色背景 ？

```
input:-webkit-autofill, textarea:-webkit-autofill, select:-webkit-autofill {
  background-color: rgb(250, 255, 189); /* #FAFFBD; */
  background-image: none;
  color: rgb(0, 0, 0);
}
```

- 你对 line-height 是如何理解的？

- 设置元素浮动后，该元素的 display 值是多少？

- 自动变成了 display:block

- 怎么让 Chrome 支持小于 12px 的文字？

- 1、用图片：如果是内容固定不变情况下，使用将小于 12px 文字内容切出做图片，这样不影响兼容也不影响美观。
- 2、使用 12px 及 12px 以上字体大小：为了兼容各大主流浏览器，建议设计美工图时候设置大于或等于 12px 的字体大小，如果是接单的这个时候就需要给客户讲解小于 12px 浏览器不兼容等事宜。
- 3、继续使用小于 12px 字体大小样式设置：如果不考虑 chrome 可以不用考虑兼容，同时在设置小于 12px 对象设置-webkit-text-size-adjust:none，做到最大兼容考虑。
- 4、使用 12px 以上字体：为了兼容、为了代码更简单 从新考虑权重下兼容性。

- 让页面里的字体变清晰，变细用 CSS 怎么做？

```
-webkit-font-smoothing: antialiased;
```

- font-style 属性可以让它赋值为“oblique” oblique 是什么意思？

- 倾斜的字体样式

- position:fixed;在 android 下无效怎么处理？

- fixed 的元素是相对整个页面固定位置的，你在屏幕上滑动只是在移动这个所谓的 viewport，
- 原来的网页还好好的在那，fixed 的内容也没有变过位置，
- 所以说并不是 iOS 不支持 fixed，只是 fixed 的元素不是相对手机屏幕固定的。
- <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0, user-scalable=no"/>

- 如果需要手动写动画，你认为最小时间间隔是多久，为什么？（阿里）

- 多数显示器默认频率是 60Hz，即 1 秒刷新 60 次，所以理论上最小间隔为 1/60＊1000ms = 16.7ms

- display:inline-block 什么时候会显示间隙？(携程)

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

- overflow: scroll 时不能平滑滚动的问题怎么处理？

- 有一个高度自适应的 div，里面有两个 div，一个高度 100px，希望另一个填满剩下的高度。

- png、jpg、gif 这些图片格式解释一下，分别什么时候用。有没有了解过 webp？

- 什么是 Cookie 隔离？（或者说：请求资源的时候不要让它带 cookie 怎么做）

- 如果静态文件都放在主域名下，那静态文件请求的时候都带有的 cookie 的数据提交给 server 的，非常浪费流量，
- 所以不如隔离开。
-
- 因为 cookie 有域的限制，因此不能跨域提交请求，故使用非主要域名的时候，请求头中就不会带有 cookie 数据，
- 这样可以降低请求头的大小，降低请求时间，从而达到降低整体请求延时的目的。
-
- 同时这种方式不会将 cookie 传入 Web Server，也减少了 Web Server 对 cookie 的处理分析环节，
- 提高了 webserver 的 http 请求的解析速度。

- style 标签写在 body 后与 body 前有什么区别？

- 什么是 CSS 预处理器 / 后处理器？

- - 预处理器例如：LESS、Sass、Stylus，用来预编译 Sass 或 less，增强了 css 代码的复用性，
- 还有层级、mixin、变量、循环、函数等，具有很方便的 UI 组件模块化开发能力，极大的提高工作效率。
-
- - 后处理器例如：PostCSS，通常被视为在完成的样式表中根据 CSS 规范处理 CSS，让其更有效；目前最常做的

- 是给 CSS 属性添加浏览器私有前缀，实现跨浏览器兼容性的问题。

JavaScript

- 介绍 js 的基本数据类型。

- Undefined、Null、Boolean、Number、String、
- ECMAScript 2015 新增:Symbol(创建后独一无二且不可变的数据类型)

- 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
-
- 参考：http://www.ibm.com/developerworks/cn/web/wa-objectsinjs-v1b/index.html

- 说几条写 JavaScript 的基本规范？

- 1.不要在同一行声明多个变量。
- 2.请使用 ===/!==来比较 true/false 或者数值
- 3.使用对象字面量替代 new Array 这种形式
- 4.不要使用全局函数。
- 5.Switch 语句必须带有 default 分支
- 6.函数不应该有时候有返回值，有时候没有返回值。
- 7.For 循环必须使用大括号
- 8.If 语句必须使用大括号
- 9.for-in 循环中的变量 应该使用 var 关键字明确限定作用域，从而避免作用域污染。

- 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， 如果没有的话，

- 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。
- function Func(){}
- Func.prototype.name = "Sean";
- Func.prototype.getInfo = function() {
- return this.name;
- }
- var person = new Func();//现在可以参考 var person = Object.create(oldObject);
- console.log(person.getInfo());//它拥有了 Func 的属性和方法
- //"Sean"
- console.log(Func.prototype);
- // Func { name="Sean", getInfo=function()}

- 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

- 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。

- 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 + ',';
                });
}
```

- 如何实现数组的随机排序？ 方法一：```javascript 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[i]; arr[i] = temp; } return arr; } console.log(randSort1(arr));

```
```
方法二：
```javascript
                    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：
```javascript
                    var arr = [1,2,3,4,5,6,7,8,9,10];
                    arr.sort(function(){
                            return Math.random() - 0.5;
                    })
                    console.log(arr);
```
```

- 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。
```javascript
            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
```

```
```

- JavaScript 继承的几种实现方式？
  - 参考：构造函数的继承，非构造函数的继承；
- 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。


 1、对象字面量的方式

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

 2、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...");
  }
  person.work();

 3、用 function 来模拟参构造函数来实现（用 this 关键字定义构造的上下文属性）

  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


 4、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();


  5、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();


  5、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");
     }
   var camry =new Car("凯美瑞",27);
   camry.sell();
```

- Javascript 作用链域?

```
 全局函数无法查看局部函数的内部细节，但局部函数可以查看其上层的函数细节，直至全局细节。
 当需要从局部函数查找某一属性或方法时，如果当前作用域没有找到，就会上溯到上层作用域查找，
 直至全局函数，这种组织形式就是作用域链。
```

- 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；
- 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 +')');

- 什么是 window 对象? 什么是 document 对象?

- window 对象是指浏览器打开的窗口。
- document 对象是 Documentd 对象（HTML 文档对象）的一个只读引用，window 对象的一个属性。

- null，undefined 的区别？

- null　　表示一个对象是“没有值”的值，也就是值为“空”；
- undefined　　表示一个变量声明了没有初始化(赋值)；
-
- undefined 不是一个有效的 JSON，而 null 是；
- undefined 的类型(typeof)是 undefined；
- null 的类型(typeof)是 object；
-
-
- Javascript 将未赋值的变量默认值设为 undefined；
- Javascript 从来不会将变量设为 null。它是用来让程序员表明某个用 var 声明的变量时没有值的。
-
- typeof undefined
- //"undefined"
- 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
-
- typeof null
- //"object"
- 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；
-
- 注意：
- 在验证 null 时，一定要使用 === ，因为 == 无法分别 null 和 undefined
- null == undefined // true
- null === undefined // false
-
- 再来一个例子：
-
- null
- Q：有张三这个人么？
- A：有！
- Q：张三有房子么？
- A：没有！
-

```
undefined
Q：有张三这个人么？
A：有！
Q: 张三有多少岁？
A: 不知道（没有被告诉）
```

参考阅读：[undefined 与 null 的区别]

- 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 = {
        // 页面加载完成后
        readyEvent : function(fn) {
                if (fn==null) {
                        fn=document;
                }
                var oldonload = window.onload;
                if (typeof window.onload != 'function') {
                        window.onload = fn;
                } else {
                        window.onload = function() {
                                oldonload();
                                fn();
                        };
                }
        },
        // 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
        removeEvent : function(element, type, handler) {
                if (element.removeEventListener) {
                        element.removeEventListener(type, handler, false);
                } else if (element.datachEvent) {
                        element.detachEvent('on' + type, handler);
                } 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;
        },
        // 获取 event 对象的引用，取到事件的所有信息，确保随时能使用 event；
        getEvent : function(e) {
                var ev = e || window.event;
                if (!ev) {
                        var c = this.getEvent.caller;
                        while (c) {
                                ev = c.arguments[0];
                                if (ev && Event == ev.constructor) {
                                        break;
                                }
                                c = c.caller;
                        }
                }
                return ev;
        }
 };
```

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

```
 parseInt() 函数能解析一个字符串，并返回一个整数，需要两个参数 (val, radix)，
 其中 radix 表示要解析的数字的基数。【该值介于 2 ~ 36 之间，并且字符串中的数字不能大于 radix 才能正确返回数字结果值】；
 但此处 map 传了 3 个 (element, index, array),我们重写 parseInt 函数测试一下是否符合上面的规则。

 function parseInt(str, radix) {
     return str+'-'+radix;
 };
```

```
var a=["1", "2", "3"];
a.map(parseInt);  // ["1-0", "2-1", "3-2"] 不能大于 radix

因为二进制里面，没有数字 3,导致出现超范围的 radix 赋值和不合法的进制解析，才会返回 NaN
所以["1", "2", "3"].map(parseInt) 答案也就是：[1, NaN, NaN]

详细解析：http://blog.csdn.net/justjavac/article/details/19473199
```

- 事件是？IE 与火狐的事件机制有什么区别？ 如何阻止冒泡？

```
1. 我们在网页中的某个操作（有的操作对应多个事件）。例如：当我们点击一个按钮就会产生一个事件。是可以被 JavaScript 侦测到的行为。
2. 事件处理机制：IE 是事件冒泡、Firefox 同时支持两种事件模型，也就是：捕获型事件和冒泡型事件；
3. ev.stopPropagation();（旧 ie 的方法 ev.cancelBubble = true;）
```

- 什么是闭包（closure），为什么要用它？

```
闭包是指有权访问另一个函数作用域中变量的函数，创建闭包的最常见的方式就是在一个函数内创建另一个函数，通过另一个函数访问这个函数的局部变量,利用闭包可以突破作用链域，将函数内部的变量和方法传递到外部。

闭包的特性：

1.函数内再嵌套函数
2.内部函数可以引用外层的参数和变量
3.参数和变量不会被垃圾回收机制回收

//li 节点的 onclick 事件都能正确的弹出当前被点击的 li 索引
 <ul id="testUL">
    <li> index = 0</li>
    <li> index = 1</li>
    <li> index = 2</li>
    <li> index = 3</li>
</ul>
<script type="text/javascript">
          var nodes = document.getElementsByTagName("li");
 for(i = 0;i<nodes.length;i+= 1){
     nodes[i].onclick = (function(i){
                return function() {
                   console.log(i);
                } //不用闭包的话，值每次都是 4
             })(i);
 }
</script>


执行 say667()后,say667()闭包内部变量会存在,而闭包内部函数的内部变量不会存在
```

- 使得 Javascript 的垃圾回收机制 GC 不会收回 say667()所占用的资源
- 因为 say667()的内部函数的执行需要依赖 say667()中的变量
- 这是对闭包作用的非常直白的描述
-
-
```
 function say667() {
// Local variable that ends up within closure
var num = 666;
var sayAlert = function() {
        alert(num);
}
num++;
return sayAlert;
}

var sayAlert = say667();
sayAlert()//执行结果应该弹出的 667
```

- 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。
- 默认支持的糟糕特性都会被禁用，比如不能用 with，也不能在意外的情况下给全局变量赋值;
- 全局变量的显示声明,函数必须声明在顶层，不允许在非函数代码块内声明函数,arguments.callee 也不允许使用；
- 消除代码运行的一些不安全之处，保证代码运行的安全,限制函数中的 arguments 修改，严格模式下的 eval 函数的行为和非严格模式的也不相同;
-
- 提高编译器效率，增加运行速度；
- 为未来新版本的 Javascript 标准化做铺垫。

- 如何判断一个对象是否属于某个类？

```
使用 instanceof （待完善）
 if(a instanceof Person){
     alert('yes');
 }
```

- 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 。

var obj  = {};
obj.__proto__ = Base.prototype;
```

- Base.call(obj);

- 用原生 JavaScript 的实现过什么功能吗？

- Javascript 中，有一个函数，执行时对象查找时，永远不会去查找原型，这个函数是？

- hasOwnProperty
-
- javaScript 中 hasOwnProperty 函数方法是返回一个布尔值，指出一个对象是否具有指定名称的属性。此方法无法检查该对象的原型链中是否具有该属性；该属性必须是对象本身的一个成员。
- 使用方法：
- object.hasOwnProperty(proName)
- 其中参数 object 是必选项。一个对象的实例。
- proName 是必选项。一个属性名称的字符串值。
-
- 如果 object 具有指定名称的属性，那么 JavaScript 中 hasOwnProperty 函数方法返回 true，反之则返回 false。

- JSON 的了解？

- JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。
- 它是基于 JavaScript 的一个子集。数据格式简单, 易于读写, 占用带宽小
- 如：{"age":"12", "name":"back"}
-
- JSON 字符串转换为 JSON 对象:
- 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);
- [].forEach.call($$("*"),function(a){a.style.outline="1px solid #"+(~~(Math.random()*(1<<24))).toString(16)}) 能解释一下这段代码的意思吗？

- 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

- Ajax 是什么? 如何创建一个 Ajax？

- 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

- Ajax 解决浏览器缓存问题？

- 1、在 ajax 发送请求前加上 anyAjaxObj.setRequestHeader("If-Modified-Since","0")。
-
- 2、在 ajax 发送请求前加上 anyAjaxObj.setRequestHeader("Cache-Control","no-cache")。
-
- 3、在 URL 后面加上一个随机数： "fresh=" + Math.random();。
-
- 4、在 URL 后面加上时间搓："nowtime=" + new Date().getTime();。
-
- 5、如果是使用 jQuery，直接这样就可以了 $.ajaxSetup({cache:false})。这样页面的所有 ajax 都会执行这条语句就是不需要保存缓存记录。

- 同步和异步的区别?

  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。

  （待完善）

- 如何解决跨域问题?

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

- 页面编码和被请求的资源编码如果不一致如何处理？

- 模块化开发怎么做？

  立即执行函数,不暴露私有成员

```
    var module1 = (function(){
            var _count = 0;
            var m1 = function(){
                //...
            };
            var m2 = function(){
                //...
            };
            return {
                m1 : m1,
                m2 : m2
            };
        })();
```

（待完善）

- AMD（Modules/Asynchronous-Definition）、CMD（Common Module Definition）规范区别？

  AMD 规范在这里：https://github.com/amdjs/amdjs-api/wiki/AMD
  CMD 规范在这里：https://github.com/seajs/seajs/issues/242

```
Asynchronous Module Definition，异步模块定义，所有的模块将被异步加载，模块加载不影响后面语句运行。所有依赖某些模块的语句均放置在回调函数中。

 区别：

    1. 对于依赖的模块，AMD 是提前执行，CMD 是延迟执行。不过 RequireJS 从 2.0 开始，也改成可以延迟执行（根据写法不同，处理方式不同）。CMD 推崇 as lazy as possible.
    2. CMD 推崇依赖就近，AMD 推崇依赖前置。看代码：

// CMD
define(function(require, exports, module) {
    var a = require('./a')
    a.doSomething()
    // 此处略去 100 行
    var b = require('./b') // 依赖可以就近书写
    b.doSomething()
    // ...
})

// AMD 默认推荐
define(['./a', './b'], function(a, b) { // 依赖必须一开始就写好
    a.doSomething()
    // 此处略去 100 行
    b.doSomething()
    // ...
})
```

- requireJS 的核心原理是什么？（如何动态加载的？如何避免多次加载的？如何 缓存的？）

- 参考：http://annn.me/how-to-realize-cmd-loader/

- JS 模块加载器的轮子怎么造，也就是如何实现一个模块加载器？

- 谈一谈你对 ECMAScript6 的了解？

- ECMAScript6 怎么写 class 么，为什么会出现 class 这种东西?

- 异步加载 JS 的方式有哪些？

- (1) defer，只支持 IE
-
- (2) async：
-
- (3) 创建 script，插入到 DOM 中，加载完毕后 callBack

- documen.write 和 innerHTML 的区别

- document.write 只能重绘整个页面
-
- innerHTML 可以重绘页面的一部分

- DOM 操作——怎样添加、移除、移动、复制、创建和查找节点?

- （1）创建新节点
- createDocumentFragment() //创建一个 DOM 片段
- createElement() //创建一个具体的元素
- createTextNode() //创建一个文本节点
- （2）添加、移除、替换、插入
- appendChild()
- removeChild()
- replaceChild()
- insertBefore() //在已有的子节点前插入一个新的子节点
- （3）查找
- getElementsByTagName() //通过标签名称
- getElementsByName() //通过元素的 Name 属性的值(IE 容错能力较强，会得到一个数组，其中包括 id 等于 name 值的)
- getElementById() //通过元素 Id，唯一性

- .call() 和 .apply() 的区别？

- 例子中用 add 来替换 sub，add.call(sub,3,1) == add(3,1) ，所以运行结果为：alert(4);
-
- 注意：js 中的函数其实是对象，函数名是对 Function 对象的引用。
-
- function add(a,b)

```
    {
        alert(a+b);
    }

    function sub(a,b)
    {
        alert(a-b);
    }

    add.call(sub,3,1);
```

- 数组和对象有哪些原生方法，列举一下？
- JS 怎么实现一个类。怎么实例化这个类
- JavaScript 中的作用域与变量声明提升？
- 如何编写高性能的 Javascript？
- 那些操作会造成内存泄漏？
- JQuery 的源码看过吗？能不能简单概况一下它的实现原理？
- jQuery.fn 的 init 方法返回的 this 指的是什么对象？为什么要返回 this？
- jquery 中如何将数组转化为 json 字符串，然后再转化回来？
- jQuery 的属性拷贝(extend)的实现原理是什么，如何实现深拷贝？
- jquery.extend 与 jquery.fn.extend 的区别？

```
 * jquery.extend 为 jquery 类添加类方法，可以理解为添加静态方法
 * jquery.fn.extend:
  源码中 jquery.fn = jquery.prototype，所以对 jquery.fn 的扩展，就是为 jquery 类添加成员函数
 使用：
 jquery.extend 扩展，需要通过 jquery 类来调用，而 jquery.fn.extend 扩展，所有 jquery 实例都可以直接调用。
```

- jQuery 的队列是如何实现的？队列可以用在哪些地方？
- 谈一下 Jquery 中的 bind(),live(),delegate(),on()的区别？
- JQuery 一个对象可以同时绑定多个事件，这是如何实现的？
- 是否知道自定义事件。jQuery 里的 fire 函数是什么意思，什么时候用？

- jQuery 是通过哪个方法和 Sizzle 选择器结合的？（jQuery.fn.find()进入 Sizzle）
- 针对 jQuery 性能的优化方法？
- Jquery 与 jQuery UI 有啥区别？

```
*jQuery 是一个 js 库，主要提供的功能是选择器，属性修改和事件绑定等等。

*jQuery UI 则是在 jQuery 的基础上，利用 jQuery 的扩展性，设计的插件。
 提供了一些常用的界面元素，诸如对话框、拖动行为、改变大小行为等等
```

- JQuery 的源码看过吗？能不能简单说一下它的实现原理？
- jquery 中如何将数组转化为 json 字符串，然后再转化回来？

jQuery 中没有提供这个功能，所以你需要先编写两个 jQuery 的扩展：

```
$.fn.stringifyArray = function(array) {
    return JSON.stringify(array)
}

$.fn.parseArray = function(array) {
    return JSON.parse(array)
}

然后调用：
$("").stringifyArray(array)
```

- jQuery 和 Zepto 的区别？各自的使用场景？
- 针对 jQuery 的优化方法？

```
*基于 Class 的选择性的性能相对于 Id 选择器开销很大，因为需遍历所有 DOM 元素。

*频繁操作的 DOM，先缓存起来再操作。用 Jquery 的链式调用更好。
 比如：var str=$("a").attr("href");

*for (var i = size; i < arr.length; i++) {}
 for 循环每一次循环都查找了数组 (arr) 的.length 属性，在开始循环的时候设置一个变量来存储这个数字，可以让循环跑得更快：
 for (var i = size, length = arr.length; i < length; i++) {}
```

- Zepto 的点透问题如何解决？
- jQueryUI 如何自定义组件?
- 需求：实现一个页面操作不会整页刷新的网站，并且能在浏览器前进、后退时正确响应。给出你的技术实现方案？

- 如何判断当前脚本运行在浏览器还是 node 环境中？（阿里）

```
  this === window ? 'browser' : 'node';

  通过判断 Global 对象是否为 window，如果不为 window，当前脚本没有运行在浏览器中
```

- 移动端最小触控区域是多大？
- jQuery 的 slideUp 动画，如果目标元素是被外部事件驱动, 当鼠标快速地连续触发外部元素事件, 动画会滞后的反复执行，该如何处理呢?

```
 jquery stop(): 如：$("#div").stop().animate({width:"100px"},100);
```

- 把 Script 标签 放在页面的最底部的 body 封闭之前 和封闭之后有什么区别？浏览器会如何解析它们？
- 移动端的点击事件的有延迟，时间是多久，为什么会有？怎么解决这个延时？（click 有 300ms 延迟,为了实现 safari 的双击事件的设计，浏览器要知道你是不是要双击操作。）
- 知道各种 JS 框架(Angular, Backbone, Ember, React, Meteor, Knockout...)么? 能讲出他们各自的优点和缺点么?
- Underscore 对哪些 JS 原生对象进行了扩展以及提供了哪些好用的函数方法？
- 解释 JavaScript 中的作用域与变量声明提升？
- 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。
 垃圾回收器定期扫描对象，并计算引用了每个对象的其他对象的数量。如果一个对象的引用数量为 0（没有其他对象引用过该对象），或对该对象的惟一引用是循环的，那么该对象的内存即可回收。

 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、控制台日志、循环（在两个对象彼此引用且彼此保留时，就会产生一个循环）
```

- JQuery 一个对象可以同时绑定多个事件，这是如何实现的？

```
 * 多个事件同一个函数：
  $("div").on("click mouseover", function(){});
 * 多个事件不同函数
  $("div").on({
          click: function(){},
          mouseover: function(){}
  });
```

- Node.js 的适用场景？
- (如果会用 node)知道 route, middleware, cluster, nodemon, pm2, server-side rendering 么?
- 解释一下 Backbone 的 MVC 实现方式？
- 什么是“前端路由”?什么时候适合使用“前端路由”? “前端路由”有哪些优点和缺点?
- 知道什么是 webkit 么? 知道怎么用浏览器的各种工具来调试和 debug 代码么?

```
  Chrome,Safari 浏览器内核。
```

- 如何测试前端代码么? 知道 BDD, TDD, Unit Test 么? 知道怎么测试你的前端工程么(mocha, sinon, jasmin, qUnit..)?
- 前端 templating(Mustache, underscore, handlebars)是干嘛的, 怎么用?
- 简述一下 Handlebars 的基本用法？
- 简述一下 Handlerbars 的对模板的基本处理流程， 如何编译的？如何缓存的？
- 用 js 实现千位分隔符?(来源：前端农民工，提示：正则+replace)

```
  参考：http://www.tuicool.com/articles/ArQZfui
  function commafy(num) {
      return num && num
          .toString()
          .replace/(\d)(?=(\d{3})+\.)/g, function($0, $1) {
              return $1 + ",";
          });
  }
  console.log(commafy(1234567.90)); //1,234,567.90
```

- 检测浏览器版本版本有哪些方式？

```
  功能检测、userAgent 特征检测

  比如：navigator.userAgent
  //"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_2) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/41.0.2272.101 Safari/537.36"
```

- What is a Polyfill?

```
  polyfill 是“在旧版浏览器上复制标准 API 的 JavaScript 补充”,可以动态地加载 JavaScript 代码或库，在不支持这些标准 API 的浏览器中模拟它们。
```

- 例如，geolocation（地理位置）polyfill 可以在 navigator 对象上添加全局的 geolocation 对象，还能添加 getCurrentPosition 函数以及“坐标”回调对象，
- 所有这些都是 W3C 地理位置 API 定义的对象和函数。因为 polyfill 模拟标准 API，所以能够以一种面向所有浏览器未来的方式针对这些 API 进行开发，
- 一旦对这些 API 的支持变成绝对大多数，则可以方便地去掉 polyfill，无需做任何额外工作。

- 做的项目中，有没有用过或自己实现一些 polyfill 方案（兼容性处理方案）？

- 比如： html5shiv、Geolocation、Placeholder

- 我们给一个 dom 同时绑定两个点击事件，一个用捕获，一个用冒泡。会执行几次事件，会先执行冒泡还是捕获？

- 使用 JS 实现获取文件扩展名？

```
function getFileExtension(filename) {
  return filename.slice((filename.lastIndexOf(".") - 1 >>> 0) + 2);
}
```

- String.lastIndexOf() 方法返回指定值（本例中的'.'）在调用该方法的字符串中最后出现的位置，如果没找到则返回 -1。
- 对于'filename'和'.hiddenfile'，lastIndexOf 的返回值分别为 0 和-1 无符号右移操作符(»>) 将-1 转换为 4294967295，将-2 转换为 4294967294，这个方法可以保证边缘情况时文件名不变。
- String.prototype.slice() 从上面计算的索引处提取文件的扩展名。如果索引比文件名的长度大，结果为""。

ECMAScript6 相关

- Object.is() 与原来的比较操作符“ ===”、“ ==”的区别？

- 两等号判等，会在比较时进行类型转换；
- 三等号判等(判断严格)，比较时不进行隐式类型转换，(类型不同则会返回 false）；
- Object.is 在三等号判等的基础上特别处理了 NaN 、-0 和 +0 ，保证 -0 和 +0 不再相同，
- 但 Object.is(NaN, NaN) 会返回 true.
- Object.is 应被认为有其特殊的用途，而不能用它认为它比其它的相等对比更宽松或严格。

前端框架相关

- react-router 路由系统的实现原理？

- React 中如何解决第三方类库的问题?

其他问题

- 原来公司工作流程是怎么样的，如何与其他人协作的？如何夸部门合作的？
- 你遇到过比较难的技术问题是？你是如何解决的？
- 设计模式 知道什么是 singleton, factory, strategy, decrator 么?
- 常使用的库有哪些？常用的前端开发工具？开发过什么应用或组件？
- 页面重构怎么操作？

- 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
- 深层次的网站重构应该考虑的方面
-
- 减少代码间的耦合
- 让代码保持弹性
- 严格按规范编写代码
- 设计可扩展的 API
- 代替旧有的框架、语言(如 VB)
- 增强用户体验
- 通常来说对于速度的优化也包含在重构中
-
- 压缩 JS、CSS、image 等前端资源(通常是由服务器来解决)
- 程序的性能优化(如数据读写)
- 采用 CDN 来加速资源加载
- 对于 JS DOM 的优化
- HTTP 服务器的文件缓存

- 列举 IE 与其他浏览器不一样的特性？

- 1、事件不同之处：
-
- 触发事件的元素被认为是目标（target）。而在 IE 中，目标包含在 event 对象的 srcElement 属性；
-

- 获取字符代码、如果按键代表一个字符（shift、ctrl、alt 除外），IE 的 keyCode 会返回字符代码（Unicode），DOM 中按键的代码和字符是分离的，要获取字符代码，需要使用 charCode 属性；
-
- 阻止某个事件的默认行为，IE 中阻止某个事件的默认行为，必须将 returnValue 属性设置为 false，Mozilla 中，需要调用 preventDefault() 方法；
-
- 停止事件冒泡，IE 中阻止事件进一步冒泡，需要设置 cancelBubble 为 true，Mozzilla 中，需要调用 stopPropagation()；

- 99%的网站都需要被重构是那本书上写的？

- 网站重构：应用 web 标准进行设计（第 2 版）

- 什么叫优雅降级和渐进增强？

- 优雅降级：Web 站点在所有新式浏览器中都能正常工作，如果用户使用的是老式浏览器，则代码会针对旧版本的 IE 进行降级处理了,使之在旧式浏览器上以某种形式降级体验却不至于完全不能用。
- 如：border-shadow
-
- 渐进增强：从被所有浏览器支持的基本功能开始，逐步地添加那些只有新版本浏览器才支持的功能,向页面增加不影响基础浏览器的额外样式和功能的。当浏览器支持时，它们会自动地呈现出来并发挥作用。
- 如：默认使用 flash 上传，但如果浏览器支持 HTML5 的文件上传功能，则使用 HTML5 实现更好的体验；

- 是否了解公钥加密和私钥加密。

- 一般情况下是指私钥用于对数据进行签名，公钥用于对签名进行验证;
- HTTP 网站在浏览器端用公钥加密敏感数据，然后在服务器端再用私钥解密。

- WEB 应用从服务器主动推送 Data 到客户端有那些方式？

- html5 提供的 Websocket
- 不可见的 iframe
- WebSocket 通过 Flash
- XHR 长时间连接
- XHR Multipart Streaming
- <script>标签的长时间连接(可跨域)

- 对 Node 的优点和缺点提出了自己的看法？

- *（优点）因为 Node 是基于事件驱动和无阻塞的，所以非常适合处理并发请求，
- 因此构建在 Node 上的代理服务器相比其他技术实现（如 Ruby）的服务器表现要好得多。
- 此外，与 Node 代理服务器交互的客户端代码是由 javascript 语言编写的，
- 因此客户端和服务器端都用同一种语言编写，这是非常美妙的事情。

-
- *（缺点）Node 是一个相对新的开源项目，所以不太稳定，它总是一直在变，
- 而且缺少足够多的第三方库支持。看起来，就像是 Ruby/Rails 当年的样子。

- 你有用过哪些前端性能优化的方法？

- （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。
- 对普通的网站有一个统一的思路，就是尽量向前端优化、减少数据库操作、减少磁盘 IO。向前端优化指的是，在不影响功能和体验的情况下，能在浏览器执行的不要在服务端执行，能在缓存服务器上直接返回的不要到应用服务器，程序能直接取得的结果不要到外部取得，本机内能取得的数据不要到远程取，内存能取到的不要到磁盘取，缓存中有的不要去数据库查询。减少数据库操作指减少更新次数、缓存结果减少查询次数、将数据库执行的操作尽可能的让你的程序完成（例如 join 查询），减少磁盘 IO 指尽量不使用文件系统作为缓存、减少读写文件次数等。程序优化永远要优化慢的部分，换语言是无法“优化”的。

- http 状态码有那些？分别代表是什么意思？

- 简单版
- [
- 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
- 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
- 301 Moved Permanently 请求的网页已永久移动到新位置。
- 302 Found 临时性重定向。
- 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。
-

- 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。
- 403 Forbidden 禁止访问。
- 404 Not Found 找不到如何与 URI 相匹配的资源。
-
- 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）。
-]
-
- 完整版
- 1**(信息类)：表示接收到请求并且继续处理
- 100——客户必须继续发出请求
- 101——客户要求服务器根据请求转换 HTTP 协议版本
-
- 2**(响应成功)：表示动作被成功接收、理解和接受
- 200——表明该请求被成功地完成，所请求的资源发送回客户端
- 201——提示知道新文件的 URL
- 202——接受和处理、但处理未完成
- 203——返回信息不确定或不完整
- 204——请求收到，但返回信息为空
- 205——服务器完成了请求，用户代理必须复位当前已经浏览过的文件
- 206——服务器已经完成了部分用户的 GET 请求
-
- 3**(重定向类)：为了完成指定的动作，必须接受进一步处理
- 300——请求的资源可在多处得到
- 301——本网页被永久性转移到另一个 URL
- 302——请求的网页被转移到一个新的地址，但客户访问仍继续通过原始 URL 地址，重定向，新的 URL 会在 response 中的 Location 中返回，浏览器将会使用新的 URL 发出新的 Request。
- 303——建议客户访问其他 URL 或访问方式
- 304——自从上次请求后，请求的网页未修改过，服务器返回此响应时，不会返回网页内容，代表上次的文档已经被缓存了，还可以继续使用
- 305——请求的资源必须从服务器指定的地址得到
- 306——前一版本 HTTP 中使用的代码，现行版本中不再使用
- 307——申明请求的资源临时性删除
-
- 4**(客户端错误类)：请求包含错误语法或不能正确执行
- 400——客户端请求有语法错误，不能被服务器所理解
- 401——请求未经授权，这个状态代码必须和 WWW-Authenticate 报头域一起使用
- HTTP 401.1 - 未授权：登录失败
- HTTP 401.2 - 未授权：服务器配置问题导致登录失败
- HTTP 401.3 - ACL 禁止访问资源
- HTTP 401.4 - 未授权：授权被筛选器拒绝
- HTTP 401.5 - 未授权：ISAPI 或 CGI 授权失败
- 402——保留有效 ChargeTo 头响应

- 403——禁止访问，服务器收到请求，但是拒绝提供服务
- HTTP 403.1 禁止访问：禁止可执行访问
- HTTP 403.2 - 禁止访问：禁止读访问
- HTTP 403.3 - 禁止访问：禁止写访问
- HTTP 403.4 - 禁止访问：要求 SSL
- HTTP 403.5 - 禁止访问：要求 SSL 128
- HTTP 403.6 - 禁止访问：IP 地址被拒绝
- HTTP 403.7 - 禁止访问：要求客户证书
- HTTP 403.8 - 禁止访问：禁止站点访问
- HTTP 403.9 - 禁止访问：连接的用户过多
- HTTP 403.10 - 禁止访问：配置无效
- HTTP 403.11 - 禁止访问：密码更改
- HTTP 403.12 - 禁止访问：映射器拒绝访问
- HTTP 403.13 - 禁止访问：客户证书已被吊销
- HTTP 403.15 - 禁止访问：客户访问许可过多
- HTTP 403.16 - 禁止访问：客户证书不可信或者无效
- HTTP 403.17 - 禁止访问：客户证书已经到期或者尚未生效
- 404——一个 404 错误表明可连接服务器，但服务器无法取得所请求的网页，请求资源不存在。eg：输入了错误的 URL
- 405——用户在 Request-Line 字段定义的方法不允许
- 406——根据用户发送的 Accept 拖，请求资源不可访问
- 407——类似 401，用户必须首先在代理服务器上得到授权
- 408——客户端没有在用户指定的饿时间内完成请求
- 409——对当前资源状态，请求不能完成
- 410——服务器上不再有此资源且无进一步的参考地址
- 411——服务器拒绝用户定义的 Content-Length 属性请求
- 412——一个或多个请求头字段在当前请求中错误
- 413——请求的资源大于服务器允许的大小
- 414——请求的资源 URL 长于服务器允许的长度
- 415——请求资源不支持请求项目格式
- 416——请求中包含 Range 请求头字段，在当前请求资源范围内没有 range 指示值，请求也不包含 If-Range 请求头字段
- 417——服务器不满足请求 Expect 头字段指定的期望值，如果是代理服务器，可能是下一级服务器不能满足请求长。
-
- 5**(服务端错误类)：服务器不能正确执行一个正确的请求
- HTTP 500 - 服务器遇到错误，无法完成请求
- HTTP 500.100 - 内部服务器错误 - ASP 错误
- HTTP 500-11 服务器关闭
- HTTP 500-12 应用程序重新启动
- HTTP 500-13 - 服务器太忙
- HTTP 500-14 - 应用程序无效
- HTTP 500-15 - 不允许请求 global.asa
- Error 501 - 未实现
- HTTP 502 - 网关错误
- HTTP 503：由于超载或停机维护，服务器目前无法使用，一段时间后可能恢复正常

- 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）

- 注：这题胜在区分度高，知识点覆盖广，再不懂的人，也能答出几句，
- 而高手可以根据自己擅长的领域自由发挥，从 URL 规范、HTTP 协议、DNS、CDN、数据库查询、
- 到浏览器流式解析、CSS 规则构建、layout、paint、onload/domready、JS 执行、JS API 绑定等等；
-
- 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。
-
- 简洁版：
- 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；
- 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。

- 部分地区用户反应网站很卡，请问有哪些可能性的原因，以及解决方法？

- 从打开 app 到刷新出内容，整个过程中都发生了什么，如果感觉慢，怎么定位问题，怎么解决?

- 除了前端以外还了解什么其它技术么？你最最厉害的技能是什么？

- 你用的得心应手用的熟练地编辑器&开发环境是什么样子？

- Sublime Text 3 + 相关插件编写前端代码
- Google chrome 、Mozilla Firefox 浏览器 +firebug 兼容测试和预览页面 UI、动画效果和交互功能
- Node.js+Gulp

- git 用于版本控制和 Code Review

- 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？

- 前端是最贴近用户的程序员，比后端、数据库、产品经理、运营、安全都近。
- 1、实现界面交互
- 2、提升用户体验
- 3、有了 Node.js，前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，
-
- 参与项目，快速高质量完成实现效果图，精确到 1px；
-
- 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；
-
- 处理 hack，兼容、写出优美的代码格式；
-
- 针对服务器的优化、拥抱最新前端技术。

- 你怎么看待 Web App 、hybrid App、Native App？

- 你移动端前端开发的理解？（和 Web 前端开发的主要区别是什么？）

- 你对加班的看法？

- 加班就像借钱，原则应当是------救急不救穷

- 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

- 如何设计突发大规模并发架构？

- 当团队人手不足，把功能代码写完已经需要加班的情况下，你会做前端代码的测试吗？
- 说说最近最流行的一些东西吧？常去哪些网站？
- ES6\WebAssembly\Node\MVVM\Web Components\React\React Native\Webpack 组件化
- 知道什么是 SEO 并且怎么优化么? 知道各种 meta data 的含义么?
- 移动端（Android IOS）怎么做好用户体验?
- 清晰的视觉纵线、
- 信息的分组、极致的减法、
- 利用选择代替输入、
- 标签及文字的排布方式、
- 依靠明文确认密码、
- 合理的键盘利用、
- 简单描述一下你做过的移动 APP 项目研发流程？
- 你在现在的团队处于什么样的角色，起到了什么明显的作用？
- 你认为怎样才是全端工程师（Full Stack developer）？
- 介绍一个你最得意的作品吧？
- 你有自己的技术博客吗，用了哪些技术？
- 对前端安全有什么看法？
- 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？
- 项目中遇到国哪些印象深刻的技术难题，具体是什么问题，怎么解决？。
- 最近在学什么东西？
- 你的优点是什么？缺点是什么？
- 如何管理前端团队?
- 最近在学什么？能谈谈你未来 3，5 年给自己的规划吗？

前端学习网站推荐

```
1. 极客标签：      http://www.gbtags.com/

2. 码农周刊：      http://weekly.manong.io/issues/

3. 前端周刊：      http://www.feweekly.com/issues

4. 慕课网：        http://www.imooc.com/

5. div.io：                http://div.io

6. Hacker News：  https://news.ycombinator.com/news

7. InfoQ：        http://www.infoq.com/

8. w3cplus：      http://www.w3cplus.com/

9. Stack Overflow：  http://stackoverflow.com/

10.w3school：     http://www.w3school.com.cn/

11.mozilla：      https://developer.mozilla.org/zh-CN/docs/Web/JavaScript
```