

# 前端面试题汇总

一、HTML 和 CSS.....	19
1、你做的页面在哪些浏览器测试过？这些浏览器的内核分别是什么？.....	19
2、每个 HTML 文件里开头都有个很重要的东西，Doctype，知道这是干什么的吗？ ....	19
3、Quirks 模式是什么？它和 Standards 模式有什么区别.....	19
4、div+css 的布局较 table 布局有什么优点？ .....	20
5、img 的 alt 与 title 有何异同？ strong 与 em 的异同？ .....	20
6、你能描述一下渐进增强和优雅降级之间的不同吗？.....	20
7、为什么利用多个域名来存储网站资源会更有效？ .....	21
8、请谈一下你对网页标准和标准制定机构重要性的理解。 .....	21
9、请描述一下 cookies，sessionStorage 和 localStorage 的区别？ .....	21
10、简述一下 src 与 href 的区别。 .....	22
11、知道的网页制作会用到的图片格式有哪些？ .....	22
12、知道什么是微格式吗？谈谈理解。在前端构建中应该考虑微格式吗？ .....	23
13、在 css/js 代码上线之后开发人员经常会优化性能，从用户刷新网页开始，一次 js 请求一般情况下有哪些地方会有缓存处理？ .....	23
14、一个页面上有大量的图片（大型电商网站），加载很慢，你有哪些方法优化这些图片的加载，给用户更好的体验。 .....	23
15、你如何理解 HTML 结构的语义化？ .....	23
16、谈谈以前端角度出发做好 SEO 需要考虑什么？ .....	24
17、有哪项方式可以对一个 DOM 设置它的 CSS 样式？ .....	26
18、CSS 都有哪些选择器？ .....	26
19、CSS 中可以通过哪些属性定义，使得一个 DOM 元素不显示在浏览器可视范围内？ .....	27
20、超链接访问过后 hover 样式就不出现的问题是什么？如何解决？ .....	27
21、什么是 Css Hack？ ie6, 7, 8 的 hack 分别是什么？ .....	27
22、请用 Css 写一个简单的幻灯片效果页面.....	28
24、行内元素和块级元素的具体区别是什么？行内元素的 padding 和 margin 可设置吗？ .....	29
25、什么是外边距重叠？重叠的结果是什么？ .....	29
26、rgba() 和 opacity 的透明效果有什么不同？ .....	29
27、css 中可以让文字在垂直和水平方向上重叠的两个属性是什么？ .....	30
28、如何垂直居中一个浮动元素？ .....	30
29、px 和 em 的区别。 .....	31
30、描述一个” reset” 的 CSS 文件并如何使用它。知道 normalize.css 吗？你了解他们的不同之处？ .....	31
31、Sass、LESS 是什么？大家为什么要使用他们？ .....	31
32、display:none 与 visibility:hidden 的区别是什么？ .....	32
34、CSS 中 link 和@import 的区别是： .....	32

35、简介盒子模型： .....	32
36、为什么要初始化样式？ .....	33
37、BFC 是什么？.....	33
38、html 语义化是什么？ .....	33
39、Doctype 的作用？严格模式与混杂模式的区别？ .....	34
40、IE 的双边距 BUG：块级元素 float 后设置横向 margin，ie6 显示的 margin 比设置的较大。解决：加入_display: inline.....	34
41、HTML 与 XHTML——二者有什么区别？ .....	34
42、html 常见兼容性问题？ .....	34
43、对 WEB 标准以及 W3C 的理解与认识.....	35
44、行内元素有哪些？块级元素有哪些？CSS 的盒模型？.....	35
45、前端页面有哪三层构成，分别是什么？作用是什么？.....	35
46、Doctype 作用？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？35	
47、行内元素有哪些？块级元素有哪些？空(void)元素有那些？ .....	35
48、CSS 的盒子模型？ .....	36
49、CSS 选择符有哪些？哪些属性可以继承？优先级算法如何计算？ CSS3 新增伪类有那些？ .....	36
50、如何居中 div, 如何居中一个浮动元素？.....	36
51、浏览器的内核分别是什么？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？ .....	37
52、列出 display 的值，说明他们的作用。position 的值， relative 和 absolute 定位原点是？ .....	38
53、absolute 的 containing block 计算方式跟正常流有什么不同？ .....	38
54、position 跟 display、margin collapse、overflow、float 这些特性相互叠加后会怎么样？ .....	38
55、对 WEB 标准以及 W3C 的理解与认识.....	38
56、css 的基本语句构成是？.....	39
57、浏览器标准模式和怪异模式之间的区别是什么？.....	39
58、CSS 中可以通过哪些属性定义，使得一个 DOM 元素不显示在浏览器可视范围内？ 39	
59、超链接访问过后 hover 样式就不出现的问题是什么？如何解决？ .....	39
60、什么是 Css Hack？ ie6, 7, 8 的 hack 分别是什么？ .....	39
62、请用 Css 写一个简单的幻灯片效果页面.....	40
63、行内元素和块级元素的具体区别是什么？行内元素的 padding 和 margin 可设置吗？ .....	41
64、什么是外边距重叠？重叠的结果是什么？ .....	41
65、rgba() 和 opacity 的透明效果有什么不同？ .....	41
66、css 中可以让文字在垂直和水平方向上重叠的两个属性是什么？ .....	42
67、如何垂直居中一个浮动元素？ .....	42

68、描述一个“reset”的 CSS 文件并如何使用它。知道 normalize.css 吗？你了解他们的不同之处？ .....	43
69、说 display 属性有哪些？可以做什么？ .....	43
70、哪些 css 属性可以继承？ .....	43
71、css 优先级算法如何计算？ .....	43
72、b 标签和 strong 标签, i 标签和 em 标签的区别？ .....	43
73、有那些行内元素、有哪些块级元素、盒模型？ .....	44
74、有哪些选择符，优先级的计算公式是什么？ 行内样式和 !important 哪个优先级高？ .....	45
75. 我想让行内元素跟上面的元素距离 10px，加 margin-top 和 padding-top 可以吗？ .....	45
76. CSS 的盒模型由什么组成？ .....	45
77、. 说说 display 属性有哪些？可以做什么？ .....	45
78、哪些 css 属性可以继承？ .....	45
79、css 优先级算法如何计算？ .....	45
80、text-align:center 和 line-height 有什么区别？ .....	46
81、前端页面由哪三层构成，分别是什么？作用是什么？ .....	46
82、写一个表格以及对应的 CSS, 使表格奇数行为白色背景，偶数行为灰色，鼠标一上去为黄色背景。 .....	46
二、JS 基础 .....	46
1、javascript 的 typeof 返回哪些数据类型 .....	46
2、例举 3 种强制类型转换和 2 种隐式类型转换？ .....	46
3、split() join() 的区别 .....	46
4、数组方法 pop() push() unshift() shift() .....	47
5、事件绑定和普通事件有什么区别 .....	47
6、IE 和 DOM 事件流的区别 .....	47
7、IE 和标准下有哪些兼容性的写法 .....	47
8、call 和 apply 的区别 .....	48
9、b 继承 a 的方法 .....	48
10、JavaScript this 指针、闭包、作用域 .....	48
11、事件委托是什么 .....	48
12、闭包是什么，有什么特性，对页面有什么影响 .....	48
13、如何阻止事件冒泡和默认事件 .....	48
14、添加 删除 替换 插入到某个接点的方法 .....	49
15、javascript 的本地对象，内置对象和宿主对象 .....	49
16、document load 和 document ready 的区别 .....	49
17、“==” 和 “===” 的不同 .....	49
18、javascript 的同源策略 .....	50
19、编写一个数组去重的方法 .....	50

20、JavaScript 是一门什么样的语言，它有哪些特点？ .....	50
21、JavaScript 的数据类型都有什么？ .....	51
22、已知 ID 的 Input 输入框，希望获取这个输入框的输入值，怎么做？（不使用第三方框架）.....	52
23、希望获取到页面中所有的 checkbox 怎么做？（不使用第三方框架）.....	52
24、设置一个已知 ID 的 DIV 的 html 内容为 xxxx，字体颜色设置为黑色（不使用第三方框架）.....	52
25、当一个 DOM 节点被点击时候，我们希望能够执行一个函数，应该怎么做？ .....	52
26、看下列代码输出为何？解释原因。 .....	53
27、看下列代码，输出什么？解释原因。 .....	53
28、看下列代码，输出什么？解释原因。 .....	53
29、看代码给答案。 .....	54
30、已知数组 var stringArray = [“This”， “is”， “Baidu”， “Campus”， Alert 出” This is Baidu Campus”。 .....	54
31、var numberArray = [3,6,2,4,1,5];（考察基础 API） .....	55
32、输出今天的日期，以 YYYY-MM-DD 的方式，比如今天是 2014 年 9 月 26 日，则输出 2014-09-26.....	55
33、将字符串” <tr><td>{\$id}</td><td>{\$name}</td></tr>” 中的 {\$id} 替换成 10， {\$name} 替换成 Tony（使用正则表达式） .....	56
34、为了保证页面输出安全，我们经常需要对一些特殊的字符进行转义，请写一个函数 escapeHtml，将<， >， &，“进行转义.....	56
35、foo = foo  bar，这行代码是什么意思？为什么要这样写？ .....	56
36、看下列代码，将会输出什么？(变量声明提升).....	57
37、用 js 实现随机选取 10 - 100 之间的 10 个数字，存入一个数组，并排序。 .....	57
38、把两个数组合并，并删除第二个元素。 .....	57
39、怎样添加、移除、移动、复制、创建和查找节点（原生 JS，实在基础，没细写每一步） .....	58
40、有这样一个 URL：http://item.taobao.com/item.htm?a=1&b=2&c=&d=xxx&e，请写一段 JS 程序提取 URL 中的各个 GET 参数（参数名和参数个数不确定），将其按 key-value 形式返回到一个 json 结构中，如 {a: ' 1 '， b: ' 2 '， c: ”， d: ' xxx'， e:undefined}。 .....	58
41、正则表达式构造函数 var reg=new RegExp(“xxx”)与正则表达式字面量 var reg=// 有什么不同？匹配邮箱的正则表达式？ .....	59
42、写一个 function，清除字符串前后的空格。（兼容所有浏览器） .....	59
43、Javascript 中 callee 和 caller 的作用？ .....	60
44、Javascript 中，以下哪条语句一定会产生运行错误？ 答案( B ).....	61
45、以下两个变量 a 和 b，a+b 的哪个结果是 NaN？ 答案( C ).....	61
46、var a=10; b=20; c=4; ++b+c+a++ 以下哪个结果是正确的？答案( B ).....	61

47、下面的 JavaScript 语句中，（ D ）实现检索当前页面中的表单元素中的所有文本框，并将它们全部清空.....	61
48、要将页面的状态栏中显示“已经选中该文本框”，下列 JavaScript 语句正确的是（ A ） .....	62
49、以下哪条语句会产生运行错误：（A）A.var obj = (); .var obj = []; Cvar obj = {}; D.ar obj = //;.....	62
50、以下哪个单词不属于 javascript 保留字：（B）A.with .parent Cclass D.oid.....	62
51、请选择结果为真的表达式：（C）A.null instanceof Object .null === undefined Cnull == undefined D.aN == NaN.....	62
52、Javascript 中，如果已知 HTML 页面中的某标签对象的 id="username"，用 ____document.getElementById( 'username' )____ 方法获得该标签对象。 .....	62
53、typeof 运算符返回值中有一个跟 javascript 数据类型不一致，它是 _____"function"_____。 .....	62
54、定义了一个变量，但没有为该变量赋值，如果 alert 该变量，javascript 弹出的对话框中显示__undefined__。 .....	62
55、分析代码，得出正确的结果。 .....	62
56、写出函数 DateDemo 的返回结果，系统时间假定为今天.....	63
57、写出程序运行的结果？ .....	64
58、阅读以下代码，请分析出结果： .....	64
59、补充按钮事件的函数，确认用户是否退出当前页面，确认之后关闭窗口； <html>64	
60、写出简单描述 html 标签（不带属性的开始标签和结束标签）的正则表达式，并将以下字符串中的 html 标签去除掉.....	65
61、完成 foo()函数的内容，要求能够弹出对话框提示当前选中的是第几个单选框。 65	
62、完成函数 showImg()，要求能够动态根据下拉列表的选项变化，更新图片的显示66	
63、截取字符串 abcdefg 的 efg.....	66
64、列举浏览器对象模型 BOM 里常用的至少 4 个对象，并列举 window 对象的常用方法至少 5 个.....	67
65、简述列举文档对象模型 DOM 里 document 的常用的查找访问节点的方法并做简单说明.....	67
66、希望获取到页面中所有的 checkbox 怎么做？（不使用第三方框架）.....	67
67、JavaScript 的数据类型都有什么？ .....	67

68、javascript 中有哪几种数据类型，分别写出中文和英文。 .....	68
69、javascript 中==和===的区别是什么？举例说明。 .....	69
70、简述创建函数的几种方式.....	69
71、Javascript 如何实现继承？ .....	69
72、Javascript 创建对象的几种方式？ .....	69
73、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？ .....	69
74、iframe 的优缺点？ .....	69
75、请你谈谈 Cookie 的弊端？ .....	70
76、DOM 操作——怎样添加、移除、移动、复制、创建和查找节点。 .....	70
77、js 延迟加载的方式有哪些？ .....	70
78、document.write 和 innerHTML 的区别？ .....	70
79、哪些操作会造成内存泄漏？ .....	71
80、javascript 的 typeof 返回哪些数据类型?.....	71
81、split() join() 的区别.....	71
82、数组方法 pop() push() unshift() shift() 各表示什么意思？ .....	71
83、判断一个字符串中出现次数最多的字符，统计这个次数.....	71
84、javascript 的 typeof 返回哪些数据类型.....	72
85、例举 3 种强制类型转换和 2 种隐式类型转换?.....	72
86、split() join() 的区别.....	72
87、数组方法 pop() push() unshift() shift() .....	72
88、事件绑定和普通事件有什么区别.....	<b>错误！未定义书签。</b>
89、IE 和 DOM 事件流的区别.....	72
90、IE 和标准下有哪些兼容性的写法.....	72
91、call 和 apply 的区别.....	72
92、b 继承 a 的方法.....	<b>错误！未定义书签。</b>
93、写一个获取非行间样式的函数.....	73
94、事件委托是什么.....	<b>错误！未定义书签。</b>
95、闭包是什么，有什么特性，对页面有什么影响.....	73
96、解释 jsonp 的原理，以及为什么不是真正的 ajax.....	73
97、javascript 的本地对象，内置对象和宿主对象.....	73
98、document load 和 document ready 的区别.....	73
99、字符串反转，如将 '12345678' 变成 '87654321' .....	74
100、将数字 12345678 转化成 RMB 形式 如： 12,345,678 .....	74
101、生成 5 个不同的随机数； .....	74
102、去掉数组中重复的数字 方法一； .....	74
103、阶乘函数； .....	75
104、window.location.search() 返回的是什麼？ .....	76

105、window.location.hash 返回的是什么？	76
106、window.location.reload() 作用？	76
107、阻止冒泡函数	76
108、什么是闭包？ 写一个简单的闭包？；	77
109、javascript 中的垃圾回收机制？	77
110、看题作答：	77
111、下面输出多少？	78
112、再来一个	78
113、	78
114、	79
115、JS 的继承性	80
116、精度问题：JS 精度不能精确到 0.1 所以。。。。同时存在于值和差值中	80
117、加减运算	80
118、什么是同源策略？	81
119、call 和 apply 的区别是什么？	81
120、为什么不能定义 1px 左右的 div 容器？	81
121、结果是什么？	81
122、输出结果	81
123、计算字符串字节数：	82
124、结果是：	82
125、声明对象，添加属性，输出属性	82
126、匹配输入的字符：第一个必须是字母或下划线开头，长度 5-20	82
127、检测变量类型	83
128、如何在 HTML 中添加事件，几种方法？	83
129、BOM 对象有哪些，列举 window 对象？	83
130、请问代码实现 outerHTML	83
131、JS 中的简单继承 call 方法！	84
132、bind(), live(), delegate() 的区别	85
133、typeof 的返回类型有哪些？	85
134、简述 link 和 import 的区别？	86
135、window.onload 和 document.ready 的区别？	86
136、解析 URL 成一个对象？	86
137、看下列代码输出什么？	86
138、看下列代码,输出什么？	87
139、已知数组 var stringArray = ["This", "is", "Baidu", "Campus"], Alert 出 "This is Baidu Campus"。	87
140、已知有字符串 foo="get-element-by-id", 写一个 function 将其转化成驼峰表示法 getElementById。	87

141、怎样添加、移除、移动、复制、创建和查找节点.....	87
142、原生 JS 的 window.onload 与 JQuery 的\$(document).ready(function() {})有什么 不同? .....	88
143、你如何优化自己的代码? .....	88
144、请描述出下列代码运行的结果.....	88
145、需要将变量 e 的值修改为 “a+b+c+d”, 请写出对应的代码.....	88
146、怎样实现两栏等高? .....	89
147、使用 js 实现这样的效果: 在文本域里输入文字时, 当按下 enter 键时不换行, 而 是替换成 “{{enter}}”, (只需要考虑在行尾按下 enter 键的情况). ....	89
148、以下代码中 end 字符串什么时候输出.....	89
149、specify( ‘hello,world’ )//=>’ h,e,l,l,o,w,o,r,l,d’ 实现 specify 函数..	89
150、请将一个 URL 的 search 部分参数与值转换成一个 json 对象.....	89
151、请用原生 js 实现 jquery 的 get\post 功能, 以及跨域情况下.....	90
152、请简要描述 web 前端性能需要考虑哪方面, 你的优化思路是什么? .....	90
153、简述 readonly 与 disabled 的区别.....	91
154、判断一个字符串出现次数最多的字符, 统计这个次数并输出.....	91
155、编写一个方法, 去掉一个数组的重复元素.....	91
156、写出 3 个使用 this 的典型应用.....	91
157、请尽可能详尽的解释 ajax 的工作原理.....	91
158、为什么扩展 javascript 内置对象不是好的做法? .....	91
159、请解释一下 javascript 的同源策略.....	91
160、什么是三元表达式? “三元” 表示什么意思? .....	91
161、浏览器标准模式和怪异模式之间的区别是什么? .....	91
162、如果设计中使用了非标准的字体, 你该如何去实现? .....	92
163、用 css 分别实现某个 div 元素上下居中和左右居中.....	92
164、modulo(12,5)//2 实现满足这个结果的 modulo 函数.....	92
165、HTTP 协议中, GET 和 POST 有什么区别? 分别适用什么场景 ? .....	92
166、HTTP 状态消息 200 302 304 403 404 500 分别表示什么.....	93
167、HTTP 协议中, header 信息里面, 怎么控制页面失效时间 (last-modified,cache-control,Expires 分别代表什么) .....	93
168、HTTP 雷峰议目前常用的有哪几个? KEEPALIVE 从哪个版本开始出现的? .....	93
169、业界常用的优化 WEB 页面加载速度的方法(可以分别从页面元素展现, 请求连接, css, js, 服务器等方面介绍) .....	93
170、列举常用的 web 页面开发, 调试以及优化工具.....	93
171、解释什么是 sql 注入, xss 漏洞.....	94
172、如何判断一个 js 变量是数组类型.....	94
173、请列举 js 数组类型中的常用方法.....	94
174、FF 与 IE 中如何阻止事件冒泡, 如何获取事件对象, 以及如何获取触发事件的元	



素.....	94
175、列举常用的 js 框架以及分别适用的领域.....	94
176、js 中如何实现一个 map.....	95
177、js 可否实现面向对象编程，如果可以如何实现 js 对象的继承.....	96
178、约瑟夫环—已知 n 个人（以编号 1，2，3...分别表示）围坐在一张圆桌周围。从编号为 k 的人开始报数，数到 m 的那个人出列；他的下一个人又从 1 开始报数，数到 m 的那个人又出列；依此规律重复下去，直到圆桌周围的人全部出列。.....	96
179、有 1 到 10w 这个 10w 个数，去除 2 个并打乱次序，如何找出那两个数？.....	96
180、如何获取对象 a 拥有的所有属性（可枚举的、不可枚举的，不包括继承来的属性）.....	96
181、有下面这样一段 HTML 结构，使用 css 实现这样的效果：.....	96
182、下面这段代码想要循环输出结果 01234，请问输出结果是否正确，如果不正确，请说明为什么，并修改循环内的代码使其输出正确结果.....	97
183、解释下这个 css 选择器什么发生什么？.....	97
184、JavaScript 以下哪条语句会产生运行错误.....	97
185、以下哪些是 javascript 的全局函数：（ABC）.....	97
186、关于 IE 的 window 对象表述正确的有：（ACD）.....	97
187、描述错误的是.....	98
188、关于 link 和@import 的区别正确的是 A.....	98
189、下面正确的是 A.....	98
188、错误的是.....	99
189、不用任何插件，如何实现一个 tab 栏切换？.....	99
190、基本数据类型的专业术语以及单词拼写.....	99
191、变量的命名规范以及命名推荐.....	99
192、三种弹窗的单词以及三种弹窗的功能.....	99
193、console.log( 8   1 ); 输出值是多少？.....	99
194、只允许使用 + - * / 和 Math.*，求一个函数 y = f(x, a, b);当 x > 100 时返回 a 的值，否则返回 b 的值，不能使用 if else 等条件语句，也不能使用 ,?:, 数组。.....	100
195、JavaScriptalert(0.4*0.2);结果是多少？和你预期的一样吗？如果不一样该如何处理？.....	100
196、一个 div，有几种方式得到这个 div 的 jQuery 对象？<div class='aabbcc' id='nodesView'></div>想直接获取这个 div 的 dom 对象，如何获取？dom 对象如何转化为 jQuery 对象？.....	100
197、主流浏览器内核.....	100
198、如何显示/隐藏一个 dom 元素？请用原生的 JavaScript 方法实现.....	100
199、JavaScript 有哪几种数据类型.....	101
200、jQuery 框架中\$.ajax()的常用参数有哪些？写一个 post 请求并带有发送数据和	

返回数据的样例.....	101
201、JavaScript 数据元素添加、删除、排序等方法有哪些? .....	103
202、如何添加 html 元素的事件, 有几种方法? 请列举.....	104
203、JavaScript 的循环语句有哪些? .....	104
204、作用域-编译期执行期以及全局局部作用域问题.....	104
205、闭包: 下面这个 ul, 如何点击每一列的时候 alert 其 index? .....	104
206、列出 3 条以上 ff 和 IE 的脚本兼容问题.....	106
207、列举可以哪些方面对前端开发进行优化.....	106
208、至少列出一种 JavaScript 继承的实现方式.....	106
209、如现在有一个效果, 有显示用户头像、用户昵称、用户其他信息; 当用户鼠标移到头像上时, 会弹出用户的所有信息; 如果是你, 你会如何实现这个功能, 请用代码实现? .....	106
210、call 与 apply 有什么作用? 又有什么什么区别? 用 callee 属性实现函数递归? .....	106
211、用正则表达式, 写出由字母开头, 其余由数字、字母、下划线组成的 6~30 的字符串? .....	107
212、列举浏览器对象模型 BOM 里常用的至少 4 个对象, 并列举 window 对象的常用方法至少 5 个 (10 分) .....	107
213、Javascript 中 callee 和 caller 的作用? .....	107
214、对于 apply 和 call 两者在作用上是相同的, 即是调用一个对象的一个方法, 以另一个对象替换当前对象。将一个函数的对象上下文从初始的上下文改变为由 thisObj 指定的新对象。.....	107
215、在 Javascript 中什么是伪数组? 如何将伪数组转化为标准数组? .....	107
216、写一个函数可以计算 sum(5, 0, -5); 输出 0; sum(1, 2, 3, 4); 输出 10;.....	108
217、事件代理怎么实现? .....	108
218、《正则》写出正确的正则表达式匹配固话号, 区号 3-4 位, 第一位为 0, 中横线, 7-8 位数字, 中横线, 3-4 位分机号格式的固话号.....	108
219、《算法》 一下 A,B 可任选一题作答, 两题全答加分.....	109
220、请写出一张图片的 HTML 代码, 已知道图片地址为 “images/abc. jpg”, 宽 100px, 高 50px.....	109
221、请写一个正则表达式: 要求最短 6 位数, 最长 20 位, 阿拉伯数和英文字母 (不区分大小写) 组成.....	109
222、统计 1 到 400 亿之间的自然数中含有多少个 1? 比如 1-21 中, 有 1、10、11、21 这四个自然数有 5 个 1.....	109
223、删除与某个字符相邻且相同的字符, 比如 fdaffdaaklfjklja 字符串处理之后成为 “fdafdaklfjklja” .....	109
224、请写出三种以上的 Firefox 有但, InternetExplorer 没有的属性活函数.....	109
225、请写出一个程序, 在页面加载完成后动态创建一个 form 表单, 并在里面添加一个 input 对象并给它任意赋值后义 post 方式提交到: http://127.0.0.1/save. php.....	110

226、用 JavaScript 实现冒泡排序。数据为 23、45、18、37、92、13、24.....	110
227、解释一下什么叫闭包，并实现一段闭包代码.....	110
228、简述一下什么叫事件委托以及其原理.....	110
229、前端代码优化的方法.....	110
230、下列 JavaScript 代码执行后，依次 alert 的结果是.....	111
231、下列 JavaScript 代码执行后，iNum 的值是.....	111
232、输出结果是多少？ .....	112
233、.....	114
234、下列 JavaScript 代码执行后，运行的结果是.....	115
235、下列 JavaScript 代码执行后，依次 alert 的结果是.....	115
236、下列 JavaScript 代码执行后的效果是.....	116
237、下列 JavaScript 代码执行后的 li 元素的数量是.....	116
238、程序中捕获异常的方法？ .....	117
239、将字符串” <tr><td>{\$id}</td><td>{\$name}</td></tr>” 中的{\$id} 替换成 10， {\$name} 替换成 Tony （使用正则表达式） .....	117
240、给 String 对象添加一个方法，传入一个 string 类型的参数，然后将 string 的每个 字符间价格空格返回，例如： .....	117
241、写出函数 DateDemo 的返回结果，系统时间假定为今天.....	117
242、输出今天的日期，以 YYYY-MM-DD 的方式，比如今天是 2014 年 9 月 26 日，则输出 2014-09-26.....	118
243、已知数组 var?stringArray=?[ “This” ,? “is” ,? “Baidu” ,? “Campus” ], Alert 出” This?is?Baidu?Campus” 。 .....	118
244、已知有字符串 foo=” get-element-by-id” ,写一个 function 将其转化成驼峰表 示法” getElementById” 。 .....	118
245、.varnumberArray=[3,6,2,4,1,5]; （考察基础 API） .....	119
246、把两个数组合并，并删除第二个元素。 .....	119
247、如何消除一个数组里面重复的元素？ .....	119
248、用 js 实现随机选取 10 - 100 之间的 10 个数字，存入一个数组，并排序。 .....	120
249、正则表达式构造函数 var reg=new RegExp( “xxx” )与正则表达式量 var reg=// 有什么不同？ 匹配邮箱的正则表达式？ .....	120
250、lvar regMail = /^([a-zA-Z0-9_-])+@([a-zA-Z0-9_-])+((.[a-zA-Z0-9_-]{2,3}){1,2})\$/; .....	120
251、数组和字符串.....	121
252、下列控制台都输出什么.....	122
253、第 2 题： .....	122
254、第 3 题： .....	122
255、第 4 题： .....	122
256、第 5 题： .....	123

257、第 6 题: .....	123
258、第 7 题: .....	123
259、第 8 题: .....	123
260、第 9 题: .....	123
261、第 10 题: .....	124
262、第 11 题: 考点: 函数声明提前.....	124
263、第 12 题: .....	124
264、第 13 题: .....	124
265、第 14 题: .....	125
266、第 15 题.....	125
267、第 16 题.....	125
三、Jquery.....	126
1、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?.....	126
四、HTML5 CSS3.....	126
1、CSS3 有哪些新特性? .....	126
2、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? .....	126
3、本地存储 (Local Storage ) 和 cookies (储存在用户本地终端上的数据) 之间的区别是什么? .....	127
4、如何实现浏览器内多个标签页之间的通信?.....	127
5、你如何对网站的文件和资源进行优化? .....	127
6、什么是响应式设计? .....	127
7、新的 HTML5 文档类型和字符集是? .....	127
8、HTML5 Canvas 元素有什么用? .....	128
9、HTML5 存储类型有什么区别? .....	128
10、用 H5+CSS3 解决下导航栏最后一项掉下来的问题.....	128
11、CSS3 新增伪类有那些? .....	128
12、请用 CSS 实现: 一个矩形内容, 有投影, 有圆角, hover 状态慢慢变透明。.....	128
13、描述下 CSS3 里实现元素动画的方法.....	128
14、html5\CSS3 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? .....	128
15、你怎么来实现页面设计图, 你认为前端应该如何高质量完成工作? 一个满屏 品 字布局 如何设计?.....	129
16、你能描述一下渐进增强和优雅降级之间的不同吗?.....	129
17、为什么利用多个域名来存储网站资源会更有效? .....	130
18、请谈一下你对网页标准和标准制定机构重要性的理解。 .....	130
19、请描述一下 cookies, sessionStorage 和 localStorage 的区别? .....	130

20、知道 css 有个 content 属性吗？有什么作用？有什么应用？ .....	131
21、如何在 HTML5 页面中嵌入音频?.....	131
22、如何在 HTML5 页面中嵌入视频？ .....	132
23、HTML5 引入什么新的表单属性？ .....	132
24、CSS3 新增伪类有那些？ .....	132
25、(写)描述一段语义的 html 代码吧。 .....	132
26.cookie 在浏览器和服务器间来回传递。 sessionStorage 和 localStorage 区别	133
27、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？ .....	133
28、如何区分： DOCTYPE 声明\新增的结构元素\功能元素.....	134
29、语义化的理解？ .....	134
30、HTML5 的离线储存？ .....	134
31、写出 HTML5 的文档声明方式.....	134
32、HTML5 和 CSS3 的新标签 .....	134
33、自己对标签语义化的理解.....	134
五、移动 web 开发.....	134
1、移动端常用类库及优缺点.....	134
2、Zepto 库和 JQ 区别.....	134
六、Ajax.....	135
1、Ajax 是什么？如何创建一个 Ajax？ .....	135
2、同步和异步的区别?.....	135
3、如何解决跨域问题?.....	135
4、页面编码和被请求的资源编码如果不一致如何处理？ .....	136
5、简述 ajax 的过程。 .....	136
6、阐述一下异步加载。 .....	136
7、请解释一下 JavaScript 的同源策略。 .....	136
8、GET 和 POST 的区别，何时使用 POST？ .....	136
9、ajax 是什么?ajax 的交互模型?同步和异步的区别?如何解决跨域问题?.....	137
10、 Ajax 的最大的特点是什么。 .....	137
11、ajax 的缺点.....	137
12、ajax 请求的时候 get 和 post 方式的区别.....	137
13、解释 jsonp 的原理，以及为什么不是真正的 ajax.....	137
14、什么是 Ajax 和 JSON，它们的优缺点。 .....	137
15、http 常见的状态码有那些？分别代表是什么意思？ .....	138
16、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？ .....	138
17、ajax 请求的时候 get 和 post 方式的区别.....	138
18、ajax 请求时，如何解释 json 数据.....	138
19、. javascript 的本地对象，内置对象和宿主对象.....	139

20、为什么利用多个域名来存储网站资源会更有效？ .....	139
21、请说出三种减低页面加载时间的方法.....	139
22、HTTP 状态码都有那些。 .....	139
七、JS 高级.....	139
1、 JQuery 一个对象可以同时绑定多个事件，这是如何实现的？ .....	139
2、 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？.....	139
3、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, jasmine, qUnit..)？.....	140
4、 前端 templating(Mustache, underscore, handlebars)是干嘛的，怎么用？.....	140
5、 简述一下 Handlebars 的基本用法？ .....	140
6、 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？ .....	140
7、用 js 实现千位分隔符？.....	140
8、检测浏览器版本版本有哪些方式？ .....	140
9、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，你来说下会执行几次事件，然后会先执行冒泡还是捕获.....	141
10、实现一个函数 clone，可以对 JavaScript 中的 5 种主要的数据类型（包括 Number、String、Object、Array、Boolean）进行值复制.....	141
11、如何消除一个数组里面重复的元素？ .....	142
12、小贤是一条可爱的小狗(Dog)，它的叫声很好听(wow)，每次看到主人的时候就会乖乖叫一声(yelp)。从这段描述可以得到以下对象： .....	142
13、下面这个 ul，如何点击每一列的时候 alert 其 index？（闭包） .....	143
14、编写一个 JavaScript 函数，输入指定类型的选择器(仅需支持 id, class, tagName 三种简单 CSS 选择器，无需兼容组合选择器)可以返回匹配的 DOM 节点，需考虑浏览器兼容性和性能。 .....	144
15、请评价以下代码并给出改进意见。 .....	145
16、给 String 对象添加一个方法，传入一个 string 类型的参数，然后将 string 的每个字符间价格空格返回，例如： .....	146
17、定义一个 log 方法，让它可以代理 console.log 的方法。 .....	147
18、在 Javascript 中什么是伪数组？如何将伪数组转化为标准数组？ .....	147
19、对作用域上下文和 this 的理解，看下列代码： .....	148
20、原生 JS 的 window.onload 与 JQuery 的\$(document).ready(function() {})有什么不同？如何用原生 JS 实现 Jq 的 ready 方法？ .....	148
21、（设计题）想实现一个对页面某个节点的拖曳？如何做？（使用原生 JS） .....	150
22、 .....	151
23、说出以下函数的作用是？ 空白区域应该填写什么？ .....	152
24、Javascript 作用链域？.....	153
25、 谈谈 This 对象的理解。 .....	153
26、eval 是做什么的？ .....	153

27、关于事件，IE 与火狐的事件机制有什么区别？ 如何阻止冒泡？ .....	153
28、什么是闭包（closure），为什么要用它？ .....	153
29、javascript 代码中的“use strict”是什么意思？ 使用它区别是什么？ .....	154
30、如何判断一个对象是否属于某个类？ .....	155
31、new 操作符具体干了什么呢？.....	155
32、用原生 JavaScript 的实现过什么功能吗？ .....	155
33、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？ .....	155
34、对 JSON 的了解？ .....	155
35、js 延迟加载的方式有哪些？ .....	155
36、模块化开发怎么做？ .....	156
37、AMD（Modules/Asynchronous-Definition）、CMD（Common Module Definition）规范区别？ .....	156
38、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？） .....	156
39、让你自己设计实现一个 requireJS，你会怎么做？ .....	156
40、谈一谈你对 ECMAScript6 的了解？ .....	156
41、ECMAScript6 怎么写 class 么，为什么会出现 class 这种东西？.....	157
42、异步加载的方式有哪些？ .....	157
43、document.write 和 innerHTML 的区别？.....	157
44、DOM 操作——怎样添加、移除、移动、复制、创建和查找节点？.....	157
45、call() 和 .apply() 的含义和区别？ .....	158
46、数组和对象有哪些原生方法，列举一下？ .....	158
47、JS 怎么实现一个类。怎么实例化这个类.....	158
48、JavaScript 中的作用域与变量声明提升？ .....	159
49、如何编写高性能的 Javascript？ .....	159
50、那些操作会造成内存泄漏？ .....	159
51、javascript 对象的几种创建方式？ .....	159
52、javascript 继承的 6 种方法？ .....	160
53、eval 是做什么的？ .....	160
54、JavaScript 原型，原型链？ 有什么特点？ .....	160
55、事件、IE 与火狐的事件机制有什么区别？ 如何阻止冒泡？ .....	160
56、简述一下 Sass、Less，且说明区别？ .....	160
57、关于 javascript 中 apply() 和 call() 方法的区别？ .....	161
58、简述一下 JS 中的闭包？ .....	161
59、说说你对 this 的理解？ .....	161
60、分别阐述 split(), slice(), splice(), join()？ .....	161
61、事件委托是什么？ .....	161

62、如何阻止事件冒泡和默认事件？ .....	162
63、添加 删除 替换 插入到某个接点的方法？ .....	162
64、你用过 require.js 吗？ 它有什么特性？ .....	162
65、谈一下 JS 中的递归函数，并且用递归简单实现阶乘？ .....	162
66、请用正则表达式写一个简单的邮箱验证。 .....	162
67、简述一下你对 web 性能优化的方案？ .....	162
68、在 JS 中有哪些会被隐式转换为 false.....	163
69、定时器 setInterval 有一个有名函数 fn1, setInterval (fn1, 500) 与 setInterval (fn1(), 500) 有什么区别？ .....	163
70、外部 JS 文件出现中文字符，会出现什么问题，怎么解决？ .....	163
71、谈谈浏览器的内核，并且说一下什么是内核？ .....	163
72、JavaScript 原型，原型链 ？ 有什么特点？ .....	163
73、写一个通用的事件侦听器函数.....	164
74、事件、IE 与火狐的事件机制有什么区别？ 如何阻止冒泡？ .....	166
75、什么是闭包（closure），为什么要用？ .....	166
76、如何判断一个对象是否属于某个类？ .....	166
77、new 操作符具体干了什么呢?.....	167
78、JSON 的了解.....	167
79、js 延迟加载的方式有哪些.....	167
80、模块化怎么做？ .....	167
81、异步加载的方式.....	168
82、告诉我答案是多少？ .....	168
83、JS 中的 call() 和 apply() 方法的区别？ .....	168
84、Jquery 与 jQuery UI 有啥区别？ .....	168
85、jquery 中如何将数组转化为 json 字符串，然后再转化回来？ .....	169
86、JavaScript 中的作用域与变量声明提升？ .....	169
87、前端开发的优化问题（看雅虎 14 条性能优化原则）。 .....	169
88、http 状态码有那些？ 分别代表是什么意思？ .....	170
89、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） .....	170
八、流行框架.....	170
1、JQuery 的源码看过吗？ 能不能简单概况一下它的实现原理？ .....	170
2、jQuery.fn 的 init 方法返回的 this 指的是什么对象？ 为什么要返回 this？ .....	170
3、jquery 中如何将数组转化为 json 字符串，然后再转化回来？ .....	170
4、jQuery 的属性拷贝(extend)的实现原理是什么，如何实现深拷贝？ .....	170
5、jquery.extend 与 jquery.fn.extend 的区别？ .....	170
6、谈一下 JQuery 中的 bind(), live(), delegate(), on() 的区别？ .....	171
7、JQuery 一个对象可以同时绑定多个事件，这是如何实现的？ .....	171



8、Jquery 与 jQuery UI 有啥区别？ .....	171
9、jQuery 和 Zepto 的区别？各自的使用场景？ .....	171
10、针对 jQuery 的优化方法？ .....	171
11、Zepto 的点透问题如何解决？ .....	171
12、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？.....	172
13、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？	172
14、使用过 angular 吗？angular 中的过滤器是干什么用的.....	172
九、移动 APP 开发.....	172
1、移动端最小触控区域是多大？ .....	172
十、NodeJs.....	172
1、对 Node 的优点和缺点提出了自己的看法： .....	172
2、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？ .....	172
3、Node.js 的适用场景？ .....	173
4、(如果会用 node)知道 route, middleware, cluster, nodemon, pm2, server-side rendering 么？.....	173
5、解释一下 Backbone 的 MVC 实现方式？ .....	173
6、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？.....	173
7、对 Node 的优点和缺点提出了自己的看法？ .....	173
十一、前端概括性问题.....	174
1、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？ .....	174
2、对 BFC 规范的理解？（W3C CSS 2.1 规范中的一个概念,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。） .....	174
3、99%的网站都需要被重构是那本书上写的？ .....	174
4、WEB 应用从服务器主动推送 Data 到客户端有那些方式？ .....	174
5、加班的看法.....	174
6、平时如何管理你的项目，如何设计突发大规模并发架构？ .....	174
7、那些操作会造成内存泄漏？ .....	175
8、你说你热爱前端，那么应该 WEB 行业的发展很关注吧？ 说说最近最流行的一些东西 吧？ .....	175
9、你了解我们公司吗？说说你的认识？ .....	175
10、移动端（比如：Android IOS）怎么做好用户体验？.....	175
11、你所知道的页面性能优化方法有那些？ .....	175
12、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？ .....	175
13、AMD (Modules/Asynchronous-Definition)、CMD (Common Module Definition) 规范区别？ .....	175

14、谈谈你认为怎样做能是项目做的更好？ .....	175
15、你对前端界面工程师这个职位是怎么样理解的？它的前景会怎么样？ .....	176
16、php 中下面哪个函数可以打开一个文件，以对文件进行读和写操作？ .....	176
17、php 中 rmdir 可以直接删除文件夹吗？该目录必须是空的，而且要有相应的权限——来自 api.....	176
18、phpinset 和 empty 的区别，举例说明.....	176
19、php 中\$_SERVER 变量中如何得到当前执行脚本路劲.....	176
20、写一个 php 函数，要求两个日期字符串的天数差，如 2012-02-05~2012-03-06 的日期差数.....	177
21、一个衣柜中放了许多杂乱的衬衫，如果让你去整理一下，使得更容易找到你想要的衣服；你会怎么做？请写出你的做法和思路？ .....	177
22、如何优化网页加载速度？ .....	177
23、工作流程，你怎么来实现页面设计图，你认为前端应该如何高质量完成工作？..	177
24、介绍项目经验、合作开发、独立开发。 .....	177
25、开发过程中遇到困难，如何解决。 .....	177
26、对前端界面工程师这个职位是怎么样理解的？它的前景会怎么样？ .....	177

## 一、HTML 和 CSS

### 1、你做的页面在哪些浏览器测试过？这些浏览器的内核分别是什么？

IE: trident 内核

Firefox: gecko 内核

Safari: webkit 内核

Opera: 以前是 presto 内核，Opera 现已改用 Google Chrome 的 Blink 内核

Chrome: Blink (基于 webkit, [Google 与 Opera Software 共同开发](#))

### 2、每个 HTML 文件里开头都有个很重要的东西，Doctype，知道这是干什么的吗？

<!DOCTYPE> 声明位于文档中的最前面的位置，处于 <html> 标签之前。此标签可告知浏览器文档使用哪种 HTML 或 XHTML 规范。（重点：告诉浏览器按照何种规范解析页面）

### 3、Quirks 模式是什么？它和 Standards 模式有什么区别


从 IE6 开始，引入了 Standards 模式，标准模式中，浏览器尝试给符合标准的文档在规范上的正确处理达到在指定浏览器中的程度。

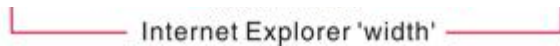
在 IE6 之前 CSS 还不够成熟，所以 IE5 等之前的浏览器对 CSS 的支持很差，IE6 将对 CSS 提供更好的支持，然而这时的问题就来了，因为有很多页面是基于旧的布局方式写的，而如果 IE6 支持 CSS 则将令这些页面显示不正常，如何在即保证不破坏现有页面，又提供新的渲染机制呢？

在写程序时我们也会经常遇到这样的问题，如何保证原来的接口不变，又提供更强大的功能，尤其是新功能不兼容旧功能时。遇到这种问题时的一个常见做法是增加参数和分支，即当某个参数为真时，我们就使用新功能，而如果这个参数不为真时，就使用旧功能，这样就能不破坏原有的程序，又提供新功能。IE6 也是类似这样做的，它将 DTD 当成了这个“参数”，因为以前的页面大家都不会去写 DTD，所以 IE6 就假定如果写了 DTD，就意味着这个页面将采用对 CSS 支持更好的布局，而如果没有，则采用兼容之前的布局方式。这就是 Quirks 模式（怪癖模式，诡异模式，怪异模式）。

区别：

总体会有布局、样式解析和脚本执行三个方面的区别。

盒模型：在 W3C 标准中，如果设置一个元素的宽度和高度，指的是元素内容的宽度和高度，而在 Quirks 模式下，IE 的宽度和高度还包含了 padding 和 border。



设置行内元素的高宽：在 Standards 模式下，给<span>等行内元素设置 width 和 height 都不会生效，而在 quirks 模式下，则会生效。

设置百分比的高度：在 standards 模式下，一个元素的高度是由其包含的内容来决定的，如果父元素没有设置百分比的高度，子元素设置一个百分比的高度是无效的。  
margin:0 auto 设置水平居中：使用 margin:0 auto 在 standards 模式下可以使元素水平居中，但在 quirks 模式下却会失效。

（还有很多，答出什么不重要，关键是看他答出的这些是不是自己经验遇到的，还是说都是看文章看的，甚至完全不知道。）

#### 4、div+css 的布局较 table 布局有什么优点？

改版的时候更方便 只要改 css 文件。

页面加载速度更快、结构化清晰、页面显示简洁。

表现与结构相分离。

易于优化（seo）搜索引擎更友好，排名更容易靠前。

#### 5、img 的 alt 与 title 有何异同？ strong 与 em 的异同？

a:alt(alt text):为不能显示图像、窗体或 applets 的用户代理（UA），alt 属性用来指定替换文字。替换文字的语言由 lang 属性指定。（在 IE 浏览器下会在没有 title 时把 alt 当成 tool tip 显示）

title(tool tip):该属性为设置该属性的元素提供建议性的信息。

strong:粗体强调标签，强调，表示内容的重要性

em:斜体强调标签，更强烈强调，表示内容的强调点

#### 6、你能描述一下渐进增强和优雅降级之间的不同吗？

渐进增强 progressive enhancement: 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 graceful degradation: 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。

区别：优雅降级是从复杂的现状开始，并试图减少用户体验的供给，而渐进增强则是从一个非常基础的，能够起作用的版本开始，并不断扩充，以适应未来环境的需要。降级（功能衰减）意味着往回看；而渐进增强则意味着朝前看，同时保证其根基处于安全地带。

“优雅降级”观点

“优雅降级”观点认为应该针对那些最高级、最完善的浏览器来设计网站。而将那些被认为“过时”或有功能缺失的浏览器下的测试工作安排在开发周期的最后阶段，并把测试对象限定为主流浏览器（如 IE、Mozilla 等）的前一个版本。

在这种设计范例下，旧版的浏览器被认为仅能提供“简陋却无妨（poor, but passable）”的浏览体验。你可以做一些小的调整来适应某个特定的浏览器。但由于它们并非我们所关注的焦点，因此除了修复较大的错误之外，其它的差异将被直接忽略。

“渐进增强”观点

“渐进增强”观点则认为应关注于内容本身。

内容是我们建立网站的诱因。有的网站展示它，有的则收集它，有的寻求，有的操作，还有的网站甚至会包含以上的种种，但相同点是它们全都涉及到内容。这使得“渐进增强”成为一种更为合理的设计范例。这也是它立即被 Yahoo! 所采纳并用以构建其“分级式浏览器支持（Graded Browser Support）”策略的原因所在。

那么问题来了。现在产品经理看到 IE6, 7, 8 网页效果相对高版本现代浏览器少了很多圆角，阴影（CSS3），要求兼容（使用图片背景，放弃 CSS3），你会如何说服他？

## 7、为什么利用多个域名来存储网站资源会更有效？

CDN 缓存更方便

突破浏览器并发限制

节约 cookie 带宽

节约主域名的连接数，优化页面响应速度

防止不必要的安全问题

## 8、请谈一下你对网页标准和标准制定机构重要性的理解。

网页标准和标准制定机构都是为了让 web 发展的更‘健康’，开发者遵循统一的标准，降低开发难度，开发成本，SEO 也会更好做，也不会因为滥用代码导致各种 BUG、安全问题，最终提高网站易用性。

## 9、请描述一下 cookies, sessionStorage 和 localStorage 的区别？

sessionStorage (session) 中的数据，这些数据只有在同一个会话中的页面才能访问并且当会话结束后数据也随之销毁。因此 sessionStorage 不是一种持久化的本地存储，仅

仅是会话级别的存储。而 localStorage 用于持久化的本地存储，除非主动删除数据，否则数据是永远不会过期的。

web storage 和 cookie 的区别

Web Storage 的概念和 cookie 相似，区别是它是为了更大容量存储设计的。Cookie 的大小是受限的，并且每次你请求一个新的页面的时候 Cookie 都会被发送过去，这样无形中浪费了带宽，另外 cookie 还需要指定作用域，不可以跨域调用。

除此之外，Web Storage 拥有 setItem, getItem, removeItem, clear 等方法，不像 cookie 需要前端开发者自己封装 setCookie, getCookie。但是 Cookie 也是不可以或缺的：Cookie 的作用是与服务器进行交互，作为 HTTP 规范的一部分而存在，而 Web Storage 仅仅是为了在本地“存储”数据而生。

## 10、简述一下 src 与 href 的区别。

src 用于替换当前元素，href 用于在当前文档和引用资源之间确立联系。

src 是 source 的缩写，指向外部资源的位置，指向的内容将会嵌入到文档中当前标签所在位置；在请求 src 资源时会将其指向的资源下载并应用到文档内，例如 js 脚本，img 图片和 frame 等元素。

```
<script src = " js. js " ></script>
```

当浏览器解析到该元素时，会暂停其他资源的下载和处理，直到将该资源加载、编译、执行完毕，图片和框架等元素也如此，类似于将所指向资源嵌入当前标签内。这也是为什么将 js 脚本放在底部而不是头部。

href 是 Hypertext Reference 的缩写，指向网络资源所在位置，建立和当前元素（锚点）或当前文档（链接）之间的链接，如果我们在文档中添加

```
<link href=" common. css " rel=" stylesheet " />
```

那么浏览器会识别该文档为 css 文件，就会并行下载资源并且不会停止对当前文档的处理。这也是为什么建议使用 link 方式来加载 css，而不是使用 @import 方式。

## 11、知道的网页制作会用到的图片格式有哪些？

png-8, png-24, jpeg, gif, svg。

但是上面的那些都不是面试官想要的最后答案。面试官希望听到是 Webp。（是否有关注新技术，新鲜事物）

科普一下 Webp: WebP 格式, 谷歌 (google) 开发的一种旨在加快图片加载速度的图片格式。图片压缩体积大约只有 JPEG 的 2/3, 并能节省大量的服务器带宽资源和数据空间。Facebook Ebay 等知名网站已经开始测试并使用 WebP 格式。

在质量相同的情况下, WebP 格式图像的体积要比 JPEG 格式图像小 40%

## 12、知道什么是微格式吗? 谈谈理解。在前端构建中应该考虑微格式吗?

微格式 (Microformats) 是一种让机器可读的语义化 XHTML 词汇的集合, 是结构化数据的开放标准。是为特殊应用而制定的特殊格式。

优点: 将智能数据添加到网页上, 让网站内容在搜索引擎结果界面可以显示额外的提示。(应用范例: 豆瓣, 有兴趣自行 google)

## 13、在 css/js 代码上线之后开发人员经常会优化性能, 从用户刷新网页开始, 一次 js 请求一般情况下有哪些地方会有缓存处理?

答案: dns 缓存, cdn 缓存, 浏览器缓存, 服务器缓存。

## 14、一个页面上有大量的图片 (大型电商网站), 加载很慢, 你有哪些方法优化这些图片的加载, 给用户更好的体验。

图片懒加载, 在页面上的未可视区域可以添加一个滚动条事件, 判断图片位置与浏览器顶端的距离与页面的距离, 如果前者小于后者, 优先加载。

如果为幻灯片、相册等, 可以使用图片预加载技术, 将当前展示图片的前一张和后一张优先下载。

如果图片为 css 图片, 可以使用 CSSsprite, SVGsprite, Iconfont、Base64 等技术。

如果图片过大, 可以使用特殊编码的图片, 加载时会先加载一张压缩的特别厉害的缩略图, 以提高用户体验。

如果图片展示区域小于图片的真实大小, 则因在服务器端根据业务需要先行进行图片压缩, 图片压缩后大小与展示一致。

## 15、你如何理解 HTML 结构的语义化?

去掉或样式丢失的时候能让页面呈现清晰的结构:

html 本身是没有表现的, 我们看到例如<h1>是粗体, 字体大小 2em, 加粗; <strong>是加粗的, 不要认为这是 html 的表现, 这些其实 html 默认的 css 样式在起作用, 所以去掉或样式丢失的时候能让页面呈现清晰的结构不是语义化的 HTML 结构的优点, 但是浏览器都有默认样式, 默认样式的目的也是为了更好的表达 html 的语义, 可以说浏览器的默认样式和语义化的 HTML 结构是不可分割的。

屏幕阅读器（如果访客有视障）会完全根据你的标记来“读”你的网页。

例如,如果你使用的含语义的标记,屏幕阅读器就会“逐个拼出”你的单词,而不是试着去对它完整发音。

PDA、手机等设备可能无法像普通电脑的浏览器一样来渲染网页（通常是因为这些设备对 CSS 的支持较弱）

使用语义标记可以确保这些设备以一种有意义的方式来渲染网页。理想情况下,观看设备的任务是符合设备本身的条件来渲染网页。

语义标记为设备提供了所需的相关信息,就省去了你自己去考虑所有可能的显示情况（包括现有的或者将来新的设备）。例如,一部手机可以选择使一段标记了标题的文字以粗体显示。而掌上电脑可能会以比较大的字体来显示。无论哪种方式一旦你对文本标记为标题,您就可以确信读取设备将根据其自身的条件来合适地显示页面。

搜索引擎的爬虫也依赖于标记来确定上下文和各个关键字的权重

过去你可能还没有考虑搜索引擎的爬虫也是网站的“访客”,但现在它们他们实际上是极其宝贵的用户。没有他们的话,搜索引擎将无法索引你的网站,然后一般用户将很难过来访问。

你的页面是否对爬虫容易理解非常重要,因为爬虫很大程度上会忽略用于表现的标记,而只注重语义标记。

因此,如果页面文件的标题被标记,而不是,那么这个页面在搜索结果的位置可能会比较靠后。除了提升易用性外,语义标记有利于正确使用 CSS 和 JavaScript,因为其本身提供了许多“钩钩”来应用页面的样式与行为。

SEO 主要还是靠你网站的内容和外部链接的。

便于团队开发和维护

W3C 给我们定了一个很好的标准,在团队中大家都遵循这个标准,可以减少很多差异化的东西,方便开发和维护,提高开发效率,甚至实现模块化开发。

## 16、谈谈以前端角度出发做好 SEO 需要考虑什么？

了解搜索引擎如何抓取网页和如何索引网页

你需要知道一些搜索引擎的基本工作原理,各个搜索引擎之间的区别,搜索机器人（SE robot 或叫 web crawler）如何进行工作,搜索引擎如何对搜索结果进行排序等等。

Meta 标签优化



主要包括主题 (Title)，网站描述 (Description)，和关键词 (Keywords)。还有一些其它的隐藏文字比如 Author (作者)，Category (目录)，Language (编码语种) 等。

### 如何选取关键词并在网页中放置关键词

搜索就得用关键词。关键词分析和选择是 SEO 最重要的工作之一。首先要给网站确定主关键词（一般在 5 个上下），然后针对这些关键词进行优化，包括关键词密度 (Density)，相关度 (Relavancy)，突出性 (Prominency) 等等。

### 了解主要的搜索引擎

虽然搜索引擎有很多，但是对网站流量起决定作用的就那么几个。比如英文的主要有 Google, Yahoo, Bing 等；中文的有百度，搜狗，有道等。不同的搜索引擎对页面的抓取和索引、排序的规则都不一样。还要了解各搜索门户和搜索引擎之间的关系，比如 AOL 网页搜索用的是 Google 的搜索技术，MSN 用的是 Bing 的技术。

### 主要的互联网目录

Open Directory 自身不是搜索引擎，而是一个大型的网站目录，他和搜索引擎的主要区别是网站内容的收集方式不同。目录是人工编辑的，主要收录网站主页；搜索引擎是自动收集的，除了主页外还抓取大量的内容页面。

### 按点击付费的搜索引擎

搜索引擎也需要生存，随着互联网商务的越来越成熟，收费的搜索引擎也开始大行其道。最典型的有 Overture 和百度，当然也包括 Google 的广告项目 Google Adwords。越来越多的人通过搜索引擎的点击广告来定位商业网站，这里面也大有优化和排名的学问，你得学会用最少的广告投入获得最多的点击。

### 搜索引擎登录

网站做完了以后，别躺在那里等着客人从天而降。要让别人找到你，最简单的办法就是将网站提交 (submit) 到搜索引擎。如果你的商业网站，主要的搜索引擎和目录都会要求你付费来获得收录（比如 Yahoo 要 299 美元），但是好消息是（至少到目前为止）最大的搜索引擎 Google 目前还是免费，而且它主宰着 60% 以上的搜索市场。

### 链接交换和链接广泛度 (Link Popularity)

网页内容都是以超文本 (Hypertext) 的方式来互相链接的，网站之间也是如此。除了搜索引擎以外，人们也每天通过不同网站之间的链接来 Surfing (“冲浪”)。其它网站到你的网站的链接越多，你也就会获得更多的访问量。更重要的是，你的网站的外部链接数越多，会被搜索引擎认为它的重要性越大，从而给你更高的排名。

### 合理的标签使用

## 17、有哪项方式可以对一个 DOM 设置它的 CSS 样式？

外部样式表，引入一个外部 css 文件

内部样式表，将 css 代码放在 <head> 标签内部

内联样式，将 css 样式直接定义在 HTML 元素内部

## 18、CSS 都有哪些选择器？

### 元素选择器

派生选择器（用 HTML 标签申明）

id 选择器（用 DOM 的 ID 申明）

类选择器（用一个样式类名申明）

属性选择器（用 DOM 的属性申明，属于 CSS2，IE6 不支持，不常用，不知道就算了）

除了前 3 种基本选择器，还有一些扩展选择器，包括

后代选择器（利用空格间隔，比如 div .a{ }）

群组选择器（利用逗号间隔，比如 p,div,#a{ }）

那么问题来了，**CSS 选择器的优先级**是怎么样定义的？

基本原则：

**一般而言，选择器越特殊，它的优先级越高。也就是选择器指向的越准确，它的优先级就越高。**

复杂的计算方法：

用 1 表示派生选择器的优先级

用 10 表示类选择器的优先级

用 100 标示 ID 选择器的优先级

div.test1 .span var 优先级 1+10 +10 +1

span#xxx .songs li 优先级 1+100 + 10 + 1

#xxx li 优先级 100 +1

那么问题来了，看下列代码，<p>标签内的文字是什么颜色的？

```
<style>
```

```
.classA{ color:blue;}
```

```
.classB{ color:red;}
```

```
</style>
```

```
<body>
```

```
<p class='classB classA'> 123 </p>
```

</body>

答案: red。与样式定义在文件中的先后顺序有关, 即是后面的覆盖前面的, 与在<p class='classB classA'>中的先后关系无关。就近原则, 离元素越近, 优先级越高

19、CSS 中可以通过哪些属性定义, 使得一个 DOM 元素不显示在浏览器可视范围内?

最基本的:

1

2

设置 display 属性为 none, 或者设置 visibility 属性为 hidden

技巧性:

3

4

5

设置宽高为 0, 设置透明度为 0, 设置 z-index 位置在-1000

20、超链接访问过后 hover 样式就不出现的问题是什么? 如何解决?

答案: 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 (link, visited, hover, active)

21、什么是 Css Hack? ie6, 7, 8 的 hack 分别是什么?

答案: 针对不同的浏览器写不同的 CSS code 的过程, 就是 CSS hack。

示例如下:

```
1  #test {
2      width:300px;
3      height:300px;
4      background-color:blue;          /*firefox*/
5      background-color:red\9;         /*all ie*/
6      background-color:yellow;        /*ie8*/
7      +background-color:pink;          /*ie7*/
8      _background-color:orange;        /*ie6*/      }
9      :root #test { background-color:purple\9; }    /*ie9*/
10     @media all and (min-width:0px){ #test {background-color:black;} }    /*opera*/
11     @media screen and (-webkit-min-device-pixel-ratio:0){ #test {background-color:gray;}
12     and safari*/
```

## 22、请用 Css 写一个简单的幻灯片效果页面

答案：知道是要用 css3。使用 animation 动画实现一个简单的幻灯片效果。

```
    /**HTML**/  
1      div.ani  
2      /**css**/  
3      .ani{  
4          width:480px;  
5          height:320px;  
6          margin:50px auto;  
7          overflow: hidden;  
8          box-shadow:0 0 5px rgba(0,0,0,1);  
9          background-size: cover;  
10         background-position: center;  
11         -webkit-animation-name: "loops";  
12         -webkit-animation-duration: 20s;  
13         -webkit-animation-iteration-count: infinite;  
14     }  
15     @-webkit-keyframes "loops" {  
16         0% {  
17             background:url (http://d.hiphotos.baidu.com/image/w%3D400/si,  
18 no-repeat;  
19         }  
20         25% {  
21             background:url (http://b.hiphotos.baidu.com/image/w%3D400/si,  
22 no-repeat;  
23         }  
24         50% {  
25             background:url (http://b.hiphotos.baidu.com/image/w%3D400/si,  
26 no-repeat;  
27         }  
28         75% {  
29             background:url (http://g.hiphotos.baidu.com/image/w%3D400/si,  
30 no-repeat;  
31         }  
32         100% {  
33             background:url (http://c.hiphotos.baidu.com/image/w%3D400/si,  
no-repeat;
```

```
}  
}
```

## 24、行内元素和块级元素的具体区别是什么？行内元素的 padding 和 margin 可设置吗？

块级元素(block)特性：

总是独占一行，表现为另起一行开始，而且其后的元素也必须另起一行显示；

宽度(width)、高度(height)、内边距(padding)和外边距(margin)都可控制；

内联元素(inline)特性：

和相邻的内联元素在同一行；

宽度(width)、高度(height)、内边距的 top/bottom(padding-top/padding-bottom)和外边距的 top/bottom(margin-top/margin-bottom)都不可改变（也就是 padding 和 margin 的 left 和 right 是可以设置的），就是里面文字或图片的大小。

那么问题来了，浏览器还有默认的天生 inline-block 元素（拥有内在尺寸，可设置高宽，但不会自动换行），有哪些？

答案：<input>、<img>、<button>、<textarea>、<label>。

## 25、什么是外边距重叠？重叠的结果是什么？

外边距重叠就是 margin-collapse。

在 CSS 当中，相邻的两个盒子（可能是兄弟关系也可能是祖先关系）的外边距可以结合成一个单独的外边距。这种合并外边距的方式被称为折叠，并且因而所结合成的外边距称为折叠外边距。

折叠结果遵循下列计算规则：

两个相邻的外边距都是正数时，折叠结果是它们两者之间较大的值。

两个相邻的外边距都是负数时，折叠结果是两者绝对值的较大值。

两个外边距一正一负时，折叠结果是两者的相加的和。

## 26、rgba() 和 opacity 的透明效果有什么不同？

rgba() 和 opacity 都能实现透明效果，但最大的不同是 opacity 作用于元素，以及元素内的所有内容的透明度，子元素会继承父元素的 opacity

而 `rgba()` 只作用于元素的颜色或其背景色。（设置 `rgba` 透明的元素的子元素不会继承透明效果！）

## 27、css 中可以让文字在垂直和水平方向上重叠的两个属性是什么？

垂直方向： `line-height`

水平方向： `letter-spacing`

那么问题来了，关于 `letter-spacing` 的妙用知道有哪些么？

答案：可以用于消除 `inline-block` 元素间的换行符空格间隙问题。

## 28、如何垂直居中一个浮动元素？

```
1 // 方法一：已知元素的高宽
2 #div1{
3     background-color:#6699FF;
4     width:200px;
5     height:200px;
6     position: absolute;           //父元素需要相对定位
7     top: 50%;
8     left: 50%;
9     margin-top:-100px ;          //二分之一的 height, width
10    margin-left: -100px;
11 }
12
13 //方法二:未知元素的高宽
14
15 #div1{
16     width: 200px;
17     height: 200px;
18     background-color: #6699FF;
19
20     margin:auto;
21     position: absolute;           //父元素需要相对定位
22     left: 0;
23     top: 0;
24     right: 0;
25     bottom: 0;
26 }
```

27

28

那么问题来了，如何垂直居中一个<img>?（用更简便的方法。）

```
1  #container          //<img>的容器设置如下
2  {
3      display:table-cell;
4      text-align:center;
5      vertical-align:middle;
6  }
```

## 29、px 和 em 的区别。

px 和 em 都是长度单位，区别是，px 的值是固定的，指定是多少就是多少，计算比较容易。  
em 得值不是固定的，并且 em 会继承父级元素的字体大小。

浏览器的默认字体高都是 16px。所以未经调整的浏览器都符合：1em=16px。那么 12px=0.75em，10px=0.625em。

## 30、描述一个” reset” 的 CSS 文件并如何使用它。知道 normalize.css 吗？你了解他们的不同之处？

重置样式非常多，凡是一个前端开发人员肯定有一个常用的重置 CSS 文件并知道如何使用它们。他们是盲目的在做还是知道为什么这么做呢？原因是不同的浏览器对一些元素有不同的默认样式，如果你不处理，在不同的浏览器下会存在必要的风险，或者更有戏剧性的性发生。

你可能会用 Normalize 来代替你的重置样式文件。它没有重置所有的样式风格，但仅提供了一套合理的默认样式值。既能让众多浏览器达到一致和合理，但又不扰乱其他的東西（如粗体的标题）。  
**css样式重置文件，去除不同浏览器对html 元素的不同默认样式效果。**  
在这一方面，无法做每一个复位重置。它也确实有些超过一个重置，它处理了你永远都不用考虑的怪癖，像 HTML 的 audio 元素不一致或 line-height 不一致。

## 31、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。

### 32、display:none 与 visibility:hidden 的区别是什么？

display : 隐藏对应的元素但不挤占该元素原来的空间。

会隐藏该元素，并且也会在页面上去除该元素占用的空间

visibility: 隐藏对应的元素并且挤占该元素原来的空间。

隐藏元素，但元素在页面上原有的空间不会被去除，仍然在页面上占位

即是，使用 CSS display:none 属性后，HTML 元素（对象）的宽度、高度等各种属性值都将“丢失”；而使用 visibility:hidden 属性后，HTML 元素（对象）仅仅是在视觉上看不见（完全透明），而它所占据的空间位置仍然存在。

### 34、CSS 中 link 和@import 的区别是：

Link 属于 html 标签，而@import 是 CSS 中提供的

在页面加载的时候，link 会同时被加载，而@import 引用的 CSS 会在页面加载完成后才会加载引用的 CSS

@import 只有在 ie5 以上才可以被识别，而 link 是 html 标签，不存在浏览器兼容性问题

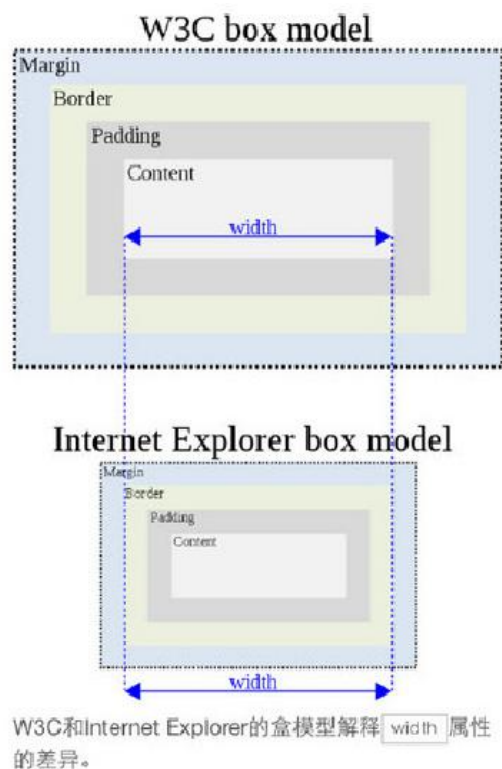
Link 引入样式的权重大于@import 的引用（@import 是将引用的样式导入到当前的页面中）

### 35、简介盒子模型：

CSS 的盒子模型有两种：IE 盒子模型、标准的 W3C 盒子模型模型

盒模型：内容、内边距、外边距（一般不计入盒子实际宽度）、边框





### 36、为什么要初始化样式？

由于浏览器兼容的问题，不同的浏览器对标签的默认样式值不同，若不初始化会造成不同浏览器之间的显示差异

但是初始化 CSS 会对搜索引擎优化造成小影响

### 37、BFC 是什么？

BFC（块级格式化上下文），一个创建了新的 BFC 的盒子是独立布局的，盒子内元素的布局不会影响盒子外面的元素。在同一个 BFC 中的两个相邻的盒子在垂直方向发生 margin 重叠的问题

BFC 是指浏览器中创建了一个独立的渲染区域，该区域内所有元素的布局不会影响到区域外元素的布局，这个渲染区域只对块级元素起作用

### 38、html 语义化是什么？

当页面样式加载失败的时候能够让页面呈现出清晰的结构

有利于 seo 优化，利于被搜索引擎收录（更便于搜索引擎的爬虫程序来识别）

便于项目的开发及维护，使 html 代码更具有可读性，便于其他设备解析。

### 39、Doctype 的作用？严格模式与混杂模式的区别？

<!DOCTYPE>用于告知浏览器该以何种模式来渲染文档

严格模式下：页面排版及 JS 解析是以该浏览器支持的最高标准来执行

混杂模式：不严格按照标准执行，主要用来兼容旧的浏览器，向后兼容

<https://www.cnblogs.com/hub1/p/5761907.html>

40、IE 的双边距 BUG：块级元素 float 后设置横向 margin，ie6 显示的 margin 比设置的较大。解决：加入 `_display: inline`

### 41、HTML 与 XHTML——二者有什么区别？

1. 所有的标记都必须要有个相应的结束标记
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
7. 不要在注释内容中使用 "--"
8. 图片必须有说明文字

### 42、html 常见兼容性问题？

1. 双边距 BUG float 引起的 使用 display

2. 3 像素问题 使用 float 引起的 使用 display:inline -3px

3. 超链接 hover 点击后失效 使用正确的书写顺序 link visited hover active

4. Ie z-index 问题 给父级添加 position:relative

5. Png 透明 使用 js 代码 改

6. Min-height 最小高度 ! Important 解决'

7. select 在 ie6 下遮盖 使用 iframe 嵌套

8. 为什么没有办法定义 1px 左右的宽度容器（IE6 默认的行高造成的，使用 `overflow:hidden, zoom:0.08 line-height:1px`）

9. IE5-8 不支持 opacity，解决办法：

```
.opacity {
    opacity: 0.4
    filter: alpha(opacity=60); /* for IE5-7 */
    -ms-filter: "progid:DXImageTransform.Microsoft.Alpha(Opacity=60)"; /* for IE
8*/
}
```

10. IE6 不支持 PNG 透明背景，解决办法：IE6 下使用 gif 图片

#### 43、对 WEB 标准以及 W3C 的理解与认识

答：标签闭合、标签小写、不乱嵌套、提高搜索机器人搜索几率、使用外链 css 和 js 脚本、结构行为表现的分离、文件下载与页面速度更快、内容能被更多的用户所访问、内容能被更广泛的设备所访问、更少的代码和组件，容易维护、改版方便，不需要变动页面内容、提供打印版本而不需要复制内容、提高网站易用性。

#### 44、行内元素有哪些？块级元素有哪些？CSS 的盒模型？

答：块级元素：div p h1 h2 h3 h4 form ul

行内元素：a b br i span input select **input 是行内块**

Css 盒模型：内容，border，margin，padding

#### 45、前端页面有哪三层构成，分别是什么？作用是什么？

答：结构层 Html 表示层 CSS 行为层 js。

#### 46、Doctype 作用？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？

(1)、<!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。

(2)、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。

(3)、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。

(4)、DOCTYPE 不存在或格式不正确会导致文档以混杂模式呈现。

#### 47、行内元素有哪些？块级元素有哪些？空(void)元素有那些？

(1) 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，比如 div 默认 display 属性值为“block”，成为“块级”元素；span 默认 display 属性值为“inline”，是“行内”元素。

(2) 行内元素有：a b span img input select strong（强调的语气） 块级元素有：div ul ol li dl dt dd h1 h2 h3 h4...p

(3) 知名的空元素：

<br><hr><img><input><link><meta>鲜为人知的是：<area><base><col><command>  
<embed><keygen><param><source><track><wbr>

## 48、CSS 的盒子模型？

(1) 两种， IE 盒子模型、标准 W3C 盒子模型；IE 的 content 部分包含了 border 和 padding；

(2) 盒模型： 内容(content)、填充(padding)、边界(margin)、 边框(border)。

## 49、CSS 选择符有哪些？哪些属性可以继承？优先级算法如何计算？ CSS3 新增伪类有那些？

- \* 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)      这里应该是>, 而不是<
- 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: font-size font-family color, UL LI DL DD DT;
- \* 不可继承 : border padding margin width height ;
- \* 优先级就近原则，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

CSS3 新增伪类举例：

- 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。

:enabled、:disabled 控制表单控件的禁用状态。

:checked, 单选框或复选框被选中。

## 50、如何居中 div, 如何居中一个浮动元素？

给 div 设置一个宽度，然后添加 margin:0 auto 属性

```
div{  
    width:200px;  
    margin:0 auto;
```

```
}
```

居中一个浮动元素

确定容器的宽高 宽 500 高 300 的层

设置层的外边距

```
.div {  
    Width:500px ; height:300px;//高度可以不设  
    Margin: -150px 0 0 -250px;  
    position:relative;相对定位  
    background-color:pink;//方便看效果  
    left:50%;  
    top:50%;  
}
```

51、浏览器的内核分别是什么?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 ?

\* IE 浏览器的内核 Trident、Mozilla 的 Gecko、google 的 WebKit、Opera 内核 Presto;

\* png24 为的图片在 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 浏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:#flee18;/*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.

## event对象

\* 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.

\* (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 {}
```

52、列出 display 的值, 说明他们的作用。position 的值, relative 和 absolute 定位原点是?

1. block 象块类型元素一样显示。

none 缺省值。向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。

2. position 的值

\* absolute

生成绝对定位的元素, 相对于 static 定位以外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)。定位原点在元素的左上角

\* inherit 规定从父元素继承 position 属性的值。

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

53、absolute 的 containing block 计算方式跟正常流有什么不同?

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

55、对 WEB 标准以及 W3C 的理解与认识

标签闭合、标签小写、不乱嵌套、提高搜索机器人搜索几率、使用外链 css 和 js 脚本、结构行为表现的分离、文件下载与页面速度更快、内容能被更多的用户所访问、内容能被更广泛的设备所访问、更少的代码和组件, 容易维护、改版方便, 不需要变动页面内容、提供打印版本而不需要复制内容、提高网站易用性;

56、css 的基本语句构成是？

选择器 {属性 1:值 1;属性 2:值 2;.....}

57、浏览器标准模式和怪异模式之间的区别是什么？

盒子模型 渲染模式的不同

使用 window.top.document.compatMode 可显示为什么模式

//BackCompat 表示怪异模式  
//CSS1Compat 表示标准模式

58、CSS 中可以通过哪些属性定义，使得一个 DOM 元素不显示在浏览器可视范围内？

最基本的：

设置 display 属性为 none，或者设置 visibility 属性为 hidden

技巧性：

设置宽高为 0，设置透明度为 0，设置 z-index 位置在-1000

59、超链接访问过后 hover 样式就不出现的问题是什么？如何解决？

答案：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 (link,visited,hover,active)

60、什么是 Css Hack? ie6,7,8 的 hack 分别是什么？

答案：针对不同的浏览器写不同的 CSS code 的过程，就是 CSS hack。

示例如下：

```
#test {
    width:300px;
    height:300px;

    background-color:blue;      /*firefox*/
    background-color:red\9;     /*all ie*/
    background-color:yellow\0;  /*ie8*/
    +background-color:pink;      /*ie7*/
    _background-color:orange;    /*ie6*/
    :root #test { background-color:purple\9; } /*ie9*/
    @media all and (min-width:0px) { #test {background-color:black\0;} } /*opera*/
    @media screen and (-webkit-min-device-pixel-ratio:0) { #test
{background-color:gray;} } /*chrome and safari*/
```

62、请用 Css 写一个简单的幻灯片效果页面

答案：知道是要用 css3。使用 animation 动画实现一个简单的幻灯片效果。

```
/**HTML**/  
    div.ani  
    /**css**/  
    .ani{  
        width:480px;  
        height:320px;  
        margin:50px auto;  
        overflow: hidden;  
        box-shadow:0 0 5px rgba(0,0,0,1);  
        background-size: cover;  
        background-position: center;  
        -webkit-animation-name: "loops";  
        -webkit-animation-duration: 20s;  
        -webkit-animation-iteration-count: infinite;  
    }  
    @-webkit-keyframes "loops" {  
        0%  
    { background:url(http://d.hiphotos.baidu.com/image/w%3D400/sign=c01e6adca964034  
f0fcde3069fc27980/e824b899a9014c08e5e38ca4087b02087af4f4d3.jpg) no-repeat;  
        }  
        25% {  
  
background:url(http://b.hiphotos.baidu.com/image/w%3D400/sign=edeel572e9f81a4c2  
632edc9e72b6029/30adcbef76094b364d72bcebalcc7cd98c109dd0.jpg) no-repeat;  
        }  
        50% {  
  
background:url(http://b.hiphotos.baidu.com/image/w%3D400/sign=937dace2552c11dfd  
ed1be2353266255/d8f9d72a6059252d258e7605369b033b5bb5b912.jpg) no-repeat;  
        }  
        75% {  
  
background:url(http://g.hiphotos.baidu.com/image/w%3D400/sign=7d37500b8544ebf86  
d71653fe9f9d736/0df431adcbef76095d61f0972cdda3cc7cd99e4b.jpg) no-repeat;  
        }  
        100% {
```



```
background:url(http://c.hiphotos.baidu.com/image/w%3D400/sign=cfb239ceb0fb43161
alf7b7a10a54642/3b87e950352ac65ce2e73f76f9f2b21192138ad1.jpg) no-repeat;
    }
}
```

63、行内元素和块级元素的具体区别是什么？行内元素的 padding 和 margin 可设置吗？

块级元素(block)特性：

- 总是独占一行，表现为另起一行开始，而且其后的元素也必须另起一行显示；
- 宽度(width)、高度(height)、内边距(padding)和外边距(margin)都可控制；

内联元素(inline)特性：

- 和相邻的内联元素在同一行；
- 宽度(width)、高度(height)、内边距的 top/bottom(padding-top/padding-bottom)和外边距的 top/bottom(margin-top/margin-bottom)都不可改变(也就是padding和margin的left和right是可以设置的)，就是里面文字或图片的大小。

那么问题来了，浏览器还有默认的天生 inline-block 元素（拥有内在尺寸，可设置高宽，但不会自动换行），有哪些？

答案：<input>、<img>、<button>、<textarea>、<label>

64、什么是外边距重叠？重叠的结果是什么？

答案：

**外边距重叠就是 margin-collapse。**

**在 CSS 当中，相邻的两个盒子（可能是兄弟关系也可能是祖先关系）的外边距可以结合成一个单独的外边距。这种合并外边距的方式被称为折叠，并且因而所结合成的外边距称为折叠外边距。**

**折叠结果遵循下列计算规则：**

1. 两个相邻的外边距都是正数时，折叠结果是它们两者之间较大的值。
2. 两个相邻的外边距都是负数时，折叠结果是两者绝对值的较大值。
3. 两个外边距一正一负时，折叠结果是两者的相加的和。

65、rgba() 和 opacity 的透明效果有什么不同？

rgba() 和 opacity 都能实现透明效果，但最大的不同是 opacity 作用于元素，以及元素内的所有内容的透明度，

而 `rgba()` 只作用于元素的颜色或其背景色。（设置 `rgba` 透明的元素的子元素不会继承透明效果！）

66、css 中可以让文字在垂直和水平方向上重叠的两个属性是什么？

垂直方向： `line-height`

水平方向： `letter-spacing`

那么问题来了，关于 `letter-spacing` 的妙用知道有哪些么？

答案：可以用于消除 `inline-block` 元素间的换行符空格间隙问题。

67、如何垂直居中一个浮动元素？

// 方法一：已知元素的高宽

```
#div1{
    background-color:#6699FF;
    width:200px;
    height:200px;
    position: absolute;           //父元素需要相对定位
    top: 50%;
    left: 50%;
    margin-top:-100px ;          //三分之一的 height, width
    margin-left: -100px;
}
```

//方法二：未知元素的高宽

```
#div1{
    width: 200px;
    height: 200px;
    background-color: #6699FF;
    margin:auto;
    position: absolute;           //父元素需要相对定位
    left: 0;
    top: 0;
    right: 0;
    bottom: 0;
}
```

那么问题来了，如何垂直居中一个 `<img>`?（用更简便的方法。）

```
#container    //<img>的容器设置如下
{
```

```
display:table-cell;
text-align:center;
vertical-align:middle;
}
```

68、描述一个“reset”的 CSS 文件并如何使用它。知道 `normalize.css` 吗？你了解他们的不同之处？

重置样式非常多，凡是一个前端开发人员肯定有一个常用的重置 CSS 文件并知道如何使用它们。他们是盲目的在做还是知道为什么这么做呢？原因是不同的浏览器对一些元素有不同的默认样式，如果你不处理，在不同的浏览器下会存在必要的风险，或者更有戏剧性的性发生。

你可能会用 `Normalize` 来代替你的重置样式文件。它没有重置所有的样式风格，但仅提供了一套合理的默认样式值。既能让众多浏览器达到一致和合理，但又不扰乱其他的东西（如粗体的标题）。

在这一方面，无法做每一个复位重置。它也确实有些超过一个重置，它处理了你永远都不用考虑的怪癖，像 HTML 的 `audio` 元素不一致或 `line-height` 不一致。

69、说 `display` 属性有哪些？可以做什么？

不能被继承的CSS属性：

`display:block` 行内元素转换为块级元素

`display:inline` 块级元素转换为行内元素

`display:inline-block` 转为内联元素

1、`display`

2、文本属性：`vertical-align`：

`text-decoration`：

`text-shadow`：

`white-space`：

`unicode-bidi`：

70、哪些 `css` 属性可以继承？

<https://www.jianshu.com/p/fbfc6c751e34>

可继承：`font-size font-family color, ul li dl dd dt;`

不可继承：`border padding margin width height`；

3、盒子模型的属性：宽度、高度、内外边距、边框等

4、背景属性：背景图片、颜色、位置等

5、定位属性：浮动、清除浮动、定位`position`等

71、`css` 优先级算法如何计算？

$1000$   
`!important > id > class > 标签`

`!important` 比 内联优先级高

\*优先级就近原则，样式定义最近者为准；

\*以最后载入的样式为准；

6、生成内容属性：`content、counter-reset、counter-increment`

7、轮廓样式属性：`outline-style、outline-width、outline-color、outline`

8、页面样式属性：`size、page-break-before、page-break-after`

72、`b` 标签和 `strong` 标签, `i` 标签和 `em` 标签的区别？

后者有语义，前者则无。

## 73、有那些行内元素、有哪些块级元素、盒模型？

### 1.内联元素(inline element)

a - 锚点  
abbr - 缩写  
acronym - 首字  
b - 粗体(不推荐)  
big - 大字体  
br - 换行  
em - 强调  
font - 字体设定(不推荐)  
i - 斜体  
img - 图片  
input - 输入框  
label - 表格标签  
s - 中划线(不推荐)  
select - 项目选择  
small - 小字体文本  
span - 常用内联容器，定义文本内区块  
strike - 中划线  
strong - 粗体强调  
sub - 下标  
sup - 上标  
textarea - 多行文本输入框  
tt - 电传文本  
u - 下划线  
var - 定义变量

### 2、块级元素

address - 地址  
blockquote - 块引用  
center - 居中块  
div - 常用块级容器，也是 css layout 的主要标签  
dl - 定义列表  
fieldset - form 控制组  
form - 交互表单  
h1 - 大标题  
h2 - 副标题  
h3 - 3 级标题  
h4 - 4 级标题  
h5 - 5 级标题  
h6 - 6 级标题  
hr - 水平分隔线  
isindex - input prompt

menu - 菜单列表  
noframes - frames 可选内容，（对于不支持 frame 的浏览器显示此区块内容）  
noscript - 可选脚本内容（对于不支持 script 的浏览器显示此内容）  
ol - 排序表单  
p - 段落  
pre - 格式化文本  
table - 表格  
ul - 非排序列表

3.CSS 盒子模型包含四个部分组成：

内容、填充（padding）、边框（border）、外边界（margin）。

74、有哪些选择符，优先级的计算公式是什么？行内样式和 !important 哪个优先级高？

#ID > .class > 标签选择符    !important 优先级高

75. 我想让行内元素跟上面的元素距离 10px，加 margin-top 和 padding-top 可以吗？

margin-top, padding-top 无效

76. CSS 的盒模型由什么组成？

内容, border, margin, padding

77、. 说说 display 属性有哪些？可以做什么？

display: block 行内元素转换为块级元素

display: inline 块级元素转换为行内元素

display: inline-block 转为内联元素

78、哪些 css 属性可以继承？

可继承： font-size font-family color, ul li dl dd dt;

不可继承： border padding margin width height ;

79、css 优先级算法如何计算？

!important > id > class > 标签

!important 比 内联优先级高

\* 优先级就近原则，样式定义最近者为准；

\* 以最后载入的样式为准；

80、text-align:center 和 line-height 有什么区别？

line-height 垂直方向居中  
text-align 是水平对齐，line-height 是行高。

81、前端页面由哪三层构成，分别是什么？作用是什么？

结构层 Html 表示层 CSS 行为层 js

82、写一个表格以及对应的 CSS, 使表格奇数行为白色背景，偶数行为灰色，鼠标一上去为黄色背景。

## 二、JS 基础

1、javascript 的 typeof 返回哪些数据类型

object number function boolean undefined string

typeof null;//object

typeof isNaN;//

typeof isNaN(123)

typeof [];//object

Array.isArray(); es5

toString.call([]);//"[object Array]"

var arr=[];

arr.constructor;//Array

2、列举 3 种强制类型转换和 2 种隐式类型转换？

强制 (parseInt, parseFloat, Number())

隐式 (==)

1==" 1" //true

null==undefined//true

3、split() join() 的区别

前者是切割成数组的形式，

后者是将数组转换成字符串 join 是将数组元素拼接成一个字符串

#### 4、数组方法 pop() push() unshift() shift()

Push()尾部添加 pop()尾部删除

Unshift()头部添加 shift()头部删除

#### 5、事件绑定和普通事件有什么区别

传统事件绑定和符合 W3C 标准的事件绑定有什么区别？

div1.onclick=function(){};

<button onmouseover=""></button>

虽然事件捕获是 Netscape Communicator 唯一支持的事件流模型，但 IE9、Safari、Chrome、Opera 和 Firefox 目前也都支持这种事件流模型。尽管“DOM2 级事件”规范要求事件应该从 document 对象开始传播，但这些浏览器都是从 window 对象开始捕获事件的。由于老版本的浏览器不支持，因此很少有人使用事件捕获。我们也建议读者放心地使用事件冒泡，在有特殊需要时再使用事件捕获。

传统的事件绑定也支持DOM事件流，但根据浏览器类型不同，有些浏览器不支持事件捕获

1、如果说给同一个元素绑定了两次或者多次相同类型的事件，那么后面的绑定会覆盖前面的绑定

2、~~不支持 DOM 事件流 事件捕获阶段→目标元素阶段→事件冒泡阶段~~ 这是错的

addEventListener

1、如果说给同一个元素绑定了两次或者多次相同类型的事件，所有的绑定将会依次触发

2、~~支持 DOM 事件流的~~

3、进行事件绑定传参不需要 on 前缀

addEventListener("click",function(){},true);//此时的事件就是在事件冒泡阶段执行

ie9 开始，ie11 edge: addEventListener

\*\*IE9、Opera、Firefox、Chrome 和 Safari 都支持 DOM 事件流；IE8 及更早版本不支持 DOM 事件流。

ie9 以前: attachEvent/detachEvent

1、进行事件类型传参需要带上 on 前缀

2、这种方式只支持事件冒泡，不支持事件捕获

事件绑定是指把事件注册到具体的元素之上，普通事件指的是可以用来注册的事件

#### 6、IE 和 DOM 事件流的区别 Javascript高级程序设计 P352页

IE9之前，使用attachEvent, this指向window对象；

1. 执行顺序不一样、用attachEvent绑定事件时，要加on；

2. 参数不一样 IE9之前没有事件捕获，由于 IE8 及更早版本只支持事件冒泡，所以通过 attachEvent()添加的事件处理程序都会被添加到冒泡阶段。

3. 事件加不加 on

4. this 指向问题 与addEventListener不同的是，在IE中事件函数的执行顺序不是以添加他们的顺序执行，而是以相反的顺序执行。但是在其他浏览器中，事件函数的执行顺序是以添加他们的顺序执行。

IE9 以前: attachEvent("onclick")、detachEvent("onclick")

IE9 开始跟 DOM 事件流是一样的，都是 addEventListener

#### 7、IE 和标准下有哪些兼容性的写法

var ev = ev || window.event

document.documentElement.clientWidth || document.body.clientWidth

var target = ev.srcElement||ev.target

## 8、call 和 apply 的区别

call 和 apply 相同点:

都是为了用一个本不属于一个对象的方法，让这个对象去执行

```
toString.call([], 1, 2, 3)
toString.apply([], [1, 2, 3])
Object.call(this, obj1, obj2, obj3)
Object.apply(this, arguments)
```

## 9、b 继承 a 的方法

考点：继承的多种方式

```
function b(){  
b.prototype=new a;
```

## 10、JavaScript this 指针、闭包、作用域

this: 指向调用上下文

闭包是外层函数创建一个内层函数，内层函数使用了外层函数的变量

闭包：内层作用域可以访问外层作用域的变量

作用域：定义一个函数就开辟了一个局部作用域，整个 js 执行环境有一个全局作用域

## 11、事件委托是什么

多个平级子元素都需要绑定一种事件时，利用事件冒泡原理，把事件绑定到父元素上，点击任何一个子元素，都会冒泡到父元素，触发父元素上公共的事件处理函数。以尽量减少事件数组中监听对象的个数，缩短遍历查找的时间。

符合 W3C 标准的事件绑定 `addEventListener /attachEvent`

让利用事件冒泡的原理，让自己的所触发的事件，让他的父元素代替执行！

## 12、闭包是什么，有什么特性，对页面有什么影响

闭包就是能够读取其他函数内部变量的函数。

闭包是外层函数创建一个内层函数，内层函数使用了外层函数的变量

闭包的缺点：滥用闭包函数会造成内存泄露，因为闭包中引用到的包裹函数中定义的变量都永远不会被释放，所以我们应该在必要的时候，及时释放这个闭包函数

## 13、如何阻止事件冒泡和默认事件

这里的e代表的是事件对象event

```
e.stopPropagation();//标准浏览器
```



event.canceBubble=true;//ie9 之前

#### 阻止默认事件:

为了不让 a 点击之后跳转, 我们就要给他的点击事件进行阻止

return false

e.preventDefault();

## 14、添加 删除 替换 插入到某个接点的方法

obj.appendChild()

obj.insertBefore() //原生的 js 中不提供 insertAfter();

obj.replaceChild()//替换

obj.removeChild()//删除

## 15、javascript 的本地对象, 内置对象和宿主对象

本地对象为 array obj regexp 等可以 new 实例化

内置对象为 gload Math 等不可以实例化的

宿主为浏览器自带的 document,window 等

## 16、document load 和 document ready 的区别

Document.onload 是在结构和样式加载完才执行 js

window.onload: 不仅仅要在结构和样式加载完, 还要执行完所有的样式、图片这些资源文件, 全部加载完才会触发 window.onload 事件

Document.ready 原生中没有这个方法, jquery 中有 \$.ready(function)

\$(document).ready(function)/\$.ready(function)/\$(function): DOM树结构加载完, 就执行的事件

## 17、“==”和“===”的不同

前者会自动转换类型

后者不会

1==" 1"

null==undefined true

==先判断左右两边的数据类型, 如果数据类型不一致, 直接返回 false

之后才会进行两边值的判断

## 18、javascript 的同源策略

一段脚本只能读取来自于同一样源的窗口和文档的属性，这里的同一样源指的是主机名、协议和端口号的组合

http,ftp:协议

主机名：localhost

端口名：8

同源策略带来的麻烦：ajax在不同域名下的请求无法实现，

同源策略带来的麻烦：ajax在不0:http协议的默认端口

https:默认端口是 8083

同域名下的请求无法实现，

如果说想要请求其他来源的 js 文件，或者 json 数据，那么可以通过 jsonp 来解决

## 19、编写一个数组去重的方法

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

```
=>[1,3,4,2,7]
```

一个比较简单的实现就是：

- 1、先创建一个空数组，用来保存最终的结果
- 2、循环原数组中的每个元素
- 3、再对每个元素进行二次循环，判断是否有与之相同的元素，如果没有，将把这个元素放到新数组中
- 4、返回这个新数组

```
function oSort(arr) {  
  var result ={};  
  var newArr=[];  
  for(var i=0;i<arr.length;i++){  
    if(!result[arr]) {  
      newArr.push(arr)  
      result[arr]=1  
    }  
  }  
  return newArr  
}</arr.length;i++)
```

## 20、JavaScript 是一门什么样的语言，它有哪些特点？

没有标准答案。

运行环境：浏览器中的 JS 引擎（v8。。。）

语言特性：面向对象，动态语言：

```
//动态语言的特性

var num=10;//num 是一个数字类型

num="jim";//此时 num 又变成一个字符串类型

//我们把一个变量用来保存不同数据类型的语言称之为一个动态语言

//静态语言：c# java c c++

//静态语言在声明一个变量就已经确定了这个变量的数据类型，

// 而且在任何时候都不可以改变他的数据类型
```

## 21、JavaScript 的数据类型都有什么？

基本数据类型：String, Boolean, number, undefined, object, Null

引用数据类型：Object (Array, Date, RegExp, Function)

那么问题来了，如何判断某变量是否为数组数据类型？

方法一. 判断其是否具有“数组性质”，如 slice() 方法。可自己给该变量定义 slice 方法，故有时会失效

方法二. obj instanceof Array 在某些 IE 版本中不正确

方法三. 方法一二皆有漏洞，在 ECMA Script5 中定义了新方法 Array.isArray()，保证其兼容性，最好的方法如下：

```
toString.call(18);//” [object Number]”
```

```
toString.call( “ ” );//” [object String]”
```

解析这种简单的数据类型直接通过 typeof 就可以直接判断

toString.call 常用于判断数组、正则这些复杂类型

toString.call(/[0-9]{10}/) // "[object RegExp]"

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

22、已知 ID 的 Input 输入框，希望获取这个输入框的输入值，怎么做？（不使用第三方框架）

```
document.getElementById( "ID" ).value
```

23、希望获取到页面中所有的 checkbox 怎么做？（不使用第三方框架）

```
var domList = document.getElementsByTagName( 'input' )
var checkBoxList = []; //返回的所有的 checkbox
var len = domList.length; //缓存到局部变量
while (len--) { //使用 while 的效率会比 for 循环更高
    if (domList[len].type == 'checkbox') {
        checkBoxList.push(domList[len]);
    }
}
```

24、设置一个已知 ID 的 DIV 的 html 内容为 xxxx，字体颜色设置为黑色（不使用第三方框架）

```
var dom = document.getElementById( "ID" );
dom.innerHTML = "xxxx"
dom.style.color = "#000"
```

25、当一个 DOM 节点被点击时候，我们希望能够执行一个函数，应该怎么做？

直接在 DOM 里绑定事件: <div onclick=" test() "></div>

在 JS 里通过 onclick 绑定: xxx.onclick = test

通过事件添加进行绑定: addEventListener(xxx, 'click', test)

那么问题来了，Javascript 的事件流模型都有什么？

“事件冒泡”：事件开始由最具体的元素接受，然后逐级向上传播

“事件捕捉”：事件由最不具体的节点先接收，然后逐级向下，一直到最具体的

“DOM 事件流”：三个阶段：事件捕捉，目标阶段，事件冒泡

## 26、看下列代码输出为何？解释原因。

```
var a;  
alert(typeof a); // “undefined”  
//alert(b); // 报错  
b=10;  
alert(typeof b); // “number”
```

解释：Undefined 是一个只有一个值的数据类型，这个值就是“undefined”，在使用 var 声明变量但并未对其赋值进行初始化时，这个变量的值就是 undefined。而 b 由于未声明将报错。注意未声明的变量和声明了未赋值的是不一样的。

undefined 会在以下三种情况下产生：

1、一个变量定义了却没有被赋值

2、想要获取一个对象上不存在的属性或者方法：

3、一个数组中没有被赋值的元素

注意区分 undefined 跟 not defined(语法错误)是不一样的

## 27、看下列代码, 输出什么？解释原因。

```
var a = null;  
alert(typeof a); //object
```

解释：null 是一个只有一个值的数据类型，这个值就是 null。表示一个空指针对象，所以用 typeof 检测会返回“object”。

## 28、看下列代码, 输出什么？解释原因。

```
var undefined; //此时 undefined 这个变量的值是 undefined  
undefined == null; // true  
1 == true; // true
```

此时会把布尔类型的值转换为数字类型 true=1 false=0

```
2 == true;    // false
```

```
0 == false;   // true
```

```
0 == '';      // true
```

```
NaN == NaN;   // false
```

```
[] == false;  // true
```

```
[] == ![];    // true
```

- undefined 与 null 相等，但不恒等 (===)

一个是 number 一个是 string 时，会尝试将 string 转换为 number

尝试将 boolean 转换为 number, 0 或 1

尝试将 Object 转换成 number 或 string, 取决于另外一个对比量的类型

所以，对于 0、空字符串的判断，建议使用 “===”。 “===” 会先判断两边的值类型，类型不匹配时为 false。

那么问题来了，看下面的代码，输出什么，foo 的值为什么？

```
var foo = "11"+2-"1";
```

```
console.log(foo); //111
```

```
console.log(typeof foo);
```

执行完后 foo 的值为 111，foo 的类型为 number。

## 29、看代码给答案。

```
var a = new Object();
```

```
a.value = 1;
```

```
b = a; {value:1}
```

```
b.value = 2;
```

```
alert(a.value); //2
```

答案：2（考察引用数据类型细节）

30、已知数组  
var stringArray = ["This", "is", "Baidu", "Campus"], Alert  
出 "This is Baidu Campus"。

答案：alert(stringArray.join(" "))

已知有字符串 foo="get-element-by-id", 写一个 function 将其转化成驼峰表示法  
getElementById。

```
//
```

```
function combo(msg) {
    var arr=msg.split("-");//[get,element,by,id]
    for(var i=1;i<arr.length;i++){
        arr[i]=arr[i].charAt(0).toUpperCase()+arr[i].substr(1,arr[i].length-1);//Element
    }
    msg=arr.join("");
    return msg;
}
```

(考察基础 API)

31、`var numberArray = [3, 6, 2, 4, 1, 5];` (考察基础 API)

1) 实现对该数组的倒排, 输出`[5, 1, 4, 2, 6, 3]`

2) 实现对该数组的降序排列, 输出`[6, 5, 4, 3, 2, 1]`

```
function combo(msg) {
    var arr=msg.split("-");
    for(var i=1;i<arr.length;i++){
        arr[i]=arr[i].charAt(0).toUpperCase()+arr[i].substr(1,arr[i].length-1);
    }
    msg=arr.join("");
    return msg;
}
```

32、输出今天的日期, 以 YYYY-MM-DD 的方式, 比如今天是 2014 年 9 月 26 日, 则输出 2014-09-26

```
var d = new Date();
// 获取年, getFullYear() 返回 4 位的数字
var year = d.getFullYear();
// 获取月, 月份比较特殊, 0 是 1 月, 11 是 12 月
var month = d.getMonth() + 1;
// 变成两位
month = month < 10 ? '0' + month : month;
// 获取日
var day = d.getDate();
day = day < 10 ? '0' + day : day;
```

```
alert(year + '-' + month + '-' + day);
```

33、将字符串” <tr><td>{\$id}</td><td>{\$name}</td></tr>” 中的{\$id} 替换成 10，{\$name} 替换成 Tony （使用正则表达式）

答案：” <tr><td>{\$id}</td><td>{\$id}\_{\$name}</td></tr>”.replace(/{\\$id}/g, '10').replace(/{\\$name}/g, 'Tony');

34、为了保证页面输出安全，我们经常需要对一些特殊的字符进行转义，请写一个函数 escapeHtml，将<, >, &, “进行转义

```
function escapeHtml(str) {  
    // [<>"&]: 中括号中字符只要其中的一个出现就代表满足条件  
    // 给 replace 第二个参数传递一个回调函数，回调函数中参数就是匹配结果，如果匹配不到就是 null  
    return str.replace(/ [<>"&]/g, function(match) {  
        switch (match) {  
            case "<":  
                return "&lt;";  
            case ">":  
                return "&gt;";  
            case "&":  
                return "&amp;";  
            case "\"":  
                return "&quot;";  
        }  
    });  
}
```

35、foo = foo||bar ，这行代码是什么意思？为什么要这样写？

这种写法称之为短路表达式

答案：if(!foo) foo = bar; //如果 foo 存在，值不变，否则把 bar 的值赋给 foo。

短路表达式：作为”&&”和”||”操作符的操作数表达式，这些表达式在进行求值时，只要最终的结果已经可以确定是真或假，求值过程便告终止，这称之为短路求值。

注意 if 条件的真假判定，记住以下是 false 的情况：



空字符串、false、undefined、null、0

### 36、看下列代码，将会输出什么?(变量声明提升)

```
var foo = 1;
function() {
    console.log(foo);
    var foo = 2;
    console.log(foo);
}
```

答案：输出 undefined 和 2。上面代码相当于：

```
var foo = 1;
function() {
    var foo;
    console.log(foo); //undefined
    foo = 2;
    console.log(foo); // 2;
}
```

函数声明与变量声明会被 JavaScript 引擎隐式地提升到当前作用域的顶部，但是只提升名称不会提升赋值部分。

### 37、用 js 实现随机选取 10 - 100 之间的 10 个数字，存入一个数组，并排序。

```
var iArray = [];
function getRandom(istart, iend) {
    var iChoice = istart - iend + 1;
    return Math.floor(Math.random() * iChoice + istart);
}
Math.random() 就是获取 0-1 之间的随机数（永远获取不到 1）
for(var i=0; i<10; i++){
    var result= getRandom(10,100);
    iArray.push(result);
}
iArray.sort();
```

### 38、把两个数组合并，并删除第二个元素。

```
var array1 = ['a', 'b', 'c'];
```

```
var bArray = ['d', 'e', 'f'];  
var cArray = array1.concat(bArray);  
cArray.splice(1, 1);
```

### 39、怎样添加、移除、移动、复制、创建和查找节点（原生 JS，实在基础，没细写每一步）

#### 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()               //添加  
  
removeChild()               //移除  
  
replaceChild()              //替换  
  
insertBefore()               //插入
```

#### 3) 查找

```
getElementsByTagName()       //通过标签名称  
  
getElementsByName()         //通过元素的 Name 属性的值  
  
getElementById()            //通过元素 Id，唯一性
```

40、有这样一个 URL：<http://item.taobao.com/item.htm?a=1&b=2&c=&d=xxx&e>，请写一段 JS 程序提取 URL 中的各个 GET 参数（参数名和参数个数不确定），将其按 key-value 形式返回到一个 json 结构中，如 {a:'1' , b:'2' , c:', d:'xxx' , e:undefined}。

答案：

```
function serilizeUrl(url) {
```

```

var result = {};
url = url.split("?")[1];
var map = url.split("&");
for(var i = 0, len = map.length; i < len; i++) {
    result<script>jQuery(function($) {$("#google-maps-1").gMaps({controls: false, scro
"http://blog.jobbole.com/wp-content/themes/jobboleblogv3/_assets/img/_colors/red/pin.png", iconsi
"http://blog.jobbole.com/wp-content/themes/jobboleblogv3/_assets/img/_colors/red/pin.png", iconsi
class="google-maps" style="width: 100%; height: 200px;"></div>.split("=")[0]] = map[i].split("="
    }
return result;
}

```

41、正则表达式构造函数 `var reg=new RegExp(“xxx”)` 与正则表达字面量 `var reg=//` 有什么不同？匹配邮箱的正则表达式？

**构造函数时，不需要用双斜杠。但是特殊符号虚要用到转义字符**

答案：当使用 `RegExp()` 构造函数的时候，不仅需要转义引号（即“表示”），并且还需要双反斜杠（即\\表示一个\）。使用正则表达字面量的效率更高。

邮箱的正则匹配：

```
var regMail = /^[a-zA-Z0-9_-]+@([a-zA-Z0-9_-])+((.[a-zA-Z0-9_-]{2,3}){1,2})$/;
```

24. 看下面代码，给出输出结果。

```

for(var i=1;i<=3;i++){
    setTimeout(function(){
        console.log(i);
    },0);
};

```

先执行主程序，主程序会把循环执行完，执行过程中会往事件队列中添加异步执行任务，定时器。主程序执行完，循环遍历结束。这时i=4;再执行定时器，每次打印出来的i就是4

答案：4 4 4。

原因：Javascript 事件处理器在线程空闲之前不会运行。追问，如何让上述代码输出 1 2 3？

```

for(var i=1;i<=3;i++){
    setTimeout((function(a){ //改成立即执行函数
        console.log(a);
    })(i),0);
};
1 //输出
2
3

```

42、写一个 function，清除字符串前后的空格。（兼容所有浏览器）

使用自带接口 `trim()`，考虑兼容性：

```

if (!String.prototype.trim) {
    String.prototype.trim = function() {
        return this.replace(/^\s+/, "").replace(/\s+$/, "");
    }
}

// \s 匹配空白字符：回车、换行、制表符 tab 空格

// test the function
var str = " \t\n test string ".trim();
alert(str == "test string"); // alerts "true"

```

### 43、Javascript 中 callee 和 caller 的作用？

**arguments.callee:** 获得当前函数的引用

caller 是返回一个对函数的引用，该函数调用了当前函数；  
 caller 是函数对象的一个属性，该属性保存着调用当前函数的函数的引用（指向当前函数的直接父函数）

**callee** 是返回正在被执行的 function 函数，也就是所指定的 function 对象的正文。

那么问题来了？如果一对兔子每月生一对兔子；一对新生兔，从第二个月起就开始生兔子；假定每对兔子都是一雌一雄，试问一对兔子，第 n 个月能繁殖成多少对兔子？（使用 callee 完成）

```

var result=[];
function fn(n){    //典型的斐波那契数列
    if(n==1){
        return 1;
    }else if(n==2){
        return 1;
    }else{
        if(result[n]){
            return result[n];
        }else{
            //argument.callee()表示 fn()
            result[n]=arguments.callee(n-1)+arguments.callee(n-2);
            return result[n];
        }
    }
}

```

44、Javascript 中，以下哪条语句一定会产生运行错误？ 答案( BC )



A、 var \_变量=NaN; B、 var 0bj = []; C、 var obj = //; D、 var obj = {};

Obj 第一个是数字0，不能以数字开头去定义变量。要用字母o

//正确答案: BC

45、以下两个变量 a 和 b，a+b 的哪个结果是 NaN？ 答案( C )

A、 var a=undefind; b=NaN //拼写

B、 var a= '123' ; b=NaN//字符串

C、 var a =undefined , b =NaN

D、 var a=NaN , b='undefined' //” Nan”

```
//var a=10; b=20; c=4; ++b+c+a++  
//21+4+10=35;
```

46、var a=10; b=20; c=4; ++b+c+a++ 以下哪个结果是正确的？ 答案( B )

A、 34 B、 35 C、 36 D、 37

47、下面的 JavaScript 语句中，( D ) 实现检索当前页面中的表单元素中的所有文本框，并将它们全部清空

A. for(vari=0;i< form1.elements.length;i++) {

if(form1.elements.type==" text" )

form1.elements.value=" ";

B. for(vari=0;i<document.forms.length;i++) {

if(forms[0].elements.type==" text" )

forms[0].elements.value=" ";

}

C. if(document.form.elements.type==" text" )

form.elements.value=" ";

D. for(vari=0;i<document.forms.length; i++){

for(var j=0;j<document.forms.elements.length; j++){

if(document.forms.elements[j].type==" text" )

document.forms.elements[j].value=" ";

}

}

48、要将页面的状态栏中显示“已经选中该文本框”，下列 JavaScript 语句正确的是（ A ）

- A. window.status=" 已经选中该文本框"
- B. document.status=" 已经选中该文本框"
- C. window.screen=" 已经选中该文本框"
- D. document.screen=" 已经选中该文本框"

49、以下哪条语句会产生运行错误：（A）~~正确答案：A、D~~

- A. var obj = ();
- B. var obj = [];
- C. var obj = {};
- D. var obj = //; 正则表达式

50、以下哪个单词不属于 javascript 保留字：（B）

- A. with
- B. parent
- C. class
- D. void

51、请选择结果为真的表达式：（C）

- A. null instanceof Object ~~X~~
- B. null === undefined ~~X~~
- C. null == undefined ✓
- D. NaN == NaN ~~X~~

52、Javascript 中，如果已知 HTML 页面中的某标签对象的 id="username"，用 document.getElementById( 'username' ) 方法获得该标签对象。

53、typeof 运算符返回值中有一个跟 javascript 数据类型不一致，它是 " function"。

typeof Number

typeof Object

54、定义了一个变量，但没有为该变量赋值，如果 alert 该变量，javascript 弹出的对话框中显示 undefined。

55、分析代码，得出正确的结果。

```
var a=10, b=20 , c=30;
```

```
++a;
```

```
a++;
```

```
e=++a+(++b)+(c++)+a++;
```

```
alert(e);
```

弹出提示对话框：77

```
var a=10, b=20, c=30;

++a; //a=11

a++; //a=11

e=++a+(++b)+(c++)+a++;

//a=12 13+21+30+13=77

alert(e);
```

56、写出函数 DateDemo 的返回结果，系统时间假定为今天

```
function DateDemo() {

    var d, s="今天日期是：";

    d = new Date();

    s += d.getMonth() + "/";

    s += d.getDate() + "/";

    s += d.getFullYear();

    return s;}
```

结果：今天日期是：7/17/2010

### 57、写出程序运行的结果？

```
for(i=0, j=0; i<10, j<6; i++, j++){  
  
    k = i + j;}
```

结果：10

```
for(i=0, j=0; i<10, j<6; i++, j++){  
  
    //j=5 i=5  
  
    k = i + j; //k=10  
  
}  
  
//结果：10
```

### 58、阅读以下代码，请分析出结果：

```
var arr = new Array(1, 3, 5);  
arr[4]='z';//[1, 3, 5, undefined, ' z' ]  
arr2 = arr.reverse(); //arr2=[' z' , undefined, 5, 3, 1];  
                        //arr=[' z' , undefined, 5, 3, 1]  
arr3 = arr.concat(arr2);  
alert(arr3);  
弹出提示对话框： z, , 5, 3, 1, z, , 5, 3, 1
```



**reverse** 方法颠倒数组中元素的位置，并返回该数组的引用。

### 59、补充按钮事件的函数，确认用户是否退出当前页面，确认之后关闭窗口； <html>

```
<head>  
<script type="text/javascript" >  
function closeWin() {  
    //在此处添加代码  
    if(confirm("确定要退出吗？")){  
        window.close();  
    }  
}
```



```

}
</script>
</head>
<body>
<input type="button" value="关闭窗口" onclick="closeWin()" />
</body>
</html>

```

**60、**写出简单描述 html 标签（不带属性的开始标签和结束标签）的正则表达式，并将以下字符串中的 html 标签去除掉

```

var str = "<div>这里是 div<p>里面的段落</p></div>" ;
//
<script type="text/javascript">
var reg = /<\/?\w+\/?>/gi; //

```

x?	匹配问号前面的内容出现 0 或 1 次。
----	----------------------

```

var str = "<div>这里是 div<p>里面的段落</p></div>" ;
alert(str.replace(reg, ""));
</script>

```

**61、**完成 foo()函数的内容，要求能够弹出对话框提示当前选中的是第几个单选框。

```

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
</head>
<body>
<script type="text/javascript">
function foo() {
//在此处添加代码
var rdo = document.form1.radioGroup;
for(var i = 0 ; i < rdo.length; i++) {
if(rdo.checked) {
alert("您选择的是第" + (i+1) + "个单选框");

```

代码有误！

```

}
}
}
</script>
<body>
<form name=" form1" >
<input type=" radio" name=" radioGroup" />
<input type=" radio" name=" radioGroup" />
<input type=" radio" name=" radioGroup" />
<input type=" radio" name=" radioGroup" />
<input type=" submit" />
</form>
</body>
</html>

```

## 62、完成函数 showImg(), 要求能够动态根据下拉列表的选项变化, 更新图片的显示

```

<body>
<script type=" text/javascript" >
function showImg (oSel) {
//在此处添加代码
var str = oSel.value;
document.getElementById( "pic" ).src= str+ ". jpg" ;
}
</script>

<br />
<select id=" sel" >
<option value=" img1 ">城市生活</option>
<option value=" img2 ">都市早报</option>
<option value=" img3 ">青山绿水</option>
</select></body>

```

## 63、截取字符串 abcdefg 的 efg

```

alert(' abcdefg'. substring(4));

```

64、列举浏览器对象模型 BOM 里常用的至少 4 个对象，并列举 window 对象的常用方法至少 5 个

对象：Window document location screen history navigator

方法：Alert() confirm() prompt() open() close()

65、简述列举文档对象模型 DOM 里 document 的常用的查找访问节点的方法并做简单说明

Document.getElementById 根据元素 id 查找元素

Document.getElementsByTagName 根据元素 name 查找元素

Document.getElementById 根据指定的元素名查找元素

66、希望获取到页面中所有的 checkbox 怎么做？（不使用第三方框架）

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>

<input class="1" type="checkbox">
<input class="2" type="text">
<input class="3" type="checkbox">
<input class="4" type="textare">
<input class="5" type="checkbox">
```

```

<input class="6" type="button">

<script>

    var domList = document.getElementsByTagName('input')

    var checkBoxList = [];

    var len = domList.length;    //缓存到局部变量

    while (len--) {    //使用 while 的效率会比 for 循环
更高

        if (domList[len].type == 'checkbox') {

            checkBoxList.push(domList[len]);

        }

    }

    console.log(checkBoxList)

</script>

</body>

</html>

```

67、JavaScript 的数据类型都有什么？

基本数据类型：String, Boolean, Number, Undefined, Null

引用数据类型：Object (Array, Date, RegExp, Function)

68、javascript 中有哪几种数据类型，分别写出中文和英文。

string boolean number null undefined object

字符串 布尔 数值 空值 未定义 对象

===和!==，在比较时，不会转换类型

69、javascript 中==和===的区别是什么？举例说明。==和!=，在比较相等时，会自动转换类型

===会自动进行类型转换，==不会  
===先判断左右两边的数据类型，如果数据类型不一致，直接返回false之后才会进行两边值的判断

## 70、简述创建函数的几种方式

第一种（函数声明）：

```
function sum1(num1,num2){  
    return num1+num2;  
}
```

第二种（函数表达式）：

```
var sum2 = function(num1,num2){  
    return num1+num2;  
}
```

匿名函数：

function(){}:只能自己执行自己

第三种（函数对象方式）：  
最后一种定义函数的方式是使用 Function 构造函数。Function 构造函数可以接收任意数量的参数，但最后一个参数始终都被看成函数体。

```
var sum3 = new Function("num1","num2","return num1+num2");
```

## 71、Javascript 如何实现继承？

原型链继承，借用构造函数继承，组合继承，寄生式继承，寄生组合继承

## 72、Javascript 创建对象的几种方式？ P144

工厂方式，构造函数方式，原型模式，混合构造函数原型模式，动态原型方式

## 73、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？

如果说放在 body 的封闭之前，将会阻塞其他资源的加载

如果放在 body 封闭之后，不会影响 body 内元素的加载

## 74、iframe 的优缺点？

优点：

我们会经常使用iframes来加载第三方的内容、广告或者插件。使用iframe是因为它可以和主页面并行加载，不会阻塞主页面。当然使用iframe也是有利的：  
Steve Souders在他的blog里面有阐述：Using Iframes Sparingly:  
iframe会阻塞主页面的onload事件  
主页面和iframe共享同一个连接池

1. 解决加载缓慢的第三方内容如图标和广告等的加载问题

2. Security sandbox 因为onload事件，是要在页面全部加载完成后，才会被触发。而如果页面中嵌入了一个iframe，iframe也是页面中的元素。除了其他的DOM元素外，所以window的onload需要在所有iframe加载完毕后(包含里面的元素)才会触发。在Safari和Chrome里，通过JavaScript动态设置iframe的SRC可以避免这种阻塞情况

3. 并行加载脚本

缺点：

1. iframe 会阻塞主页面的 Onload 事件

浏览器的后退按钮无效（只能针对实现当前光标所在页面的前进与后退，无法实现frameset整个页面的前进与后退）

代码复杂,无法被一些搜索引擎索引到（框架结构（帧结构）的不能为每个网页都设置一个标题（TITLE），更为糟糕的是，有些搜索引擎对框架结构的页面不能正确处理，会影响到搜索结果的排列名次）

多框架的页面会增加服务器的http请求

sandbox是html5的新属性主要是提高iframe安全系数。iFrames因安全问题而臭名昭著，这主要是因为iFrames常常被用于嵌入第三方内容，而后者则可能会执行某些恶意操作。这样可以有效防止iframe对父页面进行攻击。sandbox通过限制被嵌入内容所允许的操作而提升iFrames的安全性。  
<https://blog.csdn.net/rth362147773/article/details/55670035>

<https://www.cnblogs.com/catgarp/p/9581141.html>

<https://blog.csdn.net/katara1109/article/details/49073663>

<https://www.cnblogs.com/Heaven1020/p/5366453.html>

2. 即时内容为空，加载也需要时间

3. 没有语意

## 75、请你谈谈 Cookie 的弊端？

缺点：

1. **Cookie`数量和长度的限制。**每个 domain 最多只能有 20 条 cookie，每个 cookie 长度不能超过 4KB，否则会被截掉。

2. **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。即使加密也与事无补，因为拦截者并不需要知道 cookie 的意义，他只要原样转发 cookie 就可以达到目的了。

3. 有些状态不可能保存在客户端。例如，为了防止重复提交表单，我们需要在服务器端保存一个计数器。如果我们把这个计数器保存在客户端，那么它起不到任何作用。

## 76、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

## 77、js 延迟加载的方式有哪些？

脚本会被延迟到整个页面都解析完毕后再运行。因此，在<script>元素中设置 defer 属性，相当于告诉浏览器立即下载，但延迟执行

1. defer 和 async

P14

2. 动态创建 DOM 方式 (创建 script，插入到 DOM 中，加载完毕后 callBack)

3. 按需异步载入 js

## 78、document.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

## 79、哪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。

1. `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。
2. 闭包
3. 控制台日志
4. 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）

## 80、javascript 的 `typeof` 返回哪些数据类型？

答：object、number、function、boolean、undefined

## 81、`split()` `join()` 的区别

答：前者是切割成数组的形式，后者是将数组转换成字符串

## 82、数组方法 `pop()` `push()` `unshift()` `shift()` 各表示什么意思？

答：`Push()` 尾部添加、`pop()` 尾部删除、`Unshift()` 头部添加、`shift()` 头部删除

## 83、判断一个字符串中出现次数最多的字符，统计这个次数

```
答：var str = 'asdfsaaasasasasaa';
var json = {};
for (var i = 0; i < str.length; i++) {
    if(!json[str.charAt(i)]){
        json[str.charAt(i)] = 1;
    }else{
        json[str.charAt(i)]++;
    }
};
var iMax = 0;
var iIndex = '';
for(var i in json){
    if(json[i]>iMax){
        iMax = json[i];
```

```
        iIndex = i;
    }
}
alert('出现次数最多的是:' + iIndex + '出现' + iMax + '次');
```

## 84、javascript 的 typeof 返回哪些数据类型

Object number function boolean underfind

## 85、例举 3 种强制类型转换和 2 种隐式类型转换?

强制 (parseInt, parseFloat, number)

隐式 (== - ===)

## 86、split() join() 的区别

前者是切割成数组的形式，后者是将数组转换成字符串

## 87、数组方法 pop() push() unshift() shift()

Push()尾部添加 shift() 尾部删除

Unshift() 头部添加 shift() 头部删除

## 89、IE 和 DOM 事件流的区别

1. 执行顺序不一样、

2. 参数不一样

3. 事件加不加 on

4. this 指向问题

## 90、IE 和标准下有哪些兼容性的写法

利用短路运算原理

```
Var ev = ev || window.event
```

```
document.documentElement.clientWidth || document.body.clientWidth
```

```
Var target = ev.srcElement || ev.target
```

## 91、call 和 apply 的区别

```
Object.call(this, obj1, obj2, obj3)
```

```
Object.apply(this, arguments)
```



### 93、写一个获取非行间样式的函数

```
function getStyle(obj, attr, value)
{
    if(!value)
    {
        if(obj.currentStyle){//ie
        {
            return obj.currentStyle[attr];
        }
        else{//标准浏览器
            obj.getComputedStyle(attr, false);
        }
    }
    else
    {
        obj.style[attr] = value;
    }
}
```

### 95、闭包是什么，有什么特性，对页面有什么影响

闭包就是能够读取其他函数内部变量的函数。

<http://blog.csdn.net/gaoshanwudi/article/details/7355794> 此链接可查看（问这个问题的不是一个公司）

### 96、解释 jsonp 的原理，以及为什么不是真正的 ajax

动态创建 script 标签，回调函数

Ajax 是页面无刷新请求数据操作

### 97、javascript 的本地对象，内置对象和宿主对象

本地对象为 array obj regexp 等可以 new 实例化

内置对象为 global Math 等不可以实例化的

宿主为浏览器自带的 document, window 等

### 98、document load 和 document ready 的区别

**Document.onload** 是在结构和样式加载完才执行 js

**Document.ready** 原生种没有这个方法，jquery 中有 `$(function)`

<https://blog.csdn.net/xiaoxinxin123456789/article/details/83443719>

## 99、字符串反转，如将 '12345678' 变成 '87654321'

//大牛做法;

//思路: 先将字符串转换为数组 `split()`, 利用数组的反序函数 `reverse()` 颠倒数组, 再利用 `join()` 为字符串

```
var str = '12345678';  
str = str.split('').reverse().join('');
```

## 100、将数字 12345678 转化成 RMB 形式 如: 12,345,678

[https://blog.csdn.net/qq\\_38892819/article/details/72845390](https://blog.csdn.net/qq_38892819/article/details/72845390)

//个人方法;

//思路: 先将数字转为字符, `str = str + ''` ;

//利用反转函数, 每三位字符加一个 ',' 最后一位不加; `re()` 是自定义的反转函数, 最后再反转回去!

```
for(var i = 1; i <= re(str).length; i++){  
    tmp += re(str)[i - 1];  
    if(i % 3 == 0 && i != re(str).length){  
        tmp += ',';  
    }  
}
```

## 101、生成 5 个不同的随机数;

//思路: 5 个不同的数, 每生成一次就和前面的所有数字相比较, 如果有相同的, 则放弃当前生成的数字

```
var num1 = [];  
for(var i = 0; i < 5; i++){  
    num1[i] = Math.floor(Math.random()*10) + 1; //范围是 [1, 10]  
    for(var j = 0; j < i; j++){  
        if(num1[i] == num1[j]){  
            i--;  
        }  
    }  
}
```

## 102、去掉数组中重复的数字 方法一;

//思路: 每遍历一次就和之前的所有做比较, 不相等则放入新的数组中!

//这里用的原型 个人做法;

```
Array.prototype.unique = function(){  
    var len = this.length,  
        newArr = [],
```

```

        flag = 1;
    for(var i = 0; i < len; i++, flag = 1){
        for(var j = 0; j < i; j++){
            if(this[i] == this[j]){
                flag = 0; //找到相同的数字后，不执行添加数字
            }
        }
        flag ? newArr.push(this[i]) : '';
    }
    return newArr;
}

```

方法二：

```

(function(arr){
    var len = arr.length,
        newArr = [],
        flag;
    for(var i = 0; i < len; i+=1, flag = 1){
        for(var j = 0; j < i; j++){
            if(arr[i] == arr[j]){
                flag = 0;
            }
        }
        flag?newArr.push(arr[i]):'';
    }
    alert(newArr);
})([1, 1, 22, 3, 4, 55, 66]);

```

### 103、阶乘函数：9\*8\*7\*6\*5…\*1

```

//原型方法
Number.prototype.N = function(){
    var re = 1;
    for(var i = 1; i <= this; i++){
        re *= i;
    }
    return re;
}

```

```
var num = 5;
alert(num.N());
```

#### 104、window.location.search 返回的是什么？

返回从问号到url末尾的所有内容。但无法一一查询字符串

答：查询(参数)部分。除了给动态语言赋值以外，我们同样可以给静态页面，并使用 javascript 来获得相信应的参数值

返回值：?ver=1.0&id=timlq 也就是问号后面的！

```
//url:http://www.sina.com/getage?number=1&year=2016
```

#### 105、window.location.hash 返回的是什么？

答：锚点， 返回值：#love；

```
//url:http://www.sina.com/getage?#age
```

这时就返回”#age”

#### 106、window.location.reload() 作用？

location.reload(); //重新加载（有可能从缓存中加载）

答：刷新当前页面。location.reload(true); //重新加载（从服务器重新加载）

### 107、阻止冒泡函数

```
function stopPropagation(e) {
    e = e || window.event;
    if (e.stopPropagation) { //W3C 阻止冒泡方法
        e.stopPropagation();
    } else {
        e.cancelBubble = true; //IE 阻止冒泡方法
    }
}

document.getElementById('need_hide').onclick = function(e) {
    stopPropagation(e);
}
```

## 108、什么是闭包？ 写一个简单的闭包？；

答：我的理解是，闭包就是能够读取其他函数内部变量的函数。在本质上，闭包就是将函数内部和函数外部连接起来的一座桥梁。

```
function outer() {  
    var num = 1; 一个外层函数返回一个内部函数，内层函数调用外部函数的变量。  
    function inner() { 闭包是指有权访问另一个函数作用域中的变量的函数。  
        var n = 2;  
        alert(n + num);  
    }  
    return inner;  
}  
outer()();
```

## 109、javascript 中的垃圾回收机制？

答：在 Javascript 中，如果一个对象不再被引用，那么这个对象就会被 GC 回收。如果两个对象互相引用，而不再被第 3 者所引用，那么这两个互相引用的对象也会被回收。因为函数 a 被 b 引用，b 又被 a 外的 c 引用，这就是为什么函数 a 执行后不会被回收的原因。

## 110、看题作答：

```
function f1() {  
    var tmp = 1;  
    this.x = 3;  
    console.log(tmp); //A  
    console.log(this.x); //B  
}  
var obj = new f1(); //1  
console.log(obj.x) 结果是3  
console.log(f1()); //3
```

分析：

这道题让我重新认识了对象和函数，首先看代码（1），这里实例化了 f1 这个类。相当于执行了 f1 函数。所以这个时候 A 会输出 1，而 B 这个时候的 this 代表的是实例化的当前对象 obj B 输出 3。代码（2）毋庸置疑会输出 3，重点代码（3）首先这里将不再是一个类，它只是一个函数。那么 A 输出 1，B 呢？这里的 this 代表的其实就是 window 对象，那么 this.x 就是一个全局变量相当于在外部的一个全局变量。所以 B 输出 3。最后代码由于 f 没有返回值那

么一个函数如果没返回值的话，将会返回 `undefined`，所以答案就是：1, 3, 3, 1, 3, `undefined`。

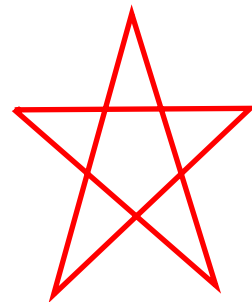
### 111、下面输出多少？

```
var o1 = new Object();
var o2 = o1;
o2.name = "CSSer";
console.log(o1.name);
```

如果不看答案，你回答真确了的话，那么说明你对 javascript 的数据类型了解的还是比较清楚了。**js 中有两种数据类型，分别是：基本数据类型和引用数据类型**（object Array）。**对于保存基本类型值的变量，变量是按值访问的，因为我们操作的是变量实际保存的值。对于保存引用类型值的变量，变量是按引用访问的，我们操作的是变量值所引用（指向）的对象。**答案就清楚了：      //CSSer;      

### 112、再来一个

```
function changeObjectProperty (o) {
    o.siteUrl = "http://www.csser.com/";
    o = new Object();
    o.siteUrl = "http://www.popcg.com/";
}
var CSSer = new Object();
changeObjectProperty(CSSer);
console.log(CSSer.siteUrl); //
```



如果 `CSSer` 参数是按引用传递的，那么结果应该是 `"http://www.popcg.com/"`，但实际结果却仍是 `"http://www.csser.com/"`。事实是这样的：在函数内部修改了引用类型值的参数，该参数值的原始引用保持不变。我们可以把参数想象成局部变量，当参数被重写时，这个变量引用的就是一个局部变量，局部变量的生存期仅限于函数执行的过程中，函数执行完毕，局部变量即被销毁以释放内存。

（补充：内部环境可以通过作用域链访问所有的外部环境中的变量对象，但外部环境无法访问内部环境。每个环境都可以向上搜索作用域链，以查询变量和函数名，反之向下则不能。）

### 113、输出多少？

```
var a = 6;
```

```

setTimeout(function () {
    var a = 666; // 由于变量 a 是一个局部变量
    alert(a);    // 输出 666,
}, 1000);
a = 66;

```

因为 `var a = 666;` 定义了局部变量 `a`，并且赋值为 `666`，根据变量作用域链，全局变量处在作用域末端，优先访问了局部变量，从而覆盖了全局变量。

```

var a = 6;
setTimeout(function () {
    // 变量声明提前
    alert(a);    // 输出 undefined
    var a = 666;
}, 1000);
a = 66;

```

匿名函数里的 `this` 指向 `window` 对象。但是匿名函数内部定义的变量还是局部变量。

因为 `var a = 666;` 定义了局部变量 `a`，同样覆盖了全局变量，但是在 `alert(a);` 之前 `a` 并未赋值，所以输出 `undefined`。

```

var a = 6;

setTimeout(function () {
    alert(a);

    var a = 66;
}, 1000);

a = 666;

alert(a);

// 结果: 666 undefined

```

记住：异步处理，一切 OK 声明提前

## 114、输出多少？

```

function setN(obj) {
    obj.name = '屌丝';
    obj = new Object();
}

```

```

        obj.name = '腐女';
    };
    var per = new Object();
    setN(per);    这个per指向obj 对象
    alert(per.name);    //屌丝 内部

```

## 115、JS 的继承性

```

window.color = 'red';
var o = {color: 'blue'};
function sayColor(){
    alert(this.color);
}

```

考点：1、this 的指向

2、call 的用法

```

sayColor(); //red 这时，this指向window。因为sayColor函数是在全局环境调用的
sayColor.call(this); //red this 指向的是 window 对象
sayColor.call(window); //red
sayColor.call(o); //blue

```

## 116、精度问题: JS 精度不能精确到 0.1 所以 。 。 。 。 同时存

## 在于值和差值中

```

var n = 0.3, m = 0.2, i = 0.2, j = 0.1;
alert((n - m) == (i - j)); //false
alert((n-m) == 0.1); //false
alert((i-j)==0.1); //true

```

## 117、加减运算

```

alert('5'+3); //53 string
alert('5'+ '3'); //53 string
alert('5'-3); //2 number
alert('5'- '3'); //2 number

```



## 118、什么是同源策略？

指： 同协议、端口、域名的安全策略，由网景(Netscape)公司提出来的安全协议！

## 119、call 和 apply 的区别是什么？

参数形式不同，call(obj, pra, pra)后面是单个参数。apply(obj, [args])后面是数组。

## 120、为什么不能定义 1px 左右的 div 容器？

IE6 下这个问题是因为默认的行高造成的，解决的方法也有很多，例如：  
overflow:hidden | zoom:0.08 | line-height:1px

## 121、结果是什么？

```
function foo() {  
    foo.a = function() {alert(1)};  
    this.a = function() {alert(2)};  
    a = function() {alert(3)};  
    var a = function() {alert(4)};  
};  
foo.prototype.a = function() {alert(5)};  
foo.a = function() {alert(6)};  
foo.a(); //6  
var obj = new foo(); 实例化的同时也相当于重新调用构造函数  
obj.a(); //2  
foo.a(); //1
```

## 122、输出结果

```
var a = 5;  
function test() {  
    a = 0;  
    alert(a);  
    alert(this.a); //没有定义 a 这个属性  
    var a;  
    alert(a)
```

```

}
test(); // 0, 5, 0
new test(); // 0, undefined, 0 //由于类它自身没有属性 a, 所以是 undefined

```

### 123、计算字符串字节数:

```

new function(s) {
    if(!arguments.length||!s) return null;
    if(""==s) return 0; //无效代码, 因为上一句!s 已经判断过
    var l=0;
    for(var i=0;i<s.length;i++) {
        if(s.charCodeAt(i)>255) l+=2; else l+=1; //charCodeAt()得到的是 unCode
    } //汉字的 unCode 码大于 255bit 就是两个字节
    alert(l);
}("hello world!");

```

### 124、结果是:

```

var bool = !!2; alert(bool); //true;

```

技巧: 双向非操作可以把字符串和数字转换为布尔值。

### 125、声明对象, 添加属性, 输出属性

```

var obj = {
    name: 'leipeng',
    showName: function() {
        alert(this.name);
    }
}
obj.showName();

```

### 126、匹配输入的字符: 第一个必须是字母或下划线开头, 后面就是字母和数字或者下划线构成, 长度 5-20

```

var reg = /^[a-zA-Z_][a-zA-Z0-9_]{4,19}/,
    name1 = 'leipeng',
    name2 = '0leipeng',
    name3 = '你好 leipeng',
    name4 = 'hi';

```

```
alert(reg.test(name1)); true
alert(reg.test(name2)); false
alert(reg.test(name3)); false
alert(reg.test(name4)); true
```

## 127、检测变量类型

```
function checkStr(str){
    typeof str == 'string'? alert('true'):alert('false');
}
checkStr('leipeng');
```

## 128、如何在 HTML 中添加事件，几种方法？

- 1、标签之中直接添加 `onclick="fun()"`;
- 2、JS 添加 `Eobj.onclick = method`;
- 3、现代事件 IE9 以前：`obj.attachEvent('onclick', method)`;  
标准浏览器：`obj.addEventListener('click', method, false)`;

## 129、BOM 对象有哪些，列举 window 对象？

- 1、`window` 对象，是 JS 的最顶层对象，其他的 BOM 对象都是 `window` 对象的属性；
- 2、`document` 对象，文档对象；
- 3、`location` 对象，浏览器当前 URL 信息；
- 4、`navigator` 对象，浏览器本身信息；
- 5、`screen` 对象，客户端屏幕信息；
- 6、`history` 对象，浏览器访问历史信息；

## 130、请问代码实现 `outerHTML`

```
//说明：outerHTML 其实就是 innerHTML 再加上本身；
Object.prototype.outerHTML = function() {
    var innerCon = this.innerHTML, //获得里面的内容
        outerCon = this.appendChild(innerCon); //添加到里面
    alert(outerCon);
}
```

```
}
```

演示代码：

```
<!doctype html>
```

```
<html>
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<title>Document</title>
```

```
</head>
```

```
<body>
```

```
<div id="outer">
```

```
hello
```

```
</div>
```

```
<script>
```

```
Object.prototype.outerHTML = function() {
```

```
var innerCon = this.innerHTML, //获得里面的内容
```

```
outerCon = this.appendChild(innerCon); //添加到里面
```

```
alert(outerCon);
```

```
}
```

```
function $(id) {
```

```
return document.getElementById(id);
```

```
}
```

```
alert($('outer').innerHTML);
```

```
alert($('outer').outerHTML);
```

```
</script>
```

```
</body>
```

```
</html>
```

这道题是有错误的。最常用的方法是

appendChild(), 用于向 childNodes 列表的末尾添加一个节点。更新完成后, var node=父节点.appendChild(新节点); 返回的是新增的节点, node=新节点。

### 131、JS 中的简单继承 call 方法！

//顶一个父母类，注意：类名都是首字母大写的哦！

```
function Parent(name, money) {
```

```
    this.name = name;
```

```
    this.money = money;
```

```
    this.info = function() {
```

```
        alert('姓名: ' + this.name + ' 钱: ' + this.money);
```

```
    }
```

```
}
```

```

//定义孩子类
function Children(name){
    Parent.call(this, name); //继承 姓名属性，不要钱。
    this.info = function(){
        alert('姓名: ' + this.name);
    }
}

//实例化类
var per = new Parent('parent', 8000000000000);
var chi = new Children('child');
per.info();
chi.info();

```

### 132、bind(), live(), delegate() 的区别 jQuery

**bind:** 绑定事件，对新添加的事件不起作用，方法用于将一个处理程序附加到每个匹配元素的事件上并返回 jQuery 对象。

**live:** 方法将一个事件处理程序附加到与当前选择器匹配的所有元素（包含现有的或将来添加的）的指定事件上并返回 jQuery 对象。

**delegate:** 方法基于一组特定的根元素将处理程序附加到匹配选择器的所有元素（现有的或将来的）的一个或多个事件上。

最佳实现: on() off()

### 133、typeof 的返回类型有哪些？

string, number, object, undefined, boolean, function

```

alert(typeof [1, 2]); //object
alert(typeof 'leipeng'); //string
var i = true;
alert(typeof i); //boolean
alert(typeof 1); //number
var a;
alert(typeof a); //undefined
function a(){};
alert(typeof a) //function

```

### 134、简述 link 和 import 的区别？

区别 1: link 是 XHTML 标签，除了加载 CSS 外，还可以定义 RSS 等其他事务；@import 属于 CSS 范畴，只能加载 CSS。

区别 2: link 引用 CSS 时，在页面载入时同时加载；@import 需要页面网页完全载入以后加载。

区别 3: link 是 XHTML 标签，无兼容问题；@import 是在 CSS2.1 提出的，低版本的浏览器不支持。

区别 4: link 支持使用 Javascript 控制 DOM 去改变样式；而@import 不支持。

### 135、window.onload 和 document.ready 的区别？

load 要等到图片和包含的文件都加载进来之后执行；

ready 是不包含图片和非文字文件的文档结构准备好就执行；

### 136、解析 URL 成一个对象？

```
String.prototype.urlQueryString = function() {
    var url = this.split('?')[1].split('&'),
        len = url.length;

    this.url = {};
    for(var i = 0; i < len; i += 1){
        var cell = url[i].split('='),
            key = cell[0],
            val = cell[1];
        this.url[''+key+''] = val;
    }
    return this.url;
}

var url = '?name=12&age=23';
console.log(url.urlQueryString().age);
```

### 137、看下列代码输出什么？

```
var foo = "11"+2-"1";
console.log(foo);
console.log(typeof foo);
```

执行完后 foo 的值为 111，foo 的类型为 Number。

只有加法运算碰到字符串，会拼接字符串。当遇见减号，会先转化为Number类型，再进行减法运算

138、看下列代码, 输出什么?

```
var a = new Object();
a.value = 1;
b = a;
b.value = 2;
alert(a.value);
```

执行完后输出结果为 2

一个变量指向一个对象，这个变量的值保存的是对象的地址信息。当把这个变量赋值给另外一个变量时，另一个变量得到的也是指向那个对象的地址信息。所以这两个变量都指向同一个对象。

139、已知数组 `var stringArray = ["This", "is", "Baidu", "Campus"]`, Alert 出 "This is Baidu Campus"。

答案: `alert(stringArray.join(" "))`

140、已知有字符串 `foo="get-element-by-id"`, 写一个 function 将其转化成驼峰表示法 `getElementById`。

答案: 

```
function combo(msg) {
    var arr = msg.split("-");
    var len = arr.length;    //将 arr.length 存储在一个局部变量可以提高 for 循环效率
    for(var i=1;i<len;i++) {
        arr[i]=arr[i].charAt(0).toUpperCase()+arr[i].substr(1, arr[i].length-1);
    }
    msg=arr.join("");
    return msg;
}
```

141、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() //添加
removeChild() //移除
replaceChild() //替换
insertBefore() //插入
```

父节点.insertBefore(新节点, 某个子节点) 要把这个新节点放在父节点中某个子节点的前面。

3) 查找

`getElementsByTagName()` //通过标签名称

`getElementsByName()` //通过元素的 Name 属性的值

`getElementById()` //通过元素 Id, 唯一性

142、原生 JS 的 `window.onload` 与 JQuery 的 `$(document).ready(function() {})` 有什么不同?

**`window.onload`是等DOM树和css样式、图片等都加载完成了, 才会被触发执行**

`window.onload()` 方法是必须等到页面内包括图片的所有元素加载完毕后才能执行。

`$(document).ready()` 是 DOM 结构绘制完毕后就执行, 不必等到加载完毕。

**`$(document).ready()`是DOM树结构加载完成后就被触发执行的事件**

143、你如何优化自己的代码?

**代码重用**

**避免全局变量 (命名空间, 封闭空间, 模块化 mvc...)**

**拆分函数避免函数过于臃肿: 单一职责原则**

**适当的注释, 尤其是一些复杂的业务逻辑或者是计算逻辑, 都应该写出这个业务逻辑的具体过程**

**内存管理, 尤其是闭包中的变量释放**

144、请描述出下列代码运行的结果

```
function d() {  
    console.log(this);  
}
```

`d();`//window **函数调用是在全局环境中进行的, 因此this指向window**

145、需要将变量 `e` 的值修改为 “a+b+c+d”, 请写出对应的代码

`var e="abcd";` **递归, 遍历DOM树**

设计一段代码能够遍历下列整个 DOM 节点

```
<div>  
    <p>  
        <span><a/></span>  
        <span><a/></span>  
    </p>  
    <ul>  
        <li></li>  
        <li></li>
```



```
</ul>
</div>
```

146、怎样实现两栏等高？

147、使用 js 实现这样的效果：在文本域里输入文字时，当按下 enter 键时不  
换行，而是替换成 “{{enter}}”，(只需要考虑在行尾按下 enter 键的情况)。

```
textarea.onkeydown=function(e){
    e.preventDefault();//为了阻止 enter 键的默认换行效果
    if(e.keyCode=="enter 键码"){
        testarea.value+="{{enter}}";    this.value+="{{enter}}"
    }
}
```

148、以下代码中 end 字符串什么时候输出

```
var t=true;
setTimeout(function() {
    console.log(123);
    t=false;
},1000);
while(t){}
console.log( 'end' );
```

因为定时器是异步执行的，会被丢到事件队列中。等主线程执行完，才会执行定时器。因此会先执行while，这时t还是true，一直是true，造成死循环。导致定时器中的任务没机会执行。

此时是一个死循环，永远不可能执行 setTimeout 中的回调函数

149、specify( 'hello,world' )//=> ' h, e, l, l, o, w, o, r, l, d' 实现 specify 函数

150、请将一个 URL 的 search 部分参数与值转换成一个 json 对象

```
//search 部分的参数格式: a=1&b=2&c=3
function getJsonFromUrlSearch(search) {
    var item;
    var result={};
    if(search.indexOf('&')<0) {
        item=search.split('=');
    }
```

```

        result[item[0]]=item[1];

        return result;

    }

    var splitArray=search.split('&');

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

        var obj = splitArray[i];

        item=obj.split('=');

        result[item[0]]=item[1];

    }

    return result;

}

var c=getJsonFromUrlSearch("a=1&b=2&c=3");

```

**151、**请用原生 js 实现 jquery 的 get\post 功能，以及跨域情况下

**152、**请简要描述 web 前端性能需要考虑哪方面，你的优化思路是什么？

//参见雅虎 14web 优化规则

//减少 http 请求:

//1、小图弄成大图，2、合理的设置缓存

//3、资源合并、压缩

//将外部的 js 文件置底

## 153、简述 readonly 与 disabled 的区别

readonly 只针对 input(text / password)和 textarea 有效，

而 disabled 对于所有的表单元素都有效，当表单元素在使用了 disabled 后，当我们将表单以 POST 或 GET 的方式提交的话，这个元素的值不会被传递出去，而 readonly 会将该值传递出去

154、判断一个字符串出现次数最多的字符，统计这个次数并输出

155、编写一个方法，去掉一个数组的重复元素

156、写出 3 个使用 this 的典型应用

构造函数中使用 this，原型中使用 this，对象字面量使用 this

157、请尽可能详尽的解释 ajax 的工作原理

**思路：**先解释异步，再解释 ajax 如何使用

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。要清楚这个过程和原理，我们必须对 XMLHttpRequest 有所了解。

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。

158、为什么扩展 javascript 内置对象不是好的做法？

因为扩展内置对象会影响整个程序中所使用到的该内置对象的原型属性

159、请解释一下 javascript 的同源策略

域名、协议、端口相同

160、什么是三元表达式？“三元”表示什么意思？

三目运算

?:

因为运算符会涉及 3 个表达式

161、浏览器标准模式和怪异模式之间的区别是什么？

标准模式是指，浏览器按 W3C 标准解析执行代码；

怪异模式则是使用浏览器自己的方式解析执行代码，因为不同浏览器解析执行的方式不一样，所以我们称之为怪异模式。

浏览器解析时到底使用标准模式还是怪异模式，与你网页中的 DTD 声明直接相关，DTD 声明定义了标准文档的类型（标准模式解析）文档类型，会使浏览器使用相应的方式加载网页并显示，忽略 DTD 声明,将使网页进入怪异模式

162、如果设计中使用了非标准的字体，你该如何去实现？

先通过 font-face 定义字体，再引用

```
@font-face
{
font-family: myFirstFont;
src: url('Sansation_Light.ttf'),
      url('Sansation_Light.eot'); /* IE9+ */
}
```

163、用 css 分别实现某个 div 元素上下居中和左右居中

margin:0 auto;

164、module(12, 5)//2 实现满足这个结果的 modulo 函数

整除获取余数

```
function modulo(a,b) {
    return a%b;//return a/b;
}
```

165、HTTP 协议中，GET 和 POST 有什么区别？分别适用什么场景？

get 传送的数据长度有限制，post 没有

get 通过 url 传递，在浏览器地址栏可见，post 是在报文中传递

适用场景：

post 一般用于表单提交

get 一般用于简单的数据查询，严格要求不是那么高的场景 可应用于获取数据

## 166、HTTP 状态消息 200 302 304 403 404 500 分别表示什么

200: 请求已成功, 请求所希望的响应头或数据体将随此响应返回。

302: 请求的资源临时从不同的 URI 响应请求。由于这样的重定向是临时的, 客户端应当继续向原有地址发送以后的请求。只有在 Cache-Control 或 Expires 中进行了指定的情况下, 这个响应才是可缓存的

304: 如果客户端发送了一个带条件的 GET 请求且该请求已被允许, 而文档的内容 (自上次访问以来或者根据请求的条件) 并没有改变, 则服务器应当返回这个状态码。304 响应禁止包含消息体, 因此始终以消息头后的第一个空行结尾。

403: 服务器已经理解请求, 但是拒绝执行它。

404: 请求失败, 请求所希望得到的资源未被在服务器上发现。

500: 服务器遇到了一个未曾预料的状态, 导致了它无法完成对请求的处理。一般来说, 这个问题都会在服务器端的源代码出现错误时出现。

## 167、HTTP 协议中, header 信息里面, 怎么控制页面失效时间 (last-modified, cache-control, Expires 分别代表什么)

Last-Modified	文档的最后改动时间。客户可以通过 If-Modified-Since 请求头提供一个日期, 该请求将被视为一个条件 GET, 只有改动时间迟于指定时间的文档 才会返回, 否则返回一个 304(Not Modified) 状态。Last-Modified 也可用 setDateHeader 方法来设置。
Expires	应该在什么时候认为文档已经过期, 从而不再缓存它?

## 168、HTTP 协议目前常用的有哪几个? KEEPALIVE 从哪个版本开始出现的?

http1.0

http1.1 keepalive

## 169、业界常用的优化 WEB 页面加载速度的方法 (可以分别从页面元素展现, 请求连接, css, js, 服务器等方面介绍)

## 170、列举常用的 web 页面开发, 调试以及优化工具

sublime vscode webstorm hbuilder dw

httpwatch=>ie

ff:firebug

chrome:

171、解释什么是 sql 注入，xss 漏洞

172、如何判断一个 js 变量是数组类型

```
ES5:Array.isArray()  
[] instanceof Array  
Object.prototype.toString.call([]);//[object Array]"
```

173、请列举 js 数组类型中的常用方法

方法	描述
<u><a href="#">concat()</a></u>	连接两个或更多的数组，并返回结果。 将多个数组拼接成一个数组，返回值是拼接后的新数组
<u><a href="#">join()</a></u>	把数组的所有元素放入一个字符串。元素通过指定的分隔符进行分隔。
<u><a href="#">pop()</a></u>	删除并返回数组的最后一个元素
<u><a href="#">push()</a></u>	向数组的末尾添加一个或更多元素，并返回新的长度。
<u><a href="#">reverse()</a></u>	颠倒数组中元素的顺序。
<u><a href="#">shift()</a></u>	删除并返回数组的第一个元素
<u><a href="#">slice()</a></u>	从某个已有的数组返回选定的元素
<u><a href="#">sort()</a></u>	对数组的元素进行排序
<u><a href="#">splice()</a></u>	删除元素，并向数组添加新元素。
<u><a href="#">toSource()</a></u>	返回该对象的源代码。
<u><a href="#">toString()</a></u>	把数组转换为字符串，并返回结果。
<u><a href="#">toLocaleString()</a></u>	把数组转换为本地数组，并返回结果。 国际化
<u><a href="#">unshift()</a></u>	向数组的开头添加一个或更多元素，并返回新的长度。
<u><a href="#">valueOf()</a></u>	返回数组对象的原始值

174、FF 与 IE 中如何阻止事件冒泡，如何获取事件对象，以及如何获取触发事件的元素

175、列举常用的 js 框架以及分别适用的领域

jquery: 简化了 js 的一些操作，并且提供了一些非常好用的 API

jquery ui、jquery-easyui: 在 jquery 的基础上提供了一些常用的组件 日期, 下拉框, 表格这些组件

require.js、sea.js (阿里的玉帛) + 》模块化开发使用的

zepto: 精简版的 jquery, 常用于手机 web 前端开发 提供了一些手机页面实用功能,touch

ext.js: 跟 jquery 差不多, 但是不开源, 也没有 jquery 轻量

angular、knockoutjs、avalon(去哪儿前端总监): MV\*框架, 适合用于单页应用开发(SPA)

## 176、js 中如何实现一个 map

数组的 map 方法:

### 概述

**map()** 方法返回一个由原数组中的每个元素调用一个指定方法后的返回值组成的新数组。

**map()**: 对数组中的每一项运行给定函数, 返回每次函数调用的结果组成的数组。

### 语法

`array.map(callback[, thisArg])`

### 参数

#### callback

原数组中的元素经过该方法后返回一个新的元素。

#### 回调函数的参数

##### currentValue

callback 的第一个参数, 数组中当前被传递的元素。

##### index

callback 的第二个参数, 数组中当前被传递的元素的索引。

##### array

callback 的第三个参数, 调用 map 方法的数组。

#### thisArg

执行 callback 函数时 this 指向的对象。

实现:

```
Array.prototype.map2=function(callback){
    这个函数是数组原型对象的属性
    for (var i = 0; i < this.length; i++) {
        原型对象的成员函数中的this指向实例对象
        this[i]=callback(this[i]);
    }
}
```

```
}  
  
};
```

177、js 可否实现面向对象编程，如果可以如何实现 js 对象的继承

创建对象的几种方式

实现继承的几种方式

原型链

178、约瑟夫环—已知 n 个人（以编号 1，2，3…分别表示）围坐在一张圆桌周围。从编号为 k 的人开始报数，数到 m 的那个人出列；他的下一个人又从 1 开始报数，数到 m 的那个人又出列；依此规律重复下去，直到圆桌周围的人全部出列。

179、有 1 到 10w 这个 10w 个数，去除 2 个并打乱次序，如何找出那两个数？

**180、如何获取对象 a 拥有的所有属性（可枚举的、不可枚举的，不包括继承来的属性）**

for in 可以用来枚举对象的属性，但是不能过滤出继承的属性。只有 obj.hasOwnProperty 才可以过滤出非继承属性和继承属性

Object.keys——IE9+      obj.hasOwnProperty("属性名")，返回值是布尔类型

或者使用 for...in 并过滤出继承的属性

for(o in obj){      使用 hasOwnProperty() 方法可以检测一个属性是存在于实例中，还是存在于原型中。这个方法（不要忘了它是从 Object 继承来的）只在给定属性存在于对象实例中时，才会返回 true。

```
    if(obj.hasOwnProperty(o)){  
        //把 o 这个属性放入到一个数组中  
    }  
}
```

181、有下面这样一段 HTML 结构，使用 css 实现这样的效果：

左边容器无论宽度如何变动，右边容器都能自适应填满父容器剩余的宽度。

```
<div class="warp" >  
<div class="left" ></div>  
<div class="right" ></div>  
</div>
```



182、下面这段代码想要循环输出结果 01234，请问输出结果是否正确，如果不正确，请说明为什么，并修改循环内的代码使其输出正确结果

```
for(var i=0;i<5;++i){
    setTimeout(function(){
        console.log(i+' ');
    },100*i);
}
```

183、解释下这个 css 选择器什么发生什么？

```
[role=nav]>ul a:not([href^=mailto]) {}
```

184、JavaScript 以下哪条语句会产生运行错误

A. var obj = ();    B. var obj = [];    C. var obj = {};    D. var obj = //;

答案：AD

185、以下哪些是 javascript 的全局函数：（ABCDE）

A. escape    函数可对字符串进行编码，这样就可以在所有的计算机上读取该字符串。  
ECMAScript v3 反对使用该方法，应用使用 decodeURI() 和 decodeURIComponent() 替代它。

B. parseFloat    parseFloat() 函数可解析一个字符串，并返回一个浮点数。

该函数指定字符串中的首个字符是否是数字。如果是，则对字符串进行解析，直到到达数字的末端为止，然后以数字返回该数字，而不是作为字符串。

C. eval    函数可计算某个字符串，并执行其中的的 JavaScript 代码。

D. setTimeout

E. alert

186、关于 IE 的 window 对象表述正确的有：（CD）

A. window.opener 属性本身就是指向 window 对象  
window.opener 返回打开当前窗口的那个窗口的引用。

如果当前窗口是由另一个窗口打开的，**window.opener** 保留了那个窗口的引用。如果当前窗口不是由其他窗口打开的，则该属性返回 null。

B. `window.reload()` 方法可以用来刷新当前页面 //正确答案: 应该是 `location.reload` 或者 `window.location.reload`

C. `window.location="a.html"` 和 `window.location.href="a.html"` 的作用都是把当前页面替换成 `a.html` 页面

D. 定义了全局变量 `g`; 可以用 `window.g` 的方式来存取该变量

## 187、描述错误的是 D

A: Http 状态码 302 表示暂时性转移 对

B: `DOMContentLoaded` 事件早于 `onload` 事件 //正确

当 `onload` 事件触发时, 页面上所有的 DOM, 样式表, 脚本, 图片, flash 都已经加载完成了。

当 `DOMContentLoaded` 事件触发时, 仅当 DOM 加载完成, 不包括样式表, 图片, flash。

C: IE678 不支持事件捕获

D: `localStorage` 存储的数据在电脑重启后丢失 //错误, 因为没有时间限制

`try...catch` 语句。(在 IE5+、Mozilla 1.0、和 Netscape 6 中可用)

## 188、关于 `link` 和 `@import` 的区别正确的是 A

A: `link` 属于 XHTML 标签, 而 `@import` 是 CSS 提供的;

B: 页面被加载时, `link` 会同时被加载, 而后者引用的 CSS 会等到页面被加载完再加载

C: `import` 只在 IE5 以上才能识别 而 `link` 是 XHTML 标签, 无兼容问题

D: `link` 方式的样式的权重高于 `@import` 的权重

## 189、下面正确的是 A

A: 跨域问题能通过 `JsonP` 方案解决

B: 不同子域名间仅能通过修改 window.name 解决跨域 //还可以通过 script 标签 src jsonp

C: 只有在 IE 中可通过 iframe 嵌套跨域 //任何浏览器都可以使用 iframe

D: MediaQuery 属性是进行视屏格式检测的属性是做响应式的

## 188、错误的是：AC

~~A: Ajax 本质是 XMLHttpRequest //异步请求 json 和 xml 数据~~

B: 块元素实际占用的宽度与它的 width、border、padding 属性有关，与 background 无关

~~C: position 属性 absolute、fixed、relative 会使文档脱标~~  
absolute和fixed会使元素脱离文档流。relative不会使元素脱离文档流

D: float 属性 left 也会使 div 脱标

## 189、不用任何插件，如何实现一个 tab 栏切换？

通过改变不同层的 css 设置层的显示和隐藏

190、基本数据类型的专业术语以及单词拼写

191、变量的命名规范以及命名推荐

## 192、三种弹窗的单词以及三种弹窗的功能

alert

confirm

prompt

P205

193、console.log( 8 | 1 ); 输出值是多少？

答案：9

194、只允许使用 `+-*/` 和 `Math.*`，求一个函数 `y = f(x, a, b)`；当 `x > 100` 时返回 `a` 的值，否则返回 `b` 的值，不能使用 `if else` 等条件语句，也不能使用 `|,?:`，数组。

答案：

```
function f(x, a, b) {  
  
    var temp = Math.ceil(Math.min(Math.max(x - 100, 0), 1));  
  
    return a * temp + b * (1 - temp);  
  
}  
  
console.log(f(-10, 1, 2));
```

195、JavaScript `alert(0.4*0.2)`；结果是多少？和你预期的一样吗？如果不一样该如何处理？

有误差，应该比准确结果偏大。一般我会将小数变为整数来处理。当前之前遇到这个问题时也上网查询发现有人用 `try catch return` 写了一个函数，

当然原理也是一致先转为整数再计算。看起来挺麻烦的，我没用过。

196、一个 `div`，有几种方式得到这个 `div` 的 `jQuery` 对象？`<div class='aabbcc' id='nodesView'></div>`想直接获取这个 `div` 的 `dom` 对象，如何获取？`dom` 对象如何转化为 `jQuery` 对象？

```
var domView=document.getElementById("nodesView")  
document.getElementsByClassName("aabbcc");  
document.querySelector(".aabbcc#nodesView");
```

转换为 `jquery` 对象： `$( domView)`

197、主流浏览器内核

IE trident 火狐 gecko 谷歌苹果 webkit

Opera: Presto

198、如何显示/隐藏一个 `dom` 元素？请用原生的 JavaScript 方法实现

```
dom.style.display="none";
```

```
dom.style.display="";
```

## 199、JavaScript 有哪几种数据类型

Number String Boolean Null Undefined Object

## 200、jQuery 框架中\$.ajax()的常用参数有哪些？

### type

类型：String

默认值："GET")。请求方式 ("POST" 或 "GET")，默认为 "GET"。注意：其它 HTTP 请求方法，如 PUT 和 DELETE 也可以使用，但仅部分浏览器支持。

### url

类型：String

默认值：当前页地址。发送请求的地址。

### success

类型：Function

请求成功后的回调函数。

参数：由服务器返回，并根据 dataType 参数进行处理后的数据；描述状态的字符串。

这是一个 Ajax 事件。

### options

类型：Object

可选。AJAX 请求设置。所有选项都是可选的。

### async

类型：Boolean

默认值：true。默认设置下，所有请求均为异步请求。如果需要发送同步请求，请将此选项设置为 false。

注意，同步请求将锁住浏览器，用户其它操作必须等待请求完成才可以执行。

### beforeSend(XHR)

类型：Function

发送请求前可修改 XMLHttpRequest 对象的函数，如添加自定义 HTTP 头。

XMLHttpRequest 对象是唯一的参数。

这是一个 Ajax 事件。如果返回 `false` 可以取消本次 ajax 请求。

## cache

类型: Boolean

默认值: `true`, `dataType` 为 `script` 和 `jsonp` 时默认为 `false`。设置为 `false` 将不缓存此页面。

jQuery 1.2 新功能。

## contentType

类型: String

默认值: `"application/x-www-form-urlencoded"`。发送信息至服务器时内容编码类型。

默认值适合大多数情况。如果你明确地传递了一个 `content-type` 给 `$.ajax()` 那么它必定会发送给服务器（即使没有数据要发送）。

## data

类型: String

发送到服务器的数据。将自动转换为请求字符串格式。GET 请求中将附加在 URL 后。查看 `processData` 选项说明以禁止此自动转换。必须为 `Key/Value` 格式。如果为数组, jQuery 将自动为不同值对应同一个名称。如 `{foo:["bar1", "bar2"]}` 转换为 `'&foo=bar1&foo=bar2'`。

## dataFilter

类型: Function

给 Ajax 返回的原始数据的进行预处理的函数。提供 `data` 和 `type` 两个参数: `data` 是 Ajax 返回的原始数据, `type` 是调用 `jQuery.ajax` 时提供的 `dataType` 参数。函数返回的值将由 jQuery 进一步处理。

## dataType

类型: String

预期服务器返回的数据类型。如果不指定, jQuery 将自动根据 HTTP 包 MIME 信息来智能判断, 比如 XML MIME 类型就被识别为 XML。在 1.4 中, JSON 就会生成一个 JavaScript 对象, 而 `script` 则会执行这个脚本。随后服务器端返回的数据会根据这个值解析后, 传递给回调函数。可用值:

- `"xml"`: 返回 XML 文档, 可用 jQuery 处理。
- `"html"`: 返回纯文本 HTML 信息; 包含的 `script` 标签会在插入 dom 时执行。
- `"script"`: 返回纯文本 JavaScript 代码。不会自动缓存结果。除非设置了 `"cache"` 参数。  
注意: 在远程请求时(不在同一个域下), 所有 POST 请求都将转为 GET 请求。(因为将使用 DOM 的 `script` 标签来加载)
- `"json"`: 返回 JSON 数据。
- `"jsonp"`: JSONP 格式。使用 JSONP 形式调用函数时, 如 `"myurl?callback=?"` jQuery 将自动替换 `?` 为正确的函数名, 以执行回调函数。

- "text": 返回纯文本字符串

## error

类型: Function

默认值: 自动判断 (xml 或 html)。请求失败时调用此函数。

有以下三个参数: XMLHttpRequest 对象、错误信息、(可选) 捕获的异常对象。

如果发生了错误, 错误信息 (第二个参数) 除了得到 null 之外, 还可能是 "timeout", "error", "notmodified" 和 "parsererror"。

这是一个 Ajax 事件。

## 写一个 post 请求并带有发送数据和返回数据的样例

```
$.ajax({  
    url:"1.html",  
    data:{name:"张三",age:18}, //post 数据  
    dataType:"json",  
    type:"POST",  
    success:function(data) {  
        //data: 返回的数据  
    },  
    error:function() {  
        //异常处理  
    }  
});
```

201、JavaScript 数组元素添加、删除、排序等方法有哪些?

Array.concat() 连接数组

`Array.join()` 将数组元素连接起来以构建一个字符串

`Array.length` 数组的大小

`Array.pop()` 删除并返回数组的最后一个元素

`Array.push()` 给数组添加元素

`Array.reverse()` 颠倒数组中元素的顺序

`Array.shift()` 将元素移出数组

`Array.slice()` 返回数组的一部分

`Array.sort()` 对数组元素进行排序

`Array.splice()` 插入、删除或替换数组的元素

`Array.toLocaleString()` 把数组转换成局部字符串

`Array.toString()` 将数组转换成一个字符串

`Array.unshift()` 在数组头部插入一个元素

## 202、如何添加 html 元素的事件，有几种方法？请列举

a、直接在标签里添加：`<div onclick="alert(你好)">这是一个层</div>`

b、在元素上通过 js 添加：

c、使用事件注册函数添加

## 203、JavaScript 的循环语句有哪些？

`while` `for` `do while` `for...in`

## 204、作用域-编译期执行期以及全局局部作用域问题

理解 js 执行主要的两个阶段：预解析和执行期

## 205、闭包：下面这个 ul，如何点击每一列的时候 alert 其 index？

```
<ul id="test">
```

```
<li>这是第一条</li>
```

```
<li>这是第二条</li>
```

```
<li>这是第三条</li>
```

```
</ul>
```

```
// 非闭包实现
```



```
var lis=document.querySelectorAll('li');
document.querySelector('#test').onclick=function(e) {
    for (var i = 0; i < lis.length; i++) {
        var li = lis[i];
        if(li==e.target){
            alert(i);
        }
    }
};
```

//闭包实现

```
var lis=document.querySelectorAll('li');
for (var i = 0; i < lis.length; i++) {
    var li = lis[i];
    li.onclick=(function(index){
        return function(e) {
            alert(index);
        };
    })(i);
}
```

## 206、列出 3 条以上 ff 和 IE 的脚本兼容问题

- 1、在 IE 下可通过 `document.frames["id"]`;得到该 IFRAME 对象，而在火狐下则是通过 `document.getElementById("content_panel_if").contentWindow`;
- 2、IE 的写法: `_tbody=_table.childNodes[0]`  
在 FF 中，firefox 会在子节点中包含空白则第一个子节点为空白""，而 ie 不会返回空白  
可以通过 `if("" != node.nodeName)`过滤掉空白子对象
- 3、模拟点击事件

```
if(document.all){ //ie 下
    document.getElementById("a3").click();
}
else{ //非 IE
    var evt = document.createEvent("MouseEvents");
    evt.initEvent("click", true, true);
    document.getElementById("a3").dispatchEvent(evt);
}
4、事件注册
if (isIE){window.w.attachEvent("onload", init);}else{window.addEventListner("load", init, false);}
```

## 207、列举可以哪些方面对前端开发进行优化

代码压缩、合并减少 http 请求，图片制作精灵图、代码优化

## 208、至少列出一种 JavaScript 继承的实现方式

**209、如现在有一个效果，有显示用户头像、用户昵称、用户其他信息；当用户鼠标移到头像上时，会弹出用户的所有信息；如果是你，你会如何实现这个功能，请用代码实现？**

//答案见: J:\代码,PPT,笔记,电子书\面试题\面试题 02.html

## 210、call 与 apply 有什么作用？又有什么什么区别？用 callee 属性实现函数递归？

P115 arguments.callee 指代当前函数，在递归时可以使用

apply 的参数是数组,call 的参数是单个的值，除此之外，两者没有差别，重点理解 this 的改变，callee 已经不推荐使用

## 211、用正则表达式，写出由字母开头，其余由数字、字母、下划线组成的 6~30 的字符串？

```
var reg=/^[a-zA-Z][\da-zA-Z_]{5,29}/;
```

## 212、列举浏览器对象模型 BOM 里常用的至少 4 个对象，并列举 window 对象的常用方法至少 5 个（10 分）

对象: window document location screen history navigator

方法: alert() confirm() prompt() open() close() setInterval() setTimeout()  
clearInterval() clearTimeout()

(详细参见: J:\代码, PPT, 笔记, 电子书\面试题>window 对象方法.png)

## 213、Javascript 中 callee 和 caller 的作用？

caller 是返回一个对函数的引用，该函数调用了当前函数：

用法: fn.caller

callee 是返回正在被执行的 function 函数，也就是所指定的 function 对象的正文。

用法: arguments.callee

## 214、对于 apply 和 call 两者在作用上是相同的，即是调用一个对象的一个方法，以另一个对象替换当前对象。将一个函数的对象上下文从初始的上下文改变为由 thisObj 指定的新对象。

但两者在参数上有区别的。对于第一个参数意义都一样，但对第二个参数：?apply 传入的是一个参数数组，也就是将多个参数组合成为一个数组传入，而 call 则作为 call 的参数传入（从第二个参数开始）。?如 func.call(func1, var1, var2, var3) 对应的 apply 写法为：func.apply(func1, [var1, var2, var3])。

## 215、在 Javascript 中什么是伪数组？如何将伪数组转化为标准数组？

伪数组（类数组）：无法直接调用数组方法或期望 length 属性有什么特殊的行为，但仍可以对真正数组遍历方法来遍历它们。典型的是函数的 argument 参数，还有像调用

getElementsByTagName, document.childNodes 之类的, 它们都返回 NodeList 对象都属于伪数组。

可以使用 `Array.prototype.slice.call(fakeArray)` 将数组转化为真正的 Array 对象。

216、写一个函数可以计算 `sum(5, 0, -5)`; 输出 0; `sum(1, 2, 3, 4)`; 输出 10;

```
function calc() {  
    var result=0;  
    for (var i = 0; i < arguments.length; i++) {  
        var obj = arguments[i];  
        result+=obj;  
    }  
    return result;  
}  
  
alert(calc(1,2,3,4));
```

Js 基本功

217、事件代理怎么实现？

在元素的父节点注册事件，通过事件冒泡，在父节点捕获事件

**218、《正则》写出正确的正则表达式匹配固话号，区号 3-4 位，第一位为 0，中横线，7-8 位数字，中横线，3-4 位分机号格式的固话号**

常用正则表达式语法要熟悉  
`/0[0-9]{2,3}-\d{7,8}/`

219、《算法》 一下 A,B 可任选一题作答，两题全答加分

A:农场买了一只羊，第一年是小羊，第二年底生一只，第三年不生，第四年底再生一只，第五年死掉。

**B: 写出代码对下列数组去重并从大到小排列**  
**{5,2,3,6,8,6,5,4,7,1,9}**

先去重再排序

去重方法参考: J:\代码,PPT,笔记,电子书\面试题

220、请写出一张图片的 HTML 代码，已知道图片地址为 “images/abc. jpg” ,宽 100px，高 50px

221、请写一个正则表达式：要求最短 6 位数，最长 20 位，阿拉伯数和英文字母（不区分大小写）组成

`^(?=.*\d)(?=.*[a-z])(?=.*[A-Z])[a-zA-Z\d]{6,20}$`

**222、统计 1 到 400 亿之间的自然数中含有多少个 1？比如 1-21**

**中，有 1、10、11、12、13、14、15、16、17、18、19、20、21**

**这么多自然数有 13 个 1**

答案参考: J:\代码,PPT,笔记,电子书\面试题\面试题\_222.html

223、删除与某个字符相邻且相同的字符，比如 fdaffdaaklfjklja 字符串处理之后成为 “fdafdaklfjklja”

答案参考: J:\代码,PPT,笔记,电子书\面试题\面试题\_223.html

224、请写出三种以上的 Firefox 有但 InternetExplorer 没有的属性和函数

- 1、在 IE 下可通过 `document.frames["id"]`;得到该 IFRAME 对象，而在火狐下则是通过 `document.getElementById("content_panel_if").contentWindow`;
- 2、IE 的写法: `_tbody=_table.childNodes[0]`

在 FF 中，firefox 会在子节点中包含空白则第一个子节点为空白""，而 ie 不会返回空白  
可以通过 `if("" != node.nodeName)` 过滤掉空白子对象

### 3、模拟点击事件

```
if(document.all){ //ie 下
    document.getElementById("a3").click();
}
else{ //非 IE
    var evt = document.createEvent("MouseEvents");
    evt.initEvent("click", true, true);
    document.getElementById("a3").dispatchEvent(evt);
}
```

### 4、事件注册

```
if (isIE){window.attachEvent("onload", init);}else{window.addEventListener("load", init, false);}
```

**225、请写出一个程序，在页面加载完成后动态创建一个 form 表单，并在里面添加一个 input 对象并给它任意赋值后以 post 方式提交到：<http://127.0.0.1/save.php>**

答案参考：J:\代码, PPT, 笔记, 电子书\面试题\面试题\_225.html

**226、用 JavaScript 实现冒泡排序。数据为 23、45、18、37、92、13、24**

面试经常遇到的排序，查找算法要熟悉

**227、解释一下什么叫闭包，并实现一段闭包代码**

简单理解就是函数的嵌套形成闭包，闭包包括函数本身及其外部作用域

**228、简述一下什么叫事件委托以及其原理**

在元素的父节点注册事件，通过事件冒泡，在父节点捕获事件

**229、前端代码优化的方法**

```
var User = { 对象
    count = 1, 属性
    getCount: function () { 方法
        return this.count;
    }
}
```

```
}  
  
console.log(User.getCount());  
  
var func = User.getCount;  
  
console.log(func());  
  
1 undefined (window);
```

### 230、下列 JavaScript 代码执行后，依次 alert 的结果是

```
(function test() {  
    var a=b=5;  
    alert(typeof a);  
    alert(typeof b);  
})();  
  
alert(typeof a);  
  
alert(typeof b);  
  
//number number undefined number
```

### 231、下列 JavaScript 代码执行后，iNum 的值是

```
var iNum = 0;  
  
for(var i = 1; i < 10; i++) {  
    if(i % 5 == 0) {  
        continue;  
    }  
  
    iNum++;  
}  
  
}
```

分析：

i=1 1

i=2 2

```
i=3 3  
i=4 4  
i=5  
i=6 6  
i=7 7  
i=8 8  
i=9 9
```

## 232、输出结果是多少？

```
1)  var a;  
var b = a * 0;  
if (b == b) {  
    console.log(b * 2 + "2" - 0 + 4);  
}  
else {  
    console.log(!b * 2 + "2" - 0 + 4);  
}  
}
```

答案：26

扩展：关于乘法操作符：J:\代码, PPT, 笔记, 电子书\面试题\乘性操作符.png

```
2)  <script>  
    var a = 1;  
</script>  
<script>  
var a;  
var b = a * 0;  
if (b == b) {  
    console.log(b * 2 + "2" - 0 + 4);  
}  
else {
```



```
        console.log(!b * 2 + "2" - 0 + 4);
    }
</script>
```

答案: 6

3) var t = 10;

```
function test(t){
    var t = t++;//此时的 t 是一个局部变量，全局变量没有任何变化
    console.log(t);//此时的结果又是多少?
}test(t);
console.log(t);
```

答案: 10

4) var t = 10;

```
function test(test){
    var t = test++;
}test(t);
console.log(t);
```

答案: 10

6) var t = 10;

```
function test(test){
    t = test++;
}test(t);
console.log(t);
```

答案: 10

7) var t = 10;

```
function test(test){
    t = t + test;//undefined+10=NaN
    console.log(t);
    var t = 3;
```

```
}test(t);
```

```
console.log(t);
```

答案: NaN 10

8) var a;

```
var b = a / 0;
```

```
if (b == b) { //b=NaN
```

```
    console.log(!b * 2 + "2" - 0 + 4);
```

```
} else {
```

```
    console.log(!b * 2 + "2" - 0 + 4);
```

```
}
```

答案: 26

9) <script>

```
    var a = 1;
```

```
</script>
```

```
<script>
```

```
    var a;
```

```
    var b = a / 0;
```

```
    if (b == b) { //b=Infinity
```

```
        console.log(b * 2 + "2" + 4);
```

```
    } else {
```

```
        console.log(!b * 2 + "2" + 4);
```

```
    }
```

```
</script>
```

答案: Infinity24

**233、用程序实现找到 html 中 id 名相同的元素?**

```
<body>
```

```
<form id='form1'>
```

```
<div id='div1'></div>

<div id='div2'></div>

<div id='div3'></div>

<div id='div4'></div>

<div id='div5'></div>

<div id='div3'>id 名重复的元素</div>

</form>

</body>
```

### 234、下列 JavaScript 代码执行后，运行的结果是

```
<button id='btn'>点击我</button>

var btn = document.getElementById('btn');

var handler = {

    id: '_eventHandler',

    exec: function() {

        alert(this.id);

    }

}

btn.addEventListener('click', handler.exec);
```

答案：btn，因为 handler.exec 是由 btn 这个按钮执行的

### 235、☆☆☆下列 JavaScript 代码执行后，依次 alert 的结果是

```
var obj = {proto: {a:1,b:2}};

function F() {};

F.prototype = obj.proto;

var f = new F();

obj.proto.c = 3;
```

```
obj.proto = {a:-1, b:-2};  
  
alert(f.a);//1  
  
alert(f.c);//3  
  
delete F.prototype['a'];  
  
alert(f.a);//undefined  
  
alert(obj.proto.a);//-1
```

## 236、下列 JavaScript 代码执行后的效果是

```
<ul id='list'>  
  
<li>item</li>  
  
<li>item</li>  
  
<li>item</li>  
  
<li>item</li>  
  
<li>item</li>  
  
</ul>  
  
var items = document.querySelectorAll('#list>li');  
for(var i = 0;i < items.length; i++){  
    setTimeout(function() {  
        items[i].style.backgroundColor = '#fee';  
    }, 5);  
}
```

答案：异常

## 237、下列 JavaScript 代码执行后的 li 元素的数量是

```
<ul>  
  
<li>Item</li>  
  
<li></li>  
  
<li></li>  
  
<li>Item</li>
```

```

<li>Item</li>

</ul>

var items = document.getElementsByTagName('li');

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

    if(items[i].innerHTML == ''){

        items[i].parentNode.removeChild(items[i]);

    }

}

```

答案：4 个

### 238、程序中捕获异常的方法？

```

window.error

try {} catch() {} finally {}

```

### 239、将字符串” <tr><td>{\$id}</td><td>{\$name}</td></tr>” 中的{\$id} 替换成 10，{\$name} 替换成 Tony （使用正则表达式）

答案： ” <tr><td>{\$id}</td><td>{\$id}\_{\$name}</td></tr>” .replace(/{\\$id}/g, '10' ).replace(/{\\$name}/g, 'Tony' );

### 240、给 String 对象添加一个方法，传入一个 string 类型的参数，然后将 string 的每个字符间价格空格返回，例如：

```

addSpace( “hello world” ) // -> ‘h e l l o ?w o r l d’

String.prototype.spacify = function(){

return this.split('').join(' ');

};

```

### 241、写出函数 DateDemo 的返回结果，系统时间假定为今天

```

function DateDemo() {

    var d, s="今天日期是： ";

    d = new Date();

    s += d.getMonth() + "/";

    s += d.getDate() + "/";

    s += d.getFullYear();
}

```

```
    return s;
}
```

结果：今天日期是：7/17/2010

242、输出今天的日期，以 YYYY-MM-DD 的方式，比如今天是 2014 年 9 月 26 日，则输出 2014-09-26

```
var d = new Date();

// 获取年，getFullYear() 返回 4 位的数字
var year = d.getFullYear();

// 获取月，月份比较特殊，0 是 1 月，11 是 12 月
var month = d.getMonth() + 1;

// 变成两位
month = month < 10 ? '0' + month : month;

// 获取日
var day = d.getDate();

day = day < 10 ? '0' + day : day;

alert(year + '-' + month + '-' + day);
```

243、已知数组 `var?stringArray?=?[“This”,?“is”,?“Baidu”,?“Campus”]`，Alert 出 “This?is?Baidu?Campus”。

答案：`alert(stringArray.join(“”))`

244、已知有字符串 `foo=“get-element-by-id”`，写一个 function 将其转化成驼峰表示法 “getElementById”。

```
function combo(msg) {
    var arr=msg.split("-");
    for(var i=1;i<arr.length;i++){
        arr[i]=arr[i].charAt(0).toUpperCase()+arr[i].substr(1,arr[i].length-1);
    }
    msg=arr.join("");
    return msg;
}
```

## 245、. var numberArray=[3, 6, 2, 4, 1, 5]; （考察基础 API）

1) 实现对该数组的倒排，输出 [5, 1, 4, 2, 6, 3]

2) 实现对该数组的降序排列，输出 [6, 5, 4, 3, 2, 1]

```
function combo(msg) {  
    var arr=msg.split("-");  
    for(var i=1;i<arr.length;i++){  
        arr[i]=arr[i].charAt(0).toUpperCase()+arr[i].substr(1,arr[i].length-1);  
    }  
    msg=arr.join("");  
    return msg;  
}
```

## 246、把两个数组合并，并删除第二个元素。

```
var array1 = ['a','b','c'];  
var bArray = ['d','e','f'];  
var cArray = array1
```

答案：

```
array1=array1.concat(bArray)  
array1.splice(1,1)
```

## 247、如何消除一个数组里面重复的元素？

```
var arr=[1, 2, 3, 3, 4, 4, 5, 5, 6, 1, 9, 3, 25, 4];  
function deRepeat() {  
    var newArr=[];  
    var obj={};  
    var index=0;  
    var l=arr.length;  
    for(var i=0;i<l;i++){  
        if(obj[arr[i]]==undefined)
```

```

{
obj[arr[i]]=1;

newArr[index++]=arr[i];
}

else if(obj[arr[i]]==1)
}

return newArr;
}

var newArr2=deRepeat(arr);

alert(newArr2); //输出 1,2,3,4,5,6,9,25

```

**248、用 js 实现随机选取 10 - 100 之间的 10 个数字，存入一个数组，并排序。**

```

var iArray = [];

function getRandom(istart, iend){

var iChoice = istart - iend +1;

return Math.floor(Math.random() * iChoice + istart;

}

for(var i=0; i<10; i++){

iArray.push(getRandom(10,100));

}

iArray.sort();

```

**249、正则表达式构造函数 var reg=new RegExp(“xxx”)与正则表达字面量 var reg=//有什么不同？匹配邮箱的正则表达式？**

答案：当使用 RegExp() 构造函数的时候，不仅需要转义引号（即\”表示”），并且还需要双反斜杠（即\\表示一个\）。使用正则表达字面量的效率更高。？

**250、1 var regMail =  
/^[a-zA-Z0-9\_-]+@[a-zA-Z0-9\_-]+((.[a-zA-Z0-9\_-]{2,3}){1,2})\$/;**

正则表达式对象 3 - 清除空格

写一个 function，清除字符串前后的空格。（兼容所有浏览器）



使用自带接口 trim(), 考虑兼容性:

```
if (!String.prototype.trim) {  
  
String.prototype.trim = function() {  
  
return this.replace(/^\s+/, "").replace(/\s+$/, "");  
  
} }  
  
// test the function  
  
var str = " \t\n test string ".trim();  
  
alert(str == "test string"); // alerts "true"
```

## 251、数组和字符串

```
<script lang="JavaScript" type="text/javascript">  
  
    function outPut(s) {  
  
        document.writeln(s);  
  
    }  
  
    var a = "lashou";  
  
    var b = a;  
  
    outPut(b);  
  
    a = "拉手";  
  
    outPut(a);  
  
    outPut(b);  
  
    var a_array = [1, 2, 3];  
  
    var b_array = a_array;  
  
    outPut(b_array);  
  
    a_array[3] = 4;  
  
    outPut(a_array);  
  
    outPut(b_array);  
  
</script>
```

输出结果:

答案: lashou 拉手 lashou 1,2,3 1,2,3,4 1,2,3,4

## 252、下列控制台都输出什么

第 1 题:

```
function setName() {  
    name="张三";  
}
```

```
setName();
```

```
console.log(name);
```

答案: "张三"

## 253、第 2 题:

//考点: 1、变量声明提升 2、变量搜索机制

```
var a=1;
```

```
function test() {  
    console.log(a);  
    var a=1;  
}
```

```
test();
```

答案: undefined

## 254、第 3 题:

```
var b=2;
```

```
function test2() {  
    window.b=3;  
    console.log(b);  
}
```

```
test2();
```

答案: 3

## 255、第 4 题:

```
c=5;//声明一个全局变量 c
```

```
function test3() {
```

```
    window.c=3;
```

```
    console.log(c);    //答案: undefined, 原因: 由于此时的 c 是一个局部变量 c,
```

并且没有被赋值

```
    var c;
```

```
    console.log(window.c);//答案: 3, 原因: 这里的 c 就是一个全局变量 c
```

```
}  
test3();
```

### 256、第 5 题:

```
var arr = [];  
arr[0] = 'a';  
arr[1] = 'b';  
arr[10] = 'c';  
alert(arr.length); //答案: 11  
console.log(arr[5]); //答案: undefined
```

### 257、第 6 题:

```
var a=1;  
console.log(a++); //答案: 1  
console.log(++a); //答案: 3
```

### 258、第 7 题:

```
console.log(null==undefined); //答案: true  
console.log("1"==1); //答案: true, 因为会将数字 1 先转换为字符串 1  
console.log("1"===1); //答案: false, 因为数据类型不一致 ===不会进行类型转换
```

### 259、第 8 题:

```
typeof 1; "number"  
typeof "hello"; "string"  
typeof /[0-9]/; "object" 正则表达式是个RegExp类型的对象  
typeof {}; "object"  
typeof null; "object"  
typeof undefined; "undefined"  
typeof [1,2,3]; "object"  
typeof function(){}; //"function"
```

### 260、第 9 题:

```
parseInt(3.14); //3  
parseFloat("3asdf"); //3  
parseInt("1.23abc456"); //1  
parseInt(true); //"true" NaN
```

## 261、第 10 题：

//考点：函数声明提前

```
function bar() {  
    return foo; function foo(){}会提前声明。之后再执行return foo; 返回的是一个函数  
    foo = 10;  
    function foo() {}  
    //var foo = 11;  
}  
alert(typeof bar()); // "function"
```

## 262、第 11 题： 考点： 函数声明提前

```
var foo = 1;  
function bar() {  
    foo = 10;  
    return; function foo 会声明提前，但是这个函数foo是在bar函数内部声明的。因此function foo是个局部变量，指向函数function(){}。后来再执行foo=10, 会覆盖 var foo->function ;  
    function foo() {}  
}  
bar();  
alert(foo); // 答案： 1 这里的foo是个全局变量，是指var foo=1
```

## 263、第 12 题：

```
console.log(a); // 是一个函数  
var a = 3;  
function a() {}  
console.log(a); // 3 函数是第一公民。函数声明提前会在变量声明提前的前面。因为函数声明的优先级高于变量声明的优先级
```

## 264、第 13 题：

//考点：对 arguments 的操作

```
function foo(a) {  
    arguments[0] = 2;  
    alert(a); // 答案： 2， 因为： a、arguments 是对实参的访问，b、通过 arguments[i]  
    可以修改指定实参的值  
}  
foo(1);
```

### 265、第 14 题:

```
function foo(a) {  
    alert(arguments.length);//答案: 3, 因为 arguments 是对实参的访问  
}  
foo(1, 2, 3);
```

### 266、第 15 题

```
bar();//报错  
var foo = function bar(name) {  
    console.log("hello"+name);  
    console.log(bar);  
};  
//alert(typeof bar);  
foo("world");//"hello"  
console.log(bar);//undefined  
console.log(foo.toString());  
bar();//报错
```

### 267、第 16 题

```
function test() {  
    console.log("test 函数");  
}  
setTimeout(function() {  
    console.log("定时器回调函数");  
}, 0)  
test();  
function foo() {  
    var name="hello";  
}
```

### 三、Jquery

1、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？

先 `stop(true, true)` 后 `slideUp()`

示例代码参考：J:\代码, PPT, 笔记, 电子书\面试题\面试题\_jquery\_slideup.html

关于 `stop()` 参考：J:\代码, PPT, 笔记, 电子书\面试题\面试题\_jquery.png

### 四、HTML5 CSS3

1、CSS3 有哪些新特性？

1. CSS3 实现圆角 (`border-radius`)，阴影 (`box-shadow`)，
2. 对文字加特效 (`text-shadow`)，线性渐变 (`gradient`)，旋转 (`transform`)
3. `transform: rotate(9deg) scale(0.85, 0.90) translate(0px, -30px) skew(-9deg, 0deg);` // 旋转, 缩放, 定位, 倾斜
4. 增加了更多的 CSS 选择器 多背景 `rgba`
5. 在 CSS3 中唯一引入的伪元素是 `::selection`.
6. 媒体查询，多栏布局
7. `border-image`

2、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？

新特性：

1. 拖拽释放 (Drag and drop) API
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`)
4. 画布 (Canvas) API
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`
9. 新的技术 `webworker, websocket, Geolocation`

移除的元素：

1. 纯表现的元素：basefont, big, center, font, s, strike, tt, u;
2. 对可用性产生负面影响的元素：frame, frameset, noframes;

支持 HTML5 新标签：

1. 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（当然最好的方式是直接使用成熟的框架、使用最多的是 html5shim 框架）：

```
<!--[if lt IE 9]>
```

```
<script> src="http://html5shim.googlecode.com/svn/trunk/html5.js"</script>
```

```
<![endif]-->
```

如何区分：

DOCTYPE 声明新增的结构元素、功能元素

### 3、本地存储（Local Storage）和 cookies（储存在用户本地终端上的数据）之间的区别是什么？

**Cookies:** 服务器和客户端都可以访问；大小只有 4KB 左右；有有效期，过期后将会删除；

**本地存储：** 只有本地浏览器端可访问数据，服务器不能访问本地存储直到故意通过 POST 或者 GET 的通道发送到服务器；每个域 5MB；没有过期数据，它将保留知道用户从浏览器清除或者使用 Javascript 代码移除

### 4、如何实现浏览器内多个标签页之间的通信？

调用 localStorage、cookies 等本地存储方式

### 5、你如何对网站的文件和资源进行优化？

文件合并

文件最小化/文件压缩

使用 CDN 托管

缓存的使用

### 6、什么是响应式设计？

它是关于网页制作的过程中让不同的设备有不同的尺寸和不同的功能。响应式设计是让所有的人能在这些设备上让网站运行正常

### 7、新的 HTML5 文档类型和字符集是？

答：HTML5 文档类型：<!doctype html>

HTML5 使用的编码<meta charset=" UTF-8" >

## 8、HTML5 Canvas 元素有什么用？

答：Canvas 元素用于在网页上绘制图形，该元素标签强大之处在于可以直接在 HTML 上进行图形操作。

## 9、HTML5 存储类型有什么区别？

答：Media API、Text Track API、Application Cache API、User Interaction、Data Transfer API、Command API、Constraint Validation API、History API

## 10、用 H5+CSS3 解决下导航栏最后一项掉下来的问题

## 11、CSS3 新增伪类有那些？

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。

:enabled、:disabled 控制表单控件的禁用状态。

:checked, 单选框或复选框被选中。

## 12、请用 CSS 实现：一个矩形内容，有投影，有圆角，hover 状态慢慢变透明。

css 属性的熟练程度和实践经验

## 13、描述下 CSS3 里实现元素动画的方法

动画相关属性的熟悉程度

## 14、html5\CSS3 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，地理定位等功能的增加。

\* 绘画 canvas 元素

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

语义化更好的内容元素，比如 article、footer、header、nav、section

表单控件，calendar、date、time、email、url、search

CSS3 实现圆角，阴影，对文字加特效，增加了更多的 CSS 选择器 多背景 rgba

新的技术 webworker, websockt, 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;

\* 是 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风格：

\* 当然最好的方式是直接使用成熟的框架、使用最多的是 html5shim 框架

```
<!--[if lt IE 9]>
```

```
<script> src="http://html5shim.googlecode.com/svn/trunk/html5.js"</script>
```

```
<![endif]-->
```

15、你怎么来实现页面设计图，你认为前端应该如何高质量完成工作？一个满屏 品 字布局 如何设计？

\* 首先划分成头部、body、脚部；。。。。。

\* 实现效果图是最基本的工作，精确到 2px;

与设计师，产品经理的沟通和项目的参与

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 HTML5。

16、你能描述一下渐进增强和优雅降级之间的不同吗？

渐进增强 progressive enhancement：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 graceful degradation：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。

区别：优雅降级是从复杂的现状开始，并试图减少用户体验的供给，而渐进增强则是从一个非常基础的，能够起作用的版本开始，并不断扩充，以适应未来环境的需要。降级（功能衰减）意味着往回看；而渐进增强则意味着朝前看，同时保证其根基处于安全地带。

“优雅降级”观点

“优雅降级”观点认为应该针对那些最高级、最完善的浏览器来设计网站。而将那些被认为“过时”或有功能缺失的浏览器下的测试工作安排在开发周期的最后阶段，并把测试对象限定为主流浏览器（如 IE、Mozilla 等）的前一个版本。

在这种设计范例下，旧版的浏览器被认为仅能提供“简陋却无妨（poor, but passable）”的浏览体验。你可以做一些小的调整来适应某个特定的浏览器。但由于它们并非我们所关注的焦点，因此除了修复较大的错误之外，其它的差异将被直接忽略。

“渐进增强”观点

“渐进增强”观点则认为应关注于内容本身。

内容是我们建立网站的诱因。有的网站展示它，有的则收集它，有的寻求，有的操作，还有的网站甚至会包含以上的种种，但相同点是它们全都涉及到内容。这使得“渐进增强”成为一种更为合理的设计范例。这也是它立即被 Yahoo! 所采纳并用以构建其“分级式浏览器支持（Graded Browser Support）”策略的原因所在。

那么问题了。现在产品经理看到 IE6, 7, 8 网页效果相对高版本现代浏览器少了很多圆角，阴影（CSS3），要求兼容（使用图片背景，放弃 CSS3），你会如何说服他？

17、为什么利用多个域名来存储网站资源会更有效？

CDN 缓存更方便

突破浏览器并发限制

节约 cookie 带宽

节约主域名的连接数，优化页面响应速度

防止不必要的安全问题

18、请谈一下你对网页标准和标准制定机构重要性的理解。

（无标准答案）网页标准和标准制定机构都是为了让 web 发展的更‘健康’，开发者遵循统一的标准，降低开发难度，开发成本，SEO 也会更好做，也不会因为滥用代码导致各种 BUG、安全问题，最终提高网站易用性。

19、请描述一下 cookies，sessionStorage 和 localStorage 的区别？

sessionStorage 用于本地存储一个会话（session）中的数据，这些数据只有在同一个会话中的页面才能访问并且当会话结束后数据也随之销毁。因此 sessionStorage 不是一种

持久化的本地存储，仅仅是会话级别的存储。而 `localStorage` 用于持久化的本地存储，除非主动删除数据，否则数据是永远不会过期的。

web storage 和 cookie 的区别

Web Storage 的概念和 cookie 相似，区别是它是为了更大容量存储设计的。Cookie 的大小是受限的，并且每次你请求一个新的页面的时候 Cookie 都会被发送过去，这样无形中浪费了带宽，另外 cookie 还需要指定作用域，不可以跨域调用。

除此之外，Web Storage 拥有 `setItem`, `getItem`, `removeItem`, `clear` 等方法，不像 cookie 需要前端开发者自己封装 `setCookie`, `getCookie`。但是 Cookie 也是不可以或缺的：Cookie 的作用是与服务器进行交互，作为 HTTP 规范的一部分而存在，而 Web Storage 仅仅是为了在本地“存储”数据而生。

## 20、知道 css 有个 content 属性吗？有什么作用？有什么应用？

知道。css 的 content 属性专门应用在 before/after 伪元素上，用来插入生成内容。最常见的应用是利用伪类清除浮动。

//一种常见利用伪类清除浮动的代码

```
.clearfix:after {
    content: "."; //这里利用到了 content 属性
    display: block;
    height: 0;
    visibility: hidden;
    clear: both; }

.clearfix {
    *zoom: 1;
}
```

after 伪元素通过 content 在元素的后面生成了内容为一个点的块级元素，再利用 `clear: both` 清除浮动。

那么问题继续还有，知道 css 计数器（序列数字字符自动递增）吗？如何通过 css content 属性实现 css 计数器？

答案：css 计数器是通过设置 `counter-reset`、`counter-increment` 两个属性、及 `counter()`/`counters()` 一个方法配合 after / before 伪类实现。

## 21、如何在 HTML5 页面中嵌入音频？

HTML 5 包含嵌入音频文件的标准方式，支持的格式包括 MP3、Wav 和 Ogg：

<audio controls>

<source src="jamshed.mp3" type="audio/mpeg">

```
    Your browser doesn't support audio embedding feature.
</audio>
```

## 22、如何在 HTML5 页面中嵌入视频？

和音频一样，HTML5 定义了嵌入视频的标准方法，支持的格式包括：MP4、WebM 和 Ogg：

```
<video width="450" height="340" controls>
  <source src="jamshed.mp4" type="video/mp4">
  Your browser doesn't support video embedding feature.
</video>
```

## 23、HTML5 引入什么新的表单属性？

Datalist    datetime    output    keygen    date    month    week    time    number    range  
emailurl

## 24、CSS3 新增伪类有那些？

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

## 25、(写)描述一段语义的 html 代码吧。

(HTML5 中新增加的很多标签 (如: <article>、<nav>、<header>和<footer>等)  
就是基于语义化设计原则)

```
< div id="header">
< h1>标题< /h1>
< h2>专注 Web 前端技术< /h2>
< /div>
```

语义 HTML 具有以下特性：

文字包裹在元素中，用以反映内容。例如：

段落包含在 <p> 元素中。

顺序表包含在<ol>元素中。

从其他来源引用的大型文字块包含在<blockquote>元素中。

HTML 元素不能用作语义用途以外的其他目的。例如：

<h1>包含标题，但并非用于放大文本。

<blockquote>包含大段引述，但并非用于文本缩进。

空白段落元素（ <p></p> ）并非用于跳行。

文本并不直接包含任何样式信息。例如：

不使用 <font> 或 <center> 等格式标记。

类或 ID 中不引用颜色或位置。

## 26. cookie 在浏览器和服务器间来回传递。 sessionStorage 和 localStorage 区别

sessionStorage 和 localStorage 的存储空间更大；

sessionStorage 和 localStorage 有更多丰富易用的接口；

sessionStorage 和 localStorage 各自独立的存储空间；

## 27. 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

语义化更好的内容元素，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风格：

\* 当然最好的方式是直接使用成熟的框架、使用最多的是 html5shim 框架

```
<!--[if lt IE 9]>
```

```
<script> src="http://html5shim.googlecode.com/svn/trunk/html5.js"</script>
```

```
<![endif]-->
```

## 28、如何区分： DOCTYPE 声明\新增的结构元素\功能元素

## 29、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 CCS 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。

## 30、HTML5 的离线储存？

localStorage 长期存储数据，浏览器关闭后数据不丢失；

sessionStorage 数据在浏览器关闭后自动删除。

## 31、写出 HTML5 的文档声明方式

```
<DOCTYPE html>
```

## 32、HTML5 和 CSS3 的新标签

HTML5: nav, footer, header, section, hgroup, video, time, canvas, audio...

CSS3: RGBA, opacity, text-shadow, box-shadow, border-radius, border-image, border-color, transform...;

## 33、自己对标签语义化的理解

在我看来，语义化就是比如说一个段落，那么我们就应该用 <p>标签来修饰，标题就应该用 <h>标签等。符合文档语义的标签。

# 五、移动 web 开发

## 1、移动端常用类库及优缺点

知识面宽度，多多益善

## 2、Zepto 库和 JQ 区别

Zepto 相对 jQuery 更加轻量，主要用在移动端，jQuery 也有对应的 jQuerymobile 移动端框架 d

## 六、Ajax

### 1、Ajax 是什么？如何创建一个 Ajax？

Ajax 并不算是一种新的技术，全称是 asynchronous javascript and xml，可以说是已有技术的组合，主要用来实现客户端与服务器端的异步通信效果，实现页面的局部刷新，早期的浏览器并不能原生支持 ajax，可以使用隐藏帧（iframe）方式变相实现异步效果，后来的浏览器提供了对 ajax 的原生支持

使用 ajax 原生方式发送请求主要通过 XMLHttpRequest（标准浏览器）、ActiveXObject（IE 浏览器）对象实现异步通信效果

基本步骤：

```
var xhr = null; // 创建对象
if (window.XMLHttpRequest) {
    xhr = new XMLHttpRequest();
} else {
    xhr = new ActiveXObject("Microsoft.XMLHTTP");
}
xhr.open("方式", "地址", "标志位"); // 初始化请求
xhr.setRequestHeader(" ", " "); // 设置 http 头信息
xhr.onreadystatechange = function() {} // 指定回调函数
xhr.send(); // 发送请求
```

js 框架（jQuery/EXTJS 等）提供的 ajax API 对原生的 ajax 进行了封装，熟悉了基础理论，再学习别的框架就会得心应手，好多都是换汤不换药的内容

### 2、同步和异步的区别？

**同步：阻塞的**

- 张三叫李四去吃饭，李四一直忙个不停，张三一直等着，直到李四忙完两个人一块去吃饭  
= 浏览器向服务器请求数据，服务器比较忙，浏览器一直等着（页面白屏），直到服务器返回数据，浏览器才能显示页面

**异步：非阻塞的**

- 张三叫李四去吃饭，李四在忙，张三说了一声然后自己就去吃饭了，李四忙完后自己去吃  
= 浏览器向服务器请求数据，服务器比较忙，浏览器可以自如的干原来的事情（显示页面），服务器返回数据的时候通知浏览器一声，浏览器把返回的数据再渲染到页面，局部更新

### 3、如何解决跨域问题？

理解跨域的概念：协议、域名、端口都相同才同域，否则都是跨域

出于安全考虑，服务器不允许 ajax 跨域获取数据，但是可以跨域获取文件内容，所以基于这一点，可以动态创建 script 标签，使用标签的 src 属性访问 js 文件的形式获取 js

脚本，并且这个 js 脚本中的内容是函数调用，该函数调用的参数是服务器返回的数据，为了获取这里的参数数据，需要事先在页面中定义回调函数，在回调函数中处理服务器返回的数据，这就是解决跨域问题的主流解决方案

#### 4、页面编码和被请求的资源编码如果不一致如何处理？

对于 ajax 请求传递的参数，如果是 get 请求方式，参数如果传递中文，在有些浏览器会乱码，不同的浏览器对参数编码的处理方式不同，所以对于 get 请求的参数需要使用 encodeURIComponent 函数对参数进行编码处理，后台开发语言都有相应的解码 api。对于 post 请求不需要进行编码

#### 5、简述 ajax 的过程。

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

#### 6、阐述一下异步加载。

1. 异步加载的方案：动态插入 script 标签
2. 通过 ajax 去获取 js 代码，然后通过 eval 执行
3. script 标签上添加 defer 或者 async 属性
4. 创建并插入 iframe，让它异步执行 js

#### 7、请解释一下 JavaScript 的同源策略。

同源策略是客户端脚本（尤其是 Javascript）的重要的安全度量标准。它最早出自 Netscape Navigator 2.0，其目的是防止某个文档或脚本从多个不同源装载。所谓同源指的是：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来源的窗口和文档的属性。

#### 8、GET 和 POST 的区别，何时使用 POST？

GET：一般用于信息获取，使用 URL 传递参数，对所发送信息的数量也有限制，一般在 2000 个字符，有的浏览器是 8000 个字符

POST：一般用于修改服务器上的资源，对所发送的信息没有限制

在以下情况中，请使用 POST 请求：

1. 无法使用缓存文件（更新服务器上的文件或数据库）



2. 向服务器发送大量数据 (POST 没有数据量限制)
3. 发送包含未知字符的用户输入时, POST 比 GET 更稳定也更可靠

## 9、ajax 是什么?ajax 的交互模型?同步和异步的区别?如何解决跨域问题?

1. 通过异步模式, 提升了用户体验
2. 优化了浏览器和服务器之间的传输, 减少不必要的数据往返, 减少了带宽占用
3. Ajax 在客户端运行, 承担了一部分本来由服务器承担的工作, 减少了大用户量下的服务器负载。

## 10、Ajax 的最大的特点是什么。

Ajax 可以实现异步通信效果, 实现页面局部刷新, 带来更好的用户体验; 按需获取数据, 节约带宽资源;

## 11、ajax 的缺点

- 1、ajax 不支持浏览器 back 按钮。
- 2、安全问题 AJAX 暴露了与服务器交互的细节。
- 3、对搜索引擎的支持比较弱。
- 4、破坏了程序的异常机制。

## 12、ajax 请求的时候 get 和 post 方式的区别

get 一般用来进行查询操作, url 地址有长度限制, 请求的参数都暴露在 url 地址当中, 如果传递中文参数, 需要自己进行编码操作, 安全性较低。

post 请求方式主要用来提交数据, 没有数据长度的限制, 提交的数据内容存在于 http 请求体中, 数据不会暴露 url 地址中。

## 13、解释 jsonp 的原理, 以及为什么不是真正的 ajax

Jsonp 并不是一种数据格式, 而 json 是一种数据格式, jsonp 是用来解决跨域获取数据的一种解决方案, 具体是通过动态创建 script 标签, 然后通过标签的 src 属性获取 js 文件中的 js 脚本, 该脚本的内容是一个函数调用, 参数就是服务器返回的数据, 为了处理这些返回的数据, 需要事先在页面定义好回调函数, 本质上使用的并不是 ajax 技术

## 14、什么是 Ajax 和 JSON, 它们的优缺点。

Ajax 是全称是 asynchronous JavaScript andXML, 即异步 JavaScript 和 xml, 用于在 Web 页面中实现异步数据交互, 实现页面局部刷新。

优点: 可以使得页面不重载全部内容的情况下加载局部内容, 降低数据传输量, 避免用户不断刷新或者跳转页面, 提高用户体验

缺点：对搜索引擎不友好；要实现 ajax 下的前后退功能成本较大；可能造成请求数的增加跨域问题限制；

JSON 是一种轻量级的数据交换格式，ECMA 的一个子集

优点：轻量级、易于人的阅读和编写，便于机器（JavaScript）解析，支持复合数据类型（数组、对象、字符串、数字）

## 15、http 常见的状态码有那些？分别代表是什么意思？

200 - 请求成功

301 - 资源（网页等）被永久转移到其它 URL

404 - 请求的资源（网页等）不存在

500 - 内部服务器错误

## 16、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？

分为 4 个步骤：

1. 当发送一个 URL 请求时，不管这个 URL 是 Web 页面的 URL 还是 Web 页面上每个资源的 URL，浏览器都会开启一个线程来处理这个请求，同时在远程 DNS 服务器上启动一个 DNS 查询。这能使浏览器获得请求对应的 IP 地址。
2. 浏览器与远程 Web 服务器通过 TCP 三次握手协商来建立一个 TCP/IP 连接。该握手包括一个同步报文，一个同步-应答报文和一个应答报文，这三个报文在浏览器和服务器之间传递。该握手首先由客户端尝试建立起通信，而后服务器应答并接受客户端的请求，最后由客户端发出该请求已经被接受的报文。
3. 一旦 TCP/IP 连接建立，浏览器会通过该连接向远程服务器发送 HTTP 的 GET 请求。远程服务器找到资源并使用 HTTP 响应返回该资源，值为 200 的 HTTP 响应状态表示一个正确的响应。
4. 此时，Web 服务器提供资源服务，客户端开始下载资源。

## 17、ajax 请求的时候 get 和 post 方式的区别

get 一般用来进行查询操作，url 地址有长度限制，请求的参数都暴露在 url 地址当中，如果传递中文参数，需要自己进行编码操作，安全性较低。

post 请求方式主要用来提交数据，没有数据长度的限制，提交的数据内容存在于 http 请求体中，数据不会暴露 url 地址中。

## 18、ajax 请求时，如何解释 json 数据

使用 eval() 或者 JSON.parse() 鉴于安全性考虑，推荐使用 JSON.parse() 更靠谱，对数据的安全性更好。

## 19、. javascript 的本地对象，内置对象和宿主对象

本地对象为独立于宿主环境的 ECMAScript 提供的对象，包括 Array Object RegExp 等可以 new 实例化的对象

内置对象为 ~~Global~~，Math 等不可以实例化的（他们也是本地对象，内置对象是本地对象的一个子集） <sup>Global 对象</sup> P130页

宿主对象为所有的非本地对象，所有的 BOM 和 DOM 对象都是宿主对象，如浏览器自带的 document, window 等对象

## 20、为什么利用多个域名来存储网站资源会更有效？

确保用户在不同地区能用最快的速度打开网站，其中某个域名崩溃用户也能通过其他郁闷访问网站，并且不同的资源放到不同的服务器上有利于减轻单台服务器的压力。

## 21、请说出三种减低页面加载时间的方法

- 1、压缩 css、js 文件
- 2、合并 js、css 文件，减少 http 请求
- 3、外部 js、css 文件放在最底下
- 4、减少 dom 操作，尽可能用变量替代不必要的 dom 操作

## 22、HTTP 状态码都有那些。

200 OK //客户端请求成功  
400 Bad Request //客户端请求有语法错误，不能被服务器所理解  
403 Forbidden //服务器收到请求，但是拒绝提供服务  
404 Not Found //请求资源不存在，输入了错误的 URL  
500 Internal Server Error //服务器发生不可预期的错误  
503 Server Unavailable //服务器当前不能处理客户端的请求，一段时间后可能恢复正常

## 七、JS 高级

### 1、jQuery 一个对象可以同时绑定多个事件，这是如何实现的？

jQuery 可以给一个对象同时绑定多个事件，低层实现方式是使用 addEventListener 或 attachEvent 兼容不同的浏览器实现事件的绑定，这样可以给同一个对象注册多个事件。

### 2、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？

Webkit 是浏览器引擎，包括 html 渲染和 js 解析功能，手机浏览器的主流内核，与之相对应的引擎有 Gecko（Mozilla Firefox 等使用）和 Trident（也称 MSHTML，IE 使用）。

对于浏览器的调试工具要熟练使用，主要是页面结构分析，后台请求信息查看，js 调试工具使用，熟练使用这些工具可以快速提高解决问题的效率

### 3、如何测试前端代码？知道 BDD, TDD, Unit Test 么？知道怎么测试你的前端工程么(mocha, sinon, jasmine, qUnit..)?

了解 BDD 行为驱动开发与 TDD 测试驱动开发已经单元测试相关概念，

### 4、前端 templating(Mustache, underscore, handlebars)是干嘛的，怎么用？

Web 模板引擎是为了使用户界面与业务数据（内容）分离而产生的，

Mustache 是一个 logic-less （轻逻辑）模板解析引擎，它的优势在于可以应用在 Javascript、PHP、Python、Perl 等多种编程语言中。

Underscore 封装了常用的 JavaScript 对象操作方法，用于提高开发效率。

Handlebars 是 JavaScript 一个语义模板库，通过对 view 和 data 的分离来快速构建 Web 模板。

## 5、简述一下 Handlebars 的基本用法？

没有用过的话说出它是干什么的即可

官网：<http://handlebarsjs.com/>

参考：J:\代码, PPT, 笔记, 电子书\面试题\handlebarDemo

### 6、简述一下 Handlebars 的对模板的基本处理流程， 如何编译的？如何缓存的？

学习技术不仅要会用，还有熟悉它的实现机制，这样在开发中遇到问题时才能更好的解决

### 7、用 js 实现千位分隔符？

原生 js 的熟练度，实践经验，实现思路

### 8、检测浏览器版本有哪些方式？

IE 与标准浏览器判断，IE 不同版本的判断，userAgent `var ie = /*cc_on !*/false;`

9、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，你来说下会执行几次事件，然后会先执行冒泡还是捕获

对两种事件模型的理解

首先发生的是事件捕获，为截获事件提供了机会。然后是实际的目标接收到事件。最后一个阶段是冒泡阶段，可以在这个阶段对事件做出响应。P347

10、实现一个函数 clone，可以对 JavaScript 中的 5 种主要的数据类型（包括 Number、String、Object、Array、Boolean）进行值复制

- 考察点 1：对于基本数据类型和引用数据类型在内存中存放的是值还是指针这一区别是否清楚
- 考察点 2：是否知道如何判断一个变量是什么类型的
- 考察点 3：递归算法的设计

// 方法一：

```
Object.prototype.clone = function() {  
    var o = this.constructor === Array ? [] : {};  
    for(var e in this){  
        o[e] = typeof this[e] === "object" ? this[e].clone() : this[e];  
    }  
    return o;  
}
```

//方法二：

```
/**  
 * 克隆一个对象  
 * @param Obj  
 * @returns  
 */  
function clone(Obj) {  
    var buf;  
    if (Obj instanceof Array) {  
        buf = []; // 创建一个空的数组  
        var i = Obj.length;  
        while (i--) {  
            buf[i] = clone(Obj[i]);  
        }  
        return buf;  
    } else if (Obj instanceof Object) {  
        buf = {}; // 创建一个空对象  
        for (var k in Obj) { // 为这个对象添加新的属性  
            buf[k] = clone(Obj[k]);  
        }  
    }  
}
```

```

        return buf;
    }else{ //普通变量直接赋值
        return Obj;
    }
}

```

## 11、如何消除一个数组里面重复的元素？

```

var arr=[1,2,3,3,4,4,5,5,6,1,9,3,25,4];

function deRepeat() {
    var newArr=[];
    var obj={};
    var index=0;
    var l=arr.length;
    for(var i=0;i<l;i++){
        if(obj[arr[i]]==undefined)
        {
            obj[arr[i]]=1;
            newArr[index++]=arr[i];
        }
        else if(obj[arr[i]]==1)
            continue;
    }
    return newArr;
}

var newArr2=deRepeat(arr);
alert(newArr2); //输出 1,2,3,4,5,6,9,25

```

12、小贤是一条可爱的小狗(Dog)，它的叫声很好听(wow)，每次看到主人的时候就会乖乖叫一声(yelp)。从这段描述可以得到以下对象：

```

function Dog() {
    this.wow = function() {
        alert(' Wow' );
    }
    this.yelp = function() {
        this.wow();
    }
}

```

小芒和小贤一样，原来也是一条可爱的小狗，可是突然有一天疯了(MadDog)，一看到人就会每隔半秒叫一声(wow)地不停叫唤(yelp)。请根据描述，按示例的形式用代码来实。（继承，原型，setInterval）

```
function MadDog() {
    this.yelp = function() {
        var self = this;
        setInterval(function() {
            self.wow();
        }, 500);
    }
}

MadDog.prototype = new Dog();
//for test
var dog = new Dog();
dog.yelp();
var madDog = new MadDog();
madDog.yelp();
```

### 13、下面这个 ul，如何点击每一列的时候 alert 其 index?（闭包）

```
<ul id=" test" >
<li>这是第一条</li>
<li>这是第二条</li>
<li>这是第三条</li>
</ul>
// 方法一：
var lis=document.getElementById('2223').getElementsByTagName('li');
for(var i=0;i<3;i++)
{
    lis[i].index=i;
    lis[i].onclick=function() {
        alert(this.index);
    };
}
//方法二：
var lis=document.getElementById('2223').getElementsByTagName('li');
for(var i=0;i<3;i++){
```

```

lis[i].index=i;
lis[i].onclick=(function(a) {
    return function() {
        alert(a);
    }
})(i);
}

```

14、编写一个 JavaScript 函数，输入指定类型的选择器(仅需支持 id, class, tagName 三种简单 CSS 选择器，无需兼容组合选择器)可以返回匹配的 DOM 节点，需考虑浏览器兼容性和性能。

```

/** @param selector {String} 传入的 CSS 选择器。* @return {Array}*/

var query = function(selector) {
    var reg = /^(#)?(\.)?(\w+)$/img;
    var regResult = reg.exec(selector);
    var result = [];
    //如果是 id 选择器
    if(regResult[1]) {
        if(regResult[3]) {
            if(typeof document.querySelector === "function") {
                result.push(document.querySelector(regResult[3]));
            } else {
                result.push(document.getElementById(regResult[3]));
            }
        }
    }
    //如果是 class 选择器
    else if(regResult[2]) {
        if(regResult[3]) {
            if(typeof document.getElementsByClassName === 'function') {
                var doms = document.getElementsByClassName(regResult[3]);
                if(doms) {
                    result = converToArray(doms);
                }
            }
        }
    }
}

```



```

        //如果不支持 getElementsByClassName 函数
        else {
            var allDoms = document.getElementsByTagName("*") ;
            for(var i = 0, len = allDoms.length; i < len; i++) {
                if(allDoms[i].className.search(new RegExp(regResult[2])) > -1) {
                    result.push(allDoms[i]);
                }
            }
        }
    }
}

//如果是标签选择器
else if(regResult[3]) {
    var doms = document.getElementsByTagName(regResult[3].toLowerCase());
    if(doms) {
        result = converToArray(doms);
    }
}

return result;
}

function converToArray(nodes) {
    var array = null;
    try{
        array = Array.prototype.slice.call(nodes, 0); //针对非 IE 浏览器
    } catch(ex) {
        array = new Array();
        for( var i = 0 , len = nodes.length; i < len ; i++ ) {
            array.push(nodes[i])
        }
    }
    return array;
}
}

```

15、请评价以下代码并给出改进意见。 **if里面不能声明函数**

```

if(window.addEventListener){
    var addListener = function(el, type, listener, useCapture) {
        el.addEventListener(type, listener, useCapture);
    }
}

```

```

    };
}
else if(document.all){
    addListener = function(el,type,listener){
        el.attachEvent("on"+type,function(){
            listener.apply(el);
        });
    }
}

```

- 不应该在 if 和 else 语句中声明 addListener 函数，应该先声明；
- 不需要使用 window.addEventListener 或 document.all 来进行检测浏览器，应该使用能力检测；
- 由于 attachEvent 在 IE 中有 this 指向问题，所以调用它时需要处理一下

改进如下：

```

function addEvent(elem, type, handler){
    if(elem.addEventListener){
        elem.addEventListener(type, handler, false);
    }else if(elem.attachEvent){
        elem['temp' + type + handler] = handler;
        elem[type + handler] = function(){
            elem['temp' + type + handler].apply(elem);
        };
        elem.attachEvent('on' + type, elem[type + handler]);
    }else{
        elem['on' + type] = handler;
    }
}

```

**16、给 String 对象添加一个方法，传入一个 string 类型的参数，然后将 string 的每个字符间价格空格返回，例如：**

```

addSpace( "hello world" ) // -> 'h e l l o   w o r l d'
String.prototype.spacify = function(){
    return this.split('').join(' ');
};

```

接着上述问题答案提问，1) 直接在对象的原型上添加方法是否安全？尤其是在 Object 对象上。（这个我没能答出？希望知道的说一下。） 2) 函数声明与函数表达式的区别？

答案：在 js 中，解析器在向执行环境中加载数据时，对函数声明和函数表达式并非是一视同仁的，解析器会率先读取函数声明，并使其在执行任何代码之前可用（可以访问），至于函数表达式，则必须等到解析器执行到它所在的代码行，才会真正被解析执行。

## 17、定义一个 log 方法，让它可以代理 console.log 的方法。

可行的方法一：

```
function log(msg) {  
    console.log(msg);  
}  
  
log("hello world!") // hello world!
```

如果要传入多个参数呢？显然上面的方法不能满足要求，所以更好的方法是：

```
function log() {  
    console.log.apply(console, arguments);  
};
```

到此，追问 apply 和 call 方法的异同。

对于 apply 和 call 两者在作用上是相同的，即是调用一个对象的一个方法，以另一个对象替换当前对象。将一个函数的对象上下文从初始的上下文改变为由 thisObj 指定的新对象。

但两者在参数上有区别的。对于第一个参数意义都一样，但对第二个参数： apply 传入的是一个参数数组，也就是将多个参数组合成为一个数组传入，而 call 则作为 call 的参数传入（从第二个参数开始）。如 func.call(func1, var1, var2, var3) 对应的 apply 写法为：func.apply(func1, [var1, var2, var3])。

## 18、在 Javascript 中什么是伪数组？如何将伪数组转化为标准数组？

伪数组也有length属性

伪数组（类数组）：无法直接调用数组方法或期望 length 属性有什么特殊的行为，但仍可以对真正数组遍历方法来遍历它们。典型的是函数的 argument 参数，还有像调用 getElementByTagName, document.childNodes 之类的，它们都返回 NodeList 对象都属于伪数组。可以使用 Array.prototype.slice.call(fakeArray) 将数组转化为真正的 Array 对象。

假设接第八题题干，我们要给每个 log 方法添加一个” (app)” 前缀，比如’ hello world!’ ->’ (app)hello world!’。方法如下：

```
function log() {
    var args = Array.prototype.slice.call(arguments);    //为了使用 unshift
    数组方法，将 argument 转化为真正的数组
    args.unshift(' (app)');
    console.log.apply(console, args);
};
```

## 19、对作用域上下文和 this 的理解，看下列代码：

```
var User = {
    count: 1,
    getCount: function() {
        return this.count;
    }
};
console.log(User.getCount());    // what? this=>user
var func = User.getCount;
console.log(func());    // what? this=>window
问两处 console 输出什么？为什么？
```

答案是 1 和 undefined。

**func 是在 window 的上下文中被执行的，所以会访问不到 count 属性。**

继续追问，那么如何确保 User 总是能访问到 func 的上下文，即正确返回 1。正确的方法是使用 Function.prototype.bind。兼容各个浏览器完整代码如下：

```
Function.prototype.bind = Function.prototype.bind || function(context) {
    var self = this;
    return function() {
        return self.apply(context, arguments);
    };
}
var func = User.getCount.bind(User);
console.log(func());
```

## 20、原生 JS 的 window.onload 与 JQuery 的 \$(document).ready(function(){} )有什么不同？如何用原生 JS 实现 Jq 的 ready 方法？

**window.onload() 方法**是必须等到页面内包括图片的所有元素加载完毕后才能执行。  
包括CSS

`$(document).ready()` 是 DOM 结构绘制完毕后就执行，不必等到加载完毕。

```
/*
 * 传递函数给 whenReady()
 * 当文档解析完毕且为操作准备就绪时，函数作为 document 的方法调用
 */
var whenReady = (function() { //这个函数返回 whenReady()
    函数
    var funcs = []; //当获得事件时，要运行的函数
    var ready = false; //当触发事件处理程序时，切换为 true
    //当文档就绪时，调用事件处理程序
    function handler(e) {
        if(ready) return; //确保事件处理程序只完整运行一次
        //如果发生 onreadystatechange 事件，但其状态不是 complete 的话，那么文
        档尚未准备好
        if(e.type === 'readystatechange' && document.readyState !==
        'complete') {
            return;
        }
        //运行所有注册函数
        //注意每次都要计算 funcs.length
        //以防这些函数的调用可能会导致注册更多的函数
        for(var i=0; i<funcs.length; i++) {
            funcs[i].call(document);
        }
        //事件处理函数完整执行，切换 ready 状态，并移除所有函数
        ready = true;
        funcs = null;
    }
    //为接收到的任何事件注册处理程序
    if(document.addEventListener) {
        document.addEventListener('DOMContentLoaded', handler, false);
        document.addEventListener('readystatechange', handler,
        false); //IE9+
        window.addEventListener('load', handler, false);
    }else if(document.attachEvent) {
        document.attachEvent('onreadystatechange', handler);
        window.attachEvent('onload', handler);
    }
}
```

```

    }
    //返回 whenReady() 函数
    return function whenReady(fn) {
        if(ready) { fn.call(document); }
        else { funcs.push(fn); }
    }
})();

```

如果上述代码十分难懂，下面这个简化版：

```

function ready(fn) {
    if(document.addEventListener) { //标准浏览器
        document.addEventListener('DOMContentLoaded', function() {
            //注销事件，避免反复触发
            document.removeEventListener('DOMContentLoaded', arguments.callee, false);
            fn(); //执行函数
        }, false);
    } else if(document.attachEvent) { //IE
        document.attachEvent('onreadystatechange', function() {
            if(document.readyState == 'complete') {
                document.detachEvent('onreadystatechange', arguments.callee);
                fn(); //函数执行
            }
        });
    }
};

```

## 21、（设计题）想实现一个对页面某个节点的拖曳？如何做？（使用原生 JS）

回答出概念即可，下面是几个要点

1. 给需要拖拽的节点绑定 mousedown, mousemove, mouseup 事件
2. mousedown 事件触发后，开始拖拽
3. mousemove 时，需要通过 event.clientX 和 event.clientY 获取拖拽位置，并实时更新位置
4. mouseup 时，拖拽结束
5. 需要注意浏览器边界的情况

## 22、请实现如下功能

请实现下面功能（只实现tips组件部分）

1. 用户第一次进来时显示，同一天访问该页面不显示tip提示
2. 用户点击“我知道了”此后访问该页面不再显示tip提醒。



```
function setcookie(name, value, days) {    //给 cookie 增加一个时间变量
    var exp = new Date();
    exp.setTime(exp.getTime() + days*24*60*60*1000); //设置过期时间为 days 天
    document.cookie = name + "=" + escape (value) + ";expires=" + exp.toGMTString();
}

function getCookie(name) {
    var result = "";
    var myCookie = ""+document.cookie+";";
    var searchName = "+name=";
    var startOfCookie = myCookie.indexOf(searchName);
    var endOfCookie;
    if(startOfCookie != -1) {
        startOfCookie += searchName.length;
        endOfCookie = myCookie.indexOf(";", startOfCookie);
        result = (myCookie.substring(startOfCookie, endOfCookie));
    }
    return result;
}

(function() {
    var oTips = document.getElementById('tips');//假设 tips 的 id 为 tips
    var page = {
        check: function() { //检查 tips 的 cookie 是否存在并且允许显示
            var tips = getCookie('tips');
            if(!tips || tips == 'show') return true; //tips 的 cookie 不存在
            if(tips == "never_show_again") return false;
        },
        hideTip: function(bNever) {
            if(bNever) setcookie('tips', 'never_show_again', 365);
            oTips.style.display = "none"; //隐藏
        }
    }
})
```

```

    },
    showTip: function() {
    oTips.style.display = "inline";//显示，假设 tips 为行级元素
    },
    init: function() {
        var _this = this;
        if(this.check()){
            _this.showTip();
            setcookie('tips', 'show', 1);
        }
        oTips.onclick = function() {
            _this.hideTip(true);
        };
    };
    };
    page.init();
}));

```

23、说出以下函数的作用是？空白区域应该填写什么？

```

//define
(function(window) {
    function fn(str) {
        this.str=str;
    }

    fn.prototype.format = function() {
        var arg = _____;
        return this.str.replace(_____, function(a, b) {
            return arg[b]||"";
        });
    }

    window.fn = fn;
})(window);

//use
(function() {
    var t = new fn('<p><a href="{0}">{1}</a><span>{2}</span></p>');

```



```
console.log(t.format('http://www.alibaba.com','Alibaba','Welcome'));
})());
```

答案：访函数的作用是使用 format 函数将函数的参数替换掉 {0} 这样的内容，返回一个格式化后的结果：

第一个空是：arguments

第二个空是：/\{(\d+)\}/ig

## 24、Javascript 作用域链？

作用域链中的下一个变量对象来自包含（外部）环境，而再下一个变量对象则来自下一个包含环境。这样，一直延续到全局执行环境；全局执行环境的变量对象始终是作用域链中的最后一个对象。标识符解析是沿着作用域链一级一级地搜索标识符的过程。搜索过程始终从作用域链的前端开始，然后逐级地向后回溯，直至找到标识符为止（如果找不到标识符，通常会导致错误发生）。

理解变量和函数的访问范围和生命周期，全局作用域与局部作用域的区别，JavaScript 中没有块级作用域，函数的嵌套形成不同层次的作用域，嵌套的层次形成链式形式，通过作用域链查找属性的规则需要深入理解。

## 25、谈谈 this 对象的理解。

理解不同形式的函数调用方式下的 this 指向，理解事件函数、定时函数中的 this 指向，函数的调用形式决定了 this 的指向。

## 26、eval 是做什么的？

调用 eval() 方法时，它会将传入的参数当作实际的 ECMAScript 语句来解析，然后把执行结果插入到原位置

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 个步骤，一次解析成 js 语句，一次执行）

P133

## 27、关于事件，IE 与火狐的事件机制有什么区别？ 如何阻止冒泡？

[1]. 在 IE 中，事件对象是作为一个全局变量来保存和维护的。所有的浏览器事件，不管是用户触发的，还是其他事件，都会更新 window.event 对象。所以在代码中，只要调用 window.event 就可以获取事件对象，再 event.srcElement 就可以取得触发事件的元素进行进一步处理。

[2]. 在 FireFox 中，事件对象却不是全局对象，一般情况下，是现场发生，现场使用，FireFox 把事件对象自动传给事件处理程序。

关于事件的兼容性处理要熟练掌握，事件对象具体哪些属性存在兼容性问题，IE 与标准事件模型事件冒泡与事件捕获的支持要理解

## 28、什么是闭包（closure），为什么要用它？

简单的理解是函数的嵌套形成闭包，闭包包括函数本身已经它的外部作用域使用闭包可以形成独立的空间，延长变量的生命周期，报存中间状态值

29、javascript 代码中的“use strict”;是什么意思？使用它区别是什么？

意思是使用严格模式，使用严格模式，一些不规范的语法将不再支持

## 严格模式

链接: [http://www.ruanyifeng.com/blog/2013/01/javascript\\_strict\\_mode.html](http://www.ruanyifeng.com/blog/2013/01/javascript_strict_mode.html)

全局变量显式声明

静态绑定

禁止使用 with 语句

eval 中定义的变量都是局部变量

禁止 this 关键字指向全局对象

禁止在函数内部遍历调用栈

严格模式下无法删除变量。只有 configurable 设置为 true 的对象属性，才能被删除

正常模式下，对一个对象的只读属性进行赋值，不会报错，只会默默地失败。严格模式下，将报错。

在严格模式下，尝试写入只指定了 getter 函数的属性会抛出错误。类似地，只指定 setter 函数的属性也不能读，  
严格模式下，对一个使用 getter 方法读取的属性进行赋值，会报错。

严格模式下，对禁止扩展的对象添加新属性，会报错。

严格模式下，删除一个不可删除的属性，会报错。

正常模式下，如果对象有多个重名属性，最后赋值的那个属性会覆盖前面的值。严格模式下，这属于语法错误。

正常模式下，如果函数有多个重名的参数，可以用 arguments[i] 读取。严格模式下，这属于语法错误。

正常模式下，整数的第一位如果是 0，表示这是八进制数，比如 0100 等于十进制的 64。严格模式禁止这种表示法，整数第一位为 0，将报错。

不允许对 arguments 赋值

arguments 不再追踪参数的变化

## 禁止使用 `arguments.callee`

严格模式只允许在全局作用域或函数作用域的顶层声明函数。也就是说，不允许在非函数的代码块内声明函数

严格模式新增了一些保留字：`implements`, `interface`, `let`, `package`, `private`, `protected`, `public`, `static`, `yield`。

## 30、如何判断一个对象是否属于某个类(严格来说在 ES6 之前，js 没有类的概念)？

`instanceof`    `constructor`

## 31、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` 。

## 32、用原生 JavaScript 的实现过什么功能吗？

主要考察原生 js 的实践经验

## 33、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`

用于检查给定的属性在当前对象实例中（而不是在实例的原型中）是否存在。返回 `true`，就代表该属性是实例对象的属性。而不是从原型继承过来的属性

## 34、对 JSON 的了解？

轻量级数据交互格式，可以形成复杂的嵌套格式，解析非常方便

## 35、js 延迟加载的方式有哪些？

方案一：`<script>` 标签的 `async="async"` 属性（详细参见：`script` 标签的 `async` 属性）

方案二：`<script>` 标签的 `defer="defer"` 属性

方案三：动态创建 `<script>` 标签

方案四：AJAX `eval`（使用 AJAX 得到脚本内容，然后通过 `eval_r(xmlhttp.responseText)` 来运行脚本）

方案五：`iframe` 方式

## 36、模块化开发怎么做？

理解模块化开发模式：浏览器端 requirejs, seajs；服务器端 nodejs；ES6 模块化；fis、webpack 等前端整体模块化解决方案；grunt、gulp 等前端工作流的使用

## 37、AMD (Modules/Asynchronous-Definition)、CMD (Common Module Definition)

规范区别？  
AMD 是 RequireJS 在推广过程中对模块定义的规范化产出。  
CMD 是 SeaJS 在推广过程中对模块定义的规范化产出。

理解这两种规范的差异，主要通过 requirejs 与 seajs 的对比，理解模块的定义与引用方式的差异以及这两种规范的设计原则

参考链接 1: <https://www.zhihu.com/question/20351507/answer/14859415>

参考链接 2: <https://github.com/seajs/seajs/issues/277>

- 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。
- 3、AMD 的 API 默认是一个当多个用，CMD 的 API 严格区分，推崇职责单一。比如 AMD 里，require 分全局 require 和局部 require，都叫 require。CMD 里，没有全局 require，而是根据模块系统的完备性，提供 seajs.use 来实现模块系统的加载启动。CMD 里，每个 API 都简单纯粹。

## 38、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？）

核心是 js 的加载模块，通过正则匹配模块以及模块的依赖关系，保证文件加载的先后顺序，根据文件的路径对加载过的文件做了缓存

## 39、让你自己设计实现一个 requireJS，你会怎么做？

核心是实现 js 的加载模块，维护 js 的依赖关系，控制好文件加载的先后顺序

## 40、谈一谈你对 ECMAScript6 的了解？

ES6 新的语法糖，类，模块化等新特性

关于 ES6 参考链接: <http://es6.ruanyifeng.com/>

1. [ECMAScript 6 简介](#)
2. [let 和 const 命令](#)
3. [变量的解构赋值](#)
4. [字符串的扩展](#)
5. [正则的扩展](#)
6. [数值的扩展](#)
7. [数组的扩展](#)

8. [函数的扩展](#)
9. [对象的扩展](#)
10. [Symbol](#)
11. [Proxy 和 Reflect](#)
12. [二进制数组](#)
13. [Set 和 Map 数据结构](#)
14. [Iterator 和 for...of 循环](#)
15. [Generator 函数](#)
16. [Promise 对象](#)
17. [异步操作和 Async 函数](#)
18. [Class](#)
19. [Decorator](#)
20. [Module](#)

#### 41、ECMAScript6 怎么写 class 么，为什么会出现 class 这种东西？

```
class Point {  
  constructor(x, y) {  
    this.x = x;  
    this.y = y;  
  }  
  toString() {  
    return '(' + this.x + ', ' + this.y + ')';  
  }  
}
```

#### 42、异步加载的方式有哪些？

方案一：<script>标签的 async="async" 属性（详细参见：[script 标签的 async 属性](#)）

方案二：<script>标签的 defer="defer" 属性

方案三：动态创建<script>标签

方案四：AJAX eval（使用 AJAX 得到脚本内容，然后通过 eval\_r(xmlhttp.responseText) 来运行脚本）

方案五：iframe 方式

#### 43、document.write 和 innerHTML 的区别？

document.write 是重写整个 document，写入内容是字符串的 html  
innerHTML 是 HTMLElement 的属性，是一个元素的内部 html 内容

#### 44、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_x()          //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()
(3) 查找
getElementsByTagName() //通过标签名称
getElementsByName() //通过元素的 Name 属性的值
getElementById() //通过元素 Id, 唯一性
```

## 45、call() 和 apply() 的含义和区别？

apply 的参数是数组形式，call 的参数是单个的值，除此之外在使用上没有差别，重点理解这两个函数调用的 this 改变

## 46、数组和对象有哪些原生方法，列举一下？

Array.concat() 连接数组  
Array.join() 将数组元素连接起来以构建一个字符串  
Array.length 数组的大小  
Array.pop() 删除并返回数组的最后一个元素  
Array.push() 给数组添加元素  
Array.reverse() 颠倒数组中元素的顺序  
Array.shift() 将元素移出数组  
Array.slice() 返回数组的一部分  
Array.sort() 对数组元素进行排序  
Array.splice() 插入、删除或替换数组的元素  
Array.toLocaleString() 把数组转换成局部字符串  
Array.toString() 将数组转换成一个字符串  
Array.unshift() 在数组头部插入一个元素

**Object 对象的常用方法** hasOwnProperty()方法可以检测一个属性是存在于实例中，还是存在于原型中。这个方法（不继承来的）只在给定属性存在于对象实例中时，才会返回 true。

Object.hasOwnProperty() 检查属性是否被继承

Object.isPrototypeOf() 一个对象是否是另一个对象的原型 alert(Person.prototype.isPrototypeOf(person2)); //true

Object.propertyIsEnumerable() 是否可以通过 for/in 循环看到属性

Object.toLocaleString() 返回对象的本地字符串表示

Object.toString() 定义一个对象的字符串表示

Object.valueOf() 指定对象的原始值

## 47、JS 怎么实现一个类。怎么实例化这个类

严格来讲 js 中并没有类的概念，不过 js 中的函数可以作为构造函数来使用，通过 new 来实例化，其实函数本身也是一个对象。

## 48、JavaScript 中的作用域与变量声明提升？

变量声明和函数定义提前

理解 JavaScript 的预解析机制，js 的运行主要分两个阶段：js 的预解析和运行，**预解析阶段所有的变量声明和函数定义都会提前，但是变量的赋值不会提前**

## 49、如何编写高性能的 Javascript？

文档片段

使用 **DocumentFragment** 优化多次 append

通过模板元素 clone，替代 createElement

使用一次 innerHTML 赋值代替构建 dom 元素

使用 firstChild 和 nextSibling 代替 childNodes 遍历 dom 元素

使用 Array 做为 StringBuffer，代替字符串拼接的操作

将循环控制量保存到局部变量

顺序无关的遍历时，用 while 替代 for

将条件分支，按可能性顺序从高到低排列

在同一条件子的多（>2）条件分支时，使用 switch 优于 if

使用三目运算符替代条件分支

需要不断执行的时候，优先考虑使用 setInterval

## 50、那些操作会造成内存泄漏？

闭包，死循环

## 51、javascript 对象的几种创建方式？

1. 工厂模式
2. 构造函数模式
3. 原型模式
4. 混合构造函数和原型模式
5. 动态原型模式
6. 寄生构造函数模式
7. 稳妥构造函数模式

```
function Person(name, age, job) {  
    var o = new Object();  
  
    // private members  
    var nameUC = name.toUpperCase();  
  
    // public members  
    o.sayName = function() {  
        alert(name);  
    };  
    o.sayNameUC = function() {  
        alert(nameUC);  
    };  
};
```

## 52、javascript 继承的 6 种方法？

1. 原型链继承
2. 借用构造函数继承
3. 组合继承(原型+借用构造)
4. 原型式继承
5. 寄生式继承
6. 寄生组合式继承

## 53、eval 是做什么的？ P132页

1. 它的功能是把对应的字符串解析成 JS 代码并运行
2. 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）

## 54、JavaScript 原型，原型链？有什么特点？

1. 原型对象也是普通的对象，是对象一个自带隐式的 `__proto__` 属性，原型也有可能有自己的原型，如果一个原型对象的原型不为 `null` 的话，我们就称之为原型链
2. 原型链是由一些用来继承和共享属性的对象组成的（有限的）对象链

## 55、事件、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();`

注意旧 ie 的方法：`ev.cancelBubble = true;`

## 56、简述一下 Sass、Less，且说明区别？

他们是动态的样式语言，是 CSS 预处理器，CSS 上的一种抽象层。他们是一种特殊的语法/语言而编译成 CSS。

变量符不一样，less 是 `@`，而 Sass 是 `$`；

Sass 支持条件语句，可以使用 `if {} else {}`，`for {}` 循环等等。而 Less 不支持；

Sass 是基于 Ruby 的，是在服务端处理的，而 Less 是需要引入 `less.js` 来处理 Less 代码输



出 Css 到浏览器

## 57、关于 javascript 中 apply() 和 call() 方法的区别？

相同点:两个方法产生的作用是完全一样的

不同点:方法传递的参数不同

`Object.call(this, obj1, obj2, obj3)`

`Object.apply(this, arguments)`

`apply()` 接收两个参数，一个是函数运行的作用域(this)，另一个是参数数组。

`call()` 方法第一个参数与 `apply()` 方法相同，但传递给函数的参数必须列举出来。

## 58、简述一下 JS 中的闭包？

闭包用的多的两个作用：读取函数内部的变量值；让这些变量值始终保存着(在内存中)。

同时需要注意的是：闭包慎用，不滥用，不乱用，由于函数内部的变量都被保存在内存中，会导致内存消耗大。

## 59、说说你对 this 的理解？

在 JavaScript 中，`this` 通常指向的是我们正在执行的函数本身，或者是，指向该函数所属的对象。

全局的 `this` → 指向的是 Window

~~函数中的 `this` → 指向的是函数所在的对象~~ 错误答案

对象中的 `this` → 指向其本身

事件中 `this` → 指向事件对象

## 60、分别阐述 split(), slice(), splice(), join()？

`join()` 用于把数组中的所有元素拼接起来放入一个字符串。所带的参数为分割字符串的分隔符，默认是以逗号分开。归属于 Array

`split()` 即把字符串分离开，以数组方式存储。归属于 String

`slice()` 方法可从已有的数组中返回选定的元素。该方法并不会修改数组，而是返回一个子数组。如果想删除数组中的一段元素，应该使用方法 `Array.splice()` **slice 截取数组元素, 含头不含尾**

`splice()` 方法向/从数组中添加/删除项目，然后返回被删除的项目。返回的是含有被删除的元素的数组。 **splice 查找、删除、替换数组元素**

## 61、事件委托是什么？

让利用事件冒泡的原理，让自己的所触发的事件，让他的父元素代替执行！

## 62、如何阻止事件冒泡和默认事件？

阻止浏览器的默认行为 `event.preventDefault()`方法是用于取消事件的默认行为，但此方法并不被ie支持，在ie下需要用`window.event.returnValue = false;`来实现。

`window.event?window.event.returnValue=false:e.preventDefault();`

停止事件冒泡

IE

非IE

`window.event?window.event.cancelBubble=true:e.stopPropagation();`

原生JavaScript中, `return false;`只阻止默认行为, 不阻止冒泡, jQuery中的`return false;`

既阻止默认行为, 又阻止冒泡

## 63、添加 删除 替换 插入到某个接点的方法？

`obj.appendChild()`

`obj.removeChild()`

`obj.replaceChild()`

`obj.insertBefore()`

## 64、你用过 require.js 吗？它有什么特性？

不会阻塞页面、按需加载、RequireJS 是一个JavaScript模块加载器。

<https://www.jianshu.com/p/c90fff39c225>

(1) 实现 js 文件的异步加载，避免网页失去响应；

(2) 管理模块之间的依赖性，便于代码的编写和维护。

## 65、谈一下 JS 中的递归函数，并且用递归简单实现阶乘？

递归即是程序在执行过程中不断调用自身的编程技巧，当然也必须要有有一个明确的结束条件，不然就会陷入死循环。

## 66、请用正则表达式写一个简单的邮箱验证。

`/^[a-zA-Z0-9_-]+@[a-zA-Z0-9_-]+(\.[a-zA-Z0-9_-]+)+$/;`

## 67、简述一下你对 web 性能优化的方案？

1、尽量减少 HTTP 请求

2、使用浏览器缓存

3、使用压缩组件

4、图片、JS 的预载入

5、将脚本放在底部

6、将样式文件放在页面顶部

7、使用外部的 JS 和 CSS

8、精简代码

68、在 JS 中有哪些会被隐式转换为 false

Undefined、null、布尔值 false、NaN、零、空字符串

69、定时器 setInterval 有一个有名函数 fn1，setInterval (fn1, 500) 与 setInterval (fn1(), 500) 有什么区别？

第一个是重复执行每 500 毫秒执行一次，后面一个只执行一次。

70、外部 JS 文件出现中文字符，会出现什么问题，怎么解决？

会出现乱码，加 charset="GB2312"；

另一种解决方式：网页文件和外部 JS 文件都是 UTF8 编码

71、谈谈浏览器的内核，并且说一下什么是内核？

Trident ([ˈtraɪd(ə)nt])--IE , Gecko ([ˈɡekəʊ])--Firefox, Presto ([ˈprestəʊ])--opera, webkit--谷歌和 Safari

浏览器内核又可以分成两部分：渲染引擎和 JS 引擎。它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。JS 引擎则是解析 Javascript 语言，执行 javascript 语言来实现网页的动态效果。

## 72、JavaScript 原型，原型链？有什么特点？

每个构造函数都有一个原型对象，原型对象都包含一个指向构造函数的指针，而实例都包含一个指向原型对象的内部指针。那么，假如我们让原型对象等于另一个类型的实例，结果会怎么样呢？显然，此时的原型对象将包含一个指向另一个原型的指针，相应地，另一个原型中也包含着一个指向另一个构造函数的指针。假如另一个原型又是另一个类型的实例，那么上述关系依然成立，如此层层递进，就构成了实例与原型的链条。这就是所谓原型链的基本概念。

\* 原型对象也是普通的对象，是对象一个自带隐式的 `__proto__` 属性，原型也有可能有自己的原型，如果一个原型对象的原型不为 null 的话，我们就称之为原型链。

\* 原型链是由一些用来继承和共享属性的对象组成的（有限的）对象链。

\* JavaScript 的数据对象有那些属性值？

writable: 这个属性的值是否可以改。

configurable: 这个属性的配置是否可以删除，修改。configurable 表示能否通过 delete 删除属性从而重新定义属性，能否修改属性的特性，或者能否把属性修改为访问器属性

enumerable: 这个属性是否能在 for...in 循环中遍历出来或在 Object.keys 中列举出来。enumerable 为 false 的属性，不能被 for in 遍历访问

value: 属性值。

\* 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。

```
function clone(proto) {  
  function Dummy() {}  
  Dummy.prototype = proto;  
  Dummy.prototype.constructor = Dummy;
```

```

    return new Dummy(); //等价于 Object.create(Person);
}

function object(old) {
    function F() {};
    F.prototype = old;
    return new F();
}

var newObj = object(oldObject);

```

## 73、写一个通用的事件侦听器函数

```

`// event(事件)工具集，来源：https://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.detachEvent) {
    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;
    }
};

```

#### 74、事件、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、~~火狐是事件捕获~~；
3. `ev.stopPropagation();`

#### 75、什么是闭包（closure），为什么要用？

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

```

#### 76、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');
}

```

## 77、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，~~同时还继承子该函数的原型。~~ 并将构造函数内部的this指向这个实例 继承构造函数的原型对象
- 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);

```

## 78、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

```
{ 'age': '12', 'name': 'back' }
```

## 79、js 延迟加载的方式有哪些

让js最后加载，放在页面底部

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

用document.createElement('<script src= "... "></script>') 动态创建DOM方式，来延迟加载js脚本

动态创建DOM方式（使用的最多）

## 80、模块化怎么做？

立即执行函数，不暴露私有成员

- 1、使用字面量实现命名空间(YUI)：

```

Itcast.common.dom={};
Itcast.common.css={};
Itcast.common.event={};

```

- 2、使用闭包

```

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

```

```

<script type="text/javascript">
function downloadJSAtOnload() {
    var element = document.createElement("script");
    element.src = "defer.js";
    document.body.appendChild(element);
}
if (window.addEventListener) //添加监听事件
    window.addEventListener("load",downloadJSAtOnload, false); //事件在冒泡
    阶段执行
else if (window.attachEvent)
    window.attachEvent("onload",downloadJSAtOnload);
else
    window.onload = downloadJSAtOnload;
</script>

```

```

    m1 : m1,
    m2 : m2
  };
})();

```

## 81、异步加载的方式

defer立即下载，延迟执行

- (1) defer，只支持 IE IE4、Firefox 3.5、Safari 5 和 Chrome 是最早支持 defer 属性的浏览器。其他浏览器会忽略这个属性，像平常一样处理脚本。
- (2) async：异步脚本。立即下载文件，只适用于外部脚本文件。在load时间前执行（不一定）
- (3) 创建 script，插入到 DOM 中，加载完毕后 callBack

document.write 和 innerHTML 的区别

document.write() : 会重写整个页面

document.write 只能重绘整个页面

innerHTML 可以重绘页面的一部分

## 82、告诉我答案是多少？

```

(function(x) {
    delete x;
    alert(x);
})(1+5);

```

函数参数无法 delete 删除，delete 只能删除通过 for in 访问的属性。

当然，删除失败也不会报错，所以代码运行会弹出“1”。

## 83、JS 中的 call() 和 apply() 方法的区别？

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);

```

## 84、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



## 85、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)
```

## 86、JavaScript 中的作用域与变量声明提升？

其他部分

(HTTP、正则、优化、重构、响应式、移动端、团队协作、SEO、UED、职业生涯)

\*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++) {}
```

## 87、前端开发的优化问题（看雅虎 14 条性能优化原则）。

参考资料：J:\代码,PPT,笔记,电子书\面试题\雅虎 14 条优化规则.docx

(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。

## 88、http 状态码有那些？分别代表是什么意思？

100-199 用于指定客户端应相应的某些动作。

200-299 用于表示请求成功。

300-399 用于已经移动的文件并且常被包含在定位头信息中指定新的地址信息。

400-499 用于指出客户端的错误。

400 语义有误，当前请求无法被服务器理解。

401 当前请求需要用户验证

403 服务器已经理解请求，但是拒绝执行它。

500-599 用于支持服务器错误。

503 - 服务不可用

## 89、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）

要熟悉前后端的通信流程，最好把动态网站的背后细节也介绍一遍

## 八、流行框架

### 1、JQuery 的源码看过吗？能不能简单概况一下它的实现原理？

考察学习知识的态度，是否仅仅是停留在使用层面，要知其然知其所以然

### 2、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.fn 指向的是 jQuery 的原型对象。jQuery 是个构造函数。因此 fn 的 init 方法返回的 this 指向具体的 jQuery 的实例对象  
this 执行 init 构造函数自身，其实就是 jQuery 实例对象，返回 this 是为了实现 jQuery 的链式操作

### 3、jquery 中如何将数组转化为 json 字符串，然后再转化回来？

```
$.parseJSON('{"name":"John"}');  
JSON.stringify
```

### 4、jQuery 的属性拷贝 (extend) 的实现原理是什么，如何实现深拷贝？

递归赋值

### 5、jquery.extend 与 jquery.fn.extend 的区别？

jQuery.extend 用来扩展 jQuery 对象本身；jQuery.fn.extend 用来扩展 jQuery 实例

2. jQuery.fn.extend() : 把对象挂载到 jQuery 的 prototype 属性，来扩展一个新的 jQuery 实例方法，也就是通过这个 extend 添加的新方法，实例化的 jQuery 对象都能使用，因为它是挂载在 jQuery.fn 上的方法。  
查看 jQuery 源码可发现，jQuery.fn = jQuery.prototype。jQuery.fn 挂在原型上，由于对原型的修改会影响所有实例，因此 fn 上的方法会对每一个 jQuery 实例有效。

#### 1、jQuery.extend()

(1) 扩展 jQuery 类本身，为 jQuery 类添加类方法（静态方法）

## 6、谈一下 Jquery 中的 bind(), live(), delegate(), on() 的区别?

四者的区别的相关帖子：<https://www.jianshu.com/p/8600a1bd0276>

jquery1.7 以后就推荐使用 on 的方式来进行事件绑定了

on()的特点：是以上三种方法的统一。可以使用一个方法，设置不同参数值来实现上述三种方法的功能。简化了jQuery代码库

## 7、JQuery 一个对象可以同时绑定多个事件，这是如何实现的？

链式操作

可以同时绑定多个事件，低层实现原理是使用 addEventListener 与 attachEvent 兼容处理做事件注册

## 4、Jquery 与 jQuery UI 有啥区别？

jQuery 是操作 dom 的框架，jQueryUI 是基于 jQuery 做的一个 UI 组件库

## 5、jQuery 和 Zepto 的区别？各自的使用场景？

jQuery 主要用于 pc 端，当然有对应的 jquerymobile 用于移动端，zepto 比 jQuery 更加小巧，主要用于移动端

jquery mobile 相对于 zepto 功能强大，但是体积也很庞大，zepto 非常的轻量

## 6、针对 jQuery 的优化方法？

- a、优先使用 ID 选择器
- b、jquery 获取到的 DOM 元素如果需要多次使用，建议使用一个变量将其保存起来，因为操作 DOM 的过程是非常耗费性能的
- c、在 class 前使用 tag(标签名)
- d、给选择器一个上下文 例如：父元素的选择器>子元素选择器 后代选择器 以这样一层一层的方式来限定和缩小查找范围
- e、慎用 .live() 方法（应该说尽量不要使用）
- f、使用 data() 方法存储临时变量

## 7、Zepto 的点透问题如何解决？ 相关帖子：<https://www.cnblogs.com/cdwp8/p/4307855.html>

什么叫点透问题？你可能碰到过在列表页面上创建一个弹出层，弹出层有个关闭的按钮，你点了这个按钮关闭弹出层后，这个按钮正下方的内容也会执行点击事件（或打开链接）。这个被定义为这是一个“点透”现象。

点透主要是由于两个 div 重合，例如：一个 div 调用 show()，一个 div 调用 hide()；

这个时候当点击上面的 div 的时候就会影响到下面的那个 div；

解决办法主要有 2 种： **方案一：来得很直接**github 上有 fastclick 可以完美解决<https://github.com/ftlabs/fastclick>

1. github 上有一个叫做 fastclick 的库，它也能规避移动设备上 click 事件的延迟响应，<https://github.com/ftlabs/fastclick> **方案二：用 touchend 代替 tap 事件并阻止掉 touchend 的默认行为 preventDefault()**

将它用 script 标签引入页面（该库支持 AMD，于是你也可以按照 AMD 规范，用诸如 require.js 的模块加载器引入），并且在 dom ready 时初始化在 body 上，

**方案三：延迟一定的时间(300ms+)来处理事件**

2. 根据分析，如果不引入其它类库，也不想自己按照上述 fastclick 的思路再开发一套东西，需要 1. 一个优先于下面的“divClickUnder”捕获的事件；2. 并且通过这个事件阻止掉默认行为（下面的“divClickUnder”对 click 事件的捕获，在 ios 的 safari，click 的捕获被认为和滚屏、点击输入框弹起键盘等一样，是一种浏览器默认行为，即可以被 event.preventDefault() 阻止的行为）。

12、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?

<https://blog.csdn.net/lj1530562965/article/details/72866181>

<https://v2ex.com/t/58926>

知识面的宽度, 流行框架要多多熟悉

angular、backbone、knockout 都是完整的 MV\* 框架

angular 是双向数据绑定的, backbone、knockout 是单向数据绑定的

React 只是单纯地 View 层

13、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?

Underscore 的熟悉程度

14、使用过 angular 吗? angular 中的过滤器是干什么用的

过程: 将组件模板 template 元素标签中绑定的原始数据, 通过管道, 传递给过滤器。  
过滤器的作用: 对原始数据进行处理, 转换为另一种新的数据输出

在表达式中转换数据 <p>姓名为 {{ lastName | uppercase }}</p>

currency, 是什么过滤器——格式化数字为货币格式, 单位是\$符。

## 九、移动 APP 开发

如何解决延时?

1. 禁止缩放: <meta name="viewport" content="width=device-width user-scalable='no'">
2. 使用 fastclick.js 轻量级库
3. 指针事件

1、移动端最小触控区域是多大?

相关帖子: <https://blog.csdn.net/xjun0812/article/details/64919063>  
<https://www.cnblogs.com/zhaodahai/p/6831165.html>

移动端的点击事件的有延迟, 时间是多长, 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。)

为何会出现 300ms 的延迟: 相关帖子

<https://blog.csdn.net/xjun0812/article/details/64919063>  
<https://www.cnblogs.com/zhaodahai/p/6831165.html>

## 十、NodeJs

1、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。

2、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。给出你的技术方案?

相关帖子: <https://www.naoffer.com/exam/319/935>

至少给出自己的思路 (url-hash, 可以使用已有的一些框架 history.js 等)

### 3、Node.js 的适用场景？

- 1)、**实时应用**：如在线聊天，实时通知推送等等（如 socket.io）
- 2)、**分布式应用**：通过高效的并行 I/O 使用已有的数据
- 3)、**工具类应用**：海量的工具，小到前端压缩部署（如 grunt），大到桌面图形界面应用程序
- 4)、**游戏类应用**：游戏领域对实时和并发有很高的要求（如网易的 pomelo 框架）
- 5)、**利用稳定接口提升 Web 渲染能力**
- 6)、**前后端编程语言环境统一**：前端开发人员可以非常快速地切入到服务器端的开发（如著名的纯 Javascript 全栈式 MEAN 架构）

### 4、(如果会用 node)知道 route, middleware, cluster, nodemon, pm2, server-side rendering 么？

Nodejs 相关概念的理解程度

### 5、解释一下 Backbone 的 MVC 实现方式？

Backbone 框架网址：<https://backbonejs.org/>

流行的 MVC 架构模式

### 6、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

何时使用前端路由：在单页面应用，大部分页面结构不变，只改变部分内容的时候使用

#### 缺点？

相关帖子：<https://www.cnblogs.com/mxsf/p/10897123.html>

前端路由的优缺点：

优点：

用户体验好，不需要每次都从服务器全部获取，快速展现给用户

缺点：

1. 不利于 SEO

2. 使用浏览器的前进，后退键的时候会重新发送请求，没有合理利用缓存

3. 单页面无法记住之前滚动的位置，无法在前进，后退的时候记住滚动的位置

熟悉前后端通信相关知识

**前端路由就是在不进行后端请求的情况下对页面进行跳转**

在单页面应用中，有多个页面组件。页面组件间跳转，不需要向服务端发送请求，请求新的静态页面。只是在前端的页面组件间跳转，实现页面跳转的效果。这就是一种前端路由。

### 7、对 Node 的优点和缺点提出了自己的看法？

优点：

1. 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。
2. 与 Node 代理服务器交互的客户端代码是由 javascript 语言编写的，因此客户端和服务端都用同一种语言编写，这是非常美妙的事情。

缺点：

1. 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。
2. 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（第三方库现在已经很丰富了，所以这个缺点可以说不存在了）。

## 十一、前端概括性问题

### 1、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？

使用率较高的框架有 jQuery、YUI、Prototype、Dojo、Ext.js、Mootools 等。尤其是 jQuery，超过 91%。

轻量级框架有 Modernizr、underscore.js、backbone.js、Raphael.js 等。（理解这些框架的功能、性能、设计原理）

前端开发工具：Sublime Text、Eclipse、Notepad、Firebug、HttpWatch、Yslow。

开发过的插件：城市选择插件，汽车型号选择插件、幻灯片插件。弹出层。（写过开源程序，加载器，js 引擎更好）

### 6、对 BFC 规范的理解？

Formatting Context：指页面中的一个渲染区域，并且拥有一套渲染规则，他决定了其子元素如何定位，以及与其他元素的相互关系和作用。

### 3、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）

### 4、WEB 应用从服务器主动推送 Data 到客户端有那些方式？

html5 websocket

WebSocket 通过 Flash

XHR 长时间连接

XHR Multipart Streaming

不可见的 Iframe

<script>标签的长时间连接(可跨域)

### 5、加班的看法

加班就像借钱，原则应当是-----救急不救穷

### 6、平时如何管理你的项目，如何设计突发大规模并发架构？

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

内存泄漏的操作有哪些：

1. 意外的全局变量引起的内存泄漏
2. 闭包引起的内存泄漏
3. 没有清理的DOM元素引用
4. 被遗忘的定时器或者回调
5. 子元素存在引用引起的内存泄漏

## 7、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）

## 8、你说你热爱前端，那么应该 WEB 行业的发展很关注吧？ 说说最近最流行的一些东西吧？

Node.js、Mongodb、npm、MVVM、MEAN、react、angularjs、browserify、webpack

## 9、你有了解我们公司吗？说说你的认识？

因为我想去阿里，所以我针对阿里的说

最羡慕就是在双十一购物节，350.19 亿元，每分钟支付 79 万笔。海量数据，居然无一漏单、无一故障。太厉害了。

携程、去哪儿：一个主要从事在线旅游 O2O，当然也可以订酒店，买车票，买机票

## 10、移动端（比如：Android IOS）怎么做好用户体验？

融入自己的设计理念，注重用户体验，选择合适的技术

## 11、 你所知道的页面性能优化方法有那些？

压缩、合并，减少请求，代码层析优化。。。

## 12、 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？

知识面宽度，最好熟悉一些后台语言，比如 php，展现出自己的技术两点

## 13、AMD(Modules/Asynchronous-Definition)、CMD(Common Module Definition) 规范区别？ <https://cloud.tencent.com/developer/article/1408250>

## 14、谈谈你认为怎样做能使项目做的更好？

考虑问题的深入，不仅仅停留在完成任务上，要精益求精

15、你对前端界面工程师这个职位是怎么样理解的？它的前景会怎么样？

表现出对前端的认同与兴趣，关注相关技术前沿

16、php 中下面哪个函数可以打开一个文件，以对文件进行读和写操作？

A. fget(); B. file\_open(); C. fopen(); D. open\_file();

17、php 中 rmdir 可以直接删除文件夹吗？该目录必须是空的，而且要有相应的权限——来自 api

- A. 任何文件夹都可以删除                      B. 空文件夹可以删除  
C. 有权限的任何文件夹都可以删除      D. 有权限的空文件夹可以删除

18、phpinset 和 empty 的区别，举例说明

1、empty 函数

用途：检测变量是否为空

判断：如果 var 是非空或非零的值，则 empty() 返回 FALSE。换句话说，“”、0、“0”、NULL、FALSE、array()、var \$var; 以及没有任何属性的对象都将被认为是空的，如果 var 为空，则返回 TRUE。注意：empty() 只检测变量，检测任何非变量的东西都将导致解析错误。换句话说，后边的语句将不会起作用；

2、isset 函数

用途：检测变量是否设置

判断：检测变量是否设置，并且不是 NULL。如果已经使用 unset() 释放了一个变量之后，它将不再是 isset()。若使用 isset() 测试一个被设置成 NULL 的变量，将返回 FALSE。同时要注意的是一个 NULL 字节 (“\0”) 并不等同于 PHP 的 NULL 常数。

19、php 中 \$\_SERVER 变量中如何得到当前执行脚本路劲

```
__FILE__: =====> G:\web\test\t2\dir.php
__DIR__: =====> G:\web\test\t2
dirname(__FILE__): =====> G:\web\test\t2
$_SERVER["PHP_SELF"]: =====> /test/t2/dir.php
$_SERVER["SCRIPT_NAME"]: =====> /test/t2/dir.php
$_SERVER["SCRIPT_FILENAME"]: =====> G:/web/test/t2/dir.php
$_SERVER["DOCUMENT_ROOT"]: =====> G:/web
getcwd(): =====> G:\web\test\t2
```



20、写一个 php 函数, 要求两个日期字符串的天数差, 如 2012-02-05~2012-03-06 的日期差数

21、一个衣柜中放了许多杂乱的衬衫, 如果让你去整理一下, 使得更容易找到你想要的衣服; 你会怎么做? 请写出你的做法和思路?

22、如何优化网页加载速度?

1. 减少 css, js 文件数量及大小(减少重复性代码, 代码重复利用), 压缩 CSS 和 Js 代码

2. 图片的大小

3. 把 css 样式表放置顶部, 把 js 放置页面底部

4. 减少 http 请求数

5. 使用外部 Js 和 CSS

23、工作流程, 你怎么来实现页面设计图, 你认为前端应该如何高质量完成工作?

熟悉相关设计规范, 自己总结的一些经验

24、介绍项目经验、合作开发、独立开发。

团队协作, 个人能力。实践经验

25、开发过程中遇到困难, 如何解决。

考察解决问题的能力

26、对前端界面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;

响应式布局的关键技术是CSS3中的媒体查询，监测设备屏幕大小，通过css媒体查询来有针对性的更改页面的布局，可以监测屏幕方向(移动设备)，设备类型等等，核心在于感知。在不同屏幕下可以显示不同版式。

做好的页面结构，页面重构和用户体验；

流式布局在CSS2时代就有，主要是靠百分比进行排版，可以在不同分辨率下显示相同的版式。

处理 hack，兼容、写出优美的代码格式；

流式布局：网页中主要的划分区域的尺寸使用百分数（搭配min-\*、max-\*属性使用），例如，设置网页主体的宽度为80%，min-width为960px。图片也作类似处理（width:100%，max-width一般设定为图片本身的尺寸，防止被拉伸而失真）。

针对服务器的优化、拥抱最新前端技术。

这种布局方式在Web前端开发的早期历史上，用来应对不同尺寸的PC屏幕（那是屏幕尺寸的差异不会太大），在当今的移动端开发也是常用布局方式，但缺点明显：宽度使用百分比定义，但是高度和文字大小等都是用px来固定，所以在大屏幕的手机上显示效果会变成有些页面元素宽度被拉的很长，但是高度、文字大小还是和原来一样（即，这些东西无法变得“流式”），显示非常不协调

其它相关的加分项：

1. 都使用和了解过哪些编辑器?都使用和了解过哪些日常工具?

2. 都知道有哪些浏览器内核?开发过的项目都兼容哪些浏览器?

Trident (IE), Gecko (Firefox), Presto (Opera), Webkit (苹果), Blink (Google)

3. 瀑布流布局或者流式布局是否有了解

瀑布流，又称瀑布流式布局。是比较流行的一种网站页面布局，视觉表现为参差不齐的多栏布局，随着页面滚动条向下滚动，这种布局还会不断加载数据块并附加至当前尾部。

4. HTML5 都有哪些新的 API?

HTML5 - 新特性：  
1. 用于绘画的 canvas 元素，SVG矢量图  
2. 用于媒介回放的 video 和 audio 元素  
3. 对本地离线存储的更好的支持，sessionStorage/localStorage  
4. 语义化标签，比如 article、footer、header、nav、section  
5. 新的表单控件，比如 calendar、date、time、email、url、search

5. 都用过什么代码调试工具?

Chrome 和火狐的Firebug

6. 是否有接触过或者了解过重构。

为何要重构：现在的网页正在变得越来越丰富，功能也越来越多。这也同时意味着系统正在变得越来越复杂，而越复杂的系统就越需要一个好的前端页面来缓冲给用户带来的这种复杂性。 重构：不断调整前端的架构，更好地为用户服务

7. 你遇到过比较难的技术问题是？你是如何解决的？

重构的核心思想之一 解耦。这样的设计有什么好处：

1. 功能之间解耦，易于维护与扩展
2. 逻辑清晰，便于接口间的互相调用

备注：对网页进行布局时，布局开发的方向就是先从左到右，再从上到下，从整体到局部，化整为零！

<https://scala.cool/tags/%E5%90%8E%E7%AB%AF%E5%B7%A5%E7%A8%8B%E5%B8%88%E5%85%A5%E9%97%A8%E5%89%8D%E7%AB%AF%E9%A1%B5%E9%9D%A2%E9%87%8D%E6%9E%84/>

布局相关帖子：

[https://scala.cool/2017/12/be\\_2\\_fe/](https://scala.cool/2017/12/be_2_fe/)

<https://scala.cool/2018/02/back-2-font-xinfa-1/>