前端面试题

以下内容为极客学院自己整理，供内部阅读学习，请勿外传。

极客学院

目录

[面试有几点需注意： 6](#_Toc459887470)

[前端开发面试知识点大纲： 6](#_Toc459887471)

[HTML 6](#_Toc459887472)

[1.XHTML和HTML有什么区别 6](#_Toc459887473)

[2.Doctype? 严格模式与混杂模式-如何触发这两种模式，区分它们有何意义 6](#_Toc459887474)

[3.常见兼容性问题 6](#_Toc459887475)

[4. 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 7](#_Toc459887476)

[5.语义化的理解 8](#_Toc459887477)

[6.HTML5的离线储存 8](#_Toc459887478)

[7.(写)描述一段语义的html代码 8](#_Toc459887479)

[8.iframe有那些缺点 8](#_Toc459887480)

[9.请描述一下 cookies，sessionStorage 和 localStorage 的区别 9](#_Toc459887481)

[10.行内元素有哪些?块级元素有哪些?空(void)元素有那些 9](#_Toc459887482)

[11. 你知道多少种Doctype文档类型 9](#_Toc459887483)

[12. 浮动元素引起的问题和解决办法 9](#_Toc459887484)

[13. IE 8以下版本的浏览器中的盒模型有什么不同 10](#_Toc459887485)

[14. DOM操作——怎样添加、移除、移动、复制、创建和查找节点 10](#_Toc459887486)

[15. 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 10](#_Toc459887487)

[16.如何实现浏览器内多个标签页之间的通信 11](#_Toc459887488)

[17.webSocket如何兼容低浏览器 11](#_Toc459887489)

[18.线程与进程的区别 11](#_Toc459887490)

[19.你如何对网站的文件和资源进行优化 11](#_Toc459887491)

[20.请说出三种减少页面加载时间的方法 11](#_Toc459887492)

[21.你都使用哪些工具来测试代码的性能 12](#_Toc459887493)

[22.什么是 FOUC（无样式内容闪烁）？你如何来避免 FOUC 12](#_Toc459887494)

[23.null和undefined的区别 12](#_Toc459887495)

[24.如何判断当前脚本运行在浏览器还是node环境中 12](#_Toc459887496)

[25、请你谈谈Cookie的弊端 12](#_Toc459887497)

[26.浏览器本地存储 13](#_Toc459887498)

[27.web storage和cookie的区别 13](#_Toc459887499)

[CSS 14](#_Toc459887500)

[1.介绍一下CSS的盒子模型 14](#_Toc459887501)

[2. 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新增伪类有哪些 14](#_Toc459887502)

[3.如何居中div？如何居中一个浮动元素 15](#_Toc459887503)

[4.列出display的值，说明他们的作用。position的值， relative和absolute定位原点是 15](#_Toc459887504)

[5.CSS3有哪些新特性 15](#_Toc459887505)

[6.一个满屏品字布局如何设计 16](#_Toc459887506)

[7.为什么要初始化CSS样式 16](#_Toc459887507)

[8.absolute的containing block计算方式跟正常流有什么不同 17](#_Toc459887508)

[9.position跟display、margin collapse、overflow、float这些特性相互叠加后会怎么样 18](#_Toc459887509)

[10.对BFC规范的理解 19](#_Toc459887510)

[11.css定义的权重 19](#_Toc459887511)

[12.解释下浮动和它的工作原理？清除浮动的技巧 20](#_Toc459887512)

[13.对WEB标准以及W3C的理解与认识 20](#_Toc459887513)

[14.CSS引入的方式有哪些? link和@import的区别是 20](#_Toc459887514)

[15.CSS选择符有哪些? 哪些属性可以继承?优先级算法如何计算?内联和important哪个优先级高 21](#_Toc459887515)

[16.前端页面有哪三层构成，分别是什么?作用是什么 21](#_Toc459887516)

[17.css的基本语句构成是 21](#_Toc459887517)

[18.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 21](#_Toc459887518)

[19.写出几种IE6 BUG的解决方法 22](#_Toc459887519)

[20.标签上title与alt属性的区别是什么 22](#_Toc459887520)

[21.描述css reset的作用和用途 22](#_Toc459887521)

[22.解释css sprites，如何使用 22](#_Toc459887522)

[23.浏览器标准模式和怪异模式之间的区别是什么 23](#_Toc459887523)

[24.你如何对网站的文件和资源进行优化?期待的解决方案包括： 23](#_Toc459887524)

[25.清除浮动的几种方式，各自的优缺点 23](#_Toc459887525)

[26. 什么叫优雅降级和渐进增强 23](#_Toc459887526)

[27.display:none和visibility:hidden的区别 23](#_Toc459887527)

[28. position的absolute与fixed共同点与不同点 23](#_Toc459887528)

[29. 解释下 CSS sprites，以及你要如何在页面或网站中使用它 24](#_Toc459887529)

[JavaScript 24](#_Toc459887530)

[1.JavaScript原型，原型链,有什么特点 24](#_Toc459887531)

[2.eval是做什么的 24](#_Toc459887532)

[3.写一个通用的事件侦听器函数 25](#_Toc459887533)

[4.Node.js的适用场景 27](#_Toc459887534)

[5.介绍js的基本数据类型 27](#_Toc459887535)

[6.Javascript如何实现继承 27](#_Toc459887536)

[7.["1", "2", "3"].map(parseInt) 答案是多少 27](#_Toc459887537)

[8.如何创建一个对象? （画出此对象的内存图） 27](#_Toc459887538)

[9.谈谈This对象的理解 28](#_Toc459887539)

[10.事件是，IE与火狐的事件机制有什么区别，如何阻止冒泡 28](#_Toc459887540)

[12.什么是闭包（closure），为什么要用它 28](#_Toc459887541)

[13."use strict";是什么意思，使用它的好处和坏处分别是什么 28](#_Toc459887542)

[14.如何判断一个对象是否属于某个类 29](#_Toc459887543)

[15.new操作符具体干了什么呢 29](#_Toc459887544)

[16.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 29](#_Toc459887545)

[17.JSON 的了解 29](#_Toc459887546)

[18.js延迟加载的方式有哪些 29](#_Toc459887547)

[19.ajax 是什么? 同步和异步的区别? 如何解决跨域问题 29](#_Toc459887548)

[20. 对Node的优点和缺点提出了自己的看法 30](#_Toc459887549)

[21. 说出这道题的答案 30](#_Toc459887550)

[22.模块化怎么做 30](#_Toc459887551)

[24.异步加载的方式有哪些 30](#_Toc459887552)

[25.documen.write和 innerHTML的区别 30](#_Toc459887553)

[26..call() 和 .apply() 的区别 31](#_Toc459887554)

[27.jQuery与jQuery UI 有啥区别 31](#_Toc459887555)

[28.jquery 中如何将数组转化为json字符串，然后再转化回来 31](#_Toc459887556)

[29.针对 jQuery 的优化方法 31](#_Toc459887557)

[30.JavaScript中的作用域与变量声明提升 32](#_Toc459887558)

[31. javascript的typeof返回哪些数据类型 32](#_Toc459887559)

[32.例举3种强制类型转换和2种隐式类型转换? 32](#_Toc459887560)

[33.split() join() 的区别 32](#_Toc459887561)

[34.数组方法pop() push() unshift() shift() 32](#_Toc459887562)

[35.IE和DOM事件流的区别 32](#_Toc459887563)

[36.IE和标准下有哪些兼容性的写法 32](#_Toc459887564)

[37.ajax请求的时候get 和post方式的区别 33](#_Toc459887565)

[38.call和apply的区别 33](#_Toc459887566)

[39.ajax请求时，如何解释json数据 33](#_Toc459887567)

[40.写一个获取非行间样式的函数 33](#_Toc459887568)

[41.事件委托是什么 33](#_Toc459887569)

[42.如何阻止事件冒泡和默认事件 33](#_Toc459887570)

[43.添加 删除 替换 插入到某个接点的方法 34](#_Toc459887571)

[44.解释jsonp的原理，以及为什么不是真正的ajax 34](#_Toc459887572)

[45.javascript的本地对象，内置对象和宿主对象 34](#_Toc459887573)

[49.document load 和document ready的区别 34](#_Toc459887574)

[50.”==”和“===”的不同 34](#_Toc459887575)

[51.javascript的同源策略 34](#_Toc459887576)

[52.编写一个数组去重的方法 34](#_Toc459887577)

[53. 哪些操作会造成内存泄漏 35](#_Toc459887578)

[54. jquery 中如何将数组转化为json字符串，然后再转化回来 35](#_Toc459887579)

[55. Jquery与jQuery UI 有什么区别 35](#_Toc459887580)

[初级Javascript： 39](#_Toc459887581)

[1.JavaScript的数据类型都有什么 39](#_Toc459887582)

[2.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) 40](#_Toc459887583)

[3.希望获取到页面中所有的checkbox怎么做？(不使用第三方框架) 40](#_Toc459887584)

[4.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) 40](#_Toc459887585)

[5.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 40](#_Toc459887586)

[6.什么是Ajax和JSON，它们的优缺点 41](#_Toc459887587)

[7.看下列代码输出为何，解释原因 41](#_Toc459887588)

[8.看下列代码,输出什么，解释原因 41](#_Toc459887589)

[9.看下列代码,输出什么，解释原因 41](#_Toc459887590)

[10.看代码给答案 42](#_Toc459887591)

[11.已知数组var stringArray = [“This”, “is”, “Baidu”, “Campus”]，Alert出”This is Baidu Campus”。 43](#_Toc459887592)

[12.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”。 43](#_Toc459887593)

[13.var numberArray = [3,6,2,4,1,5]; （考察基础API） 43](#_Toc459887594)

[12.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 43](#_Toc459887595)

[13.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） 44](#_Toc459887596)

[14.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 44](#_Toc459887597)

[15.foo = foo||bar ，这行代码是什么意思？为什么要这样写 45](#_Toc459887598)

[16.看下列代码，将会输出什么(变量声明提升) 45](#_Toc459887599)

[17.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。 45](#_Toc459887600)

[18.把两个数组合并，并删除第二个元素。 46](#_Toc459887601)

[19.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） 46](#_Toc459887602)

[20.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}。 46](#_Toc459887603)

[21.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 47](#_Toc459887604)

[22.看下面代码，给出输出结果 47](#_Toc459887605)

[23.写一个function，清除字符串前后的空格。（兼容所有浏览器） 48](#_Toc459887606)

[24.Javascript中callee和caller的作用 48](#_Toc459887607)

[中级Javascript： 49](#_Toc459887608)

[1.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 49](#_Toc459887609)

[2.如何消除一个数组里面重复的元素 50](#_Toc459887610)

[3.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 51](#_Toc459887611)

[4.下面这个ul，如何点击每一列的时候alert其index?（闭包） 52](#_Toc459887612)

[5.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。 54](#_Toc459887613)

[6.请评价以下代码并给出改进意见。 56](#_Toc459887614)

[7.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 57](#_Toc459887615)

[8.定义一个log方法，让它可以代理console.log的方法 58](#_Toc459887616)

[9.在Javascript中什么是伪数组？如何将伪数组转化为标准数组 59](#_Toc459887617)

[10.对作用域上下文和this的理解，看下列代码： 59](#_Toc459887618)

[11.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 60](#_Toc459887619)

[12.（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） 62](#_Toc459887620)

[13.说出以下函数的作用是？空白区域应该填写什么 62](#_Toc459887621)

[JavaScript 62](#_Toc459887622)

[其他问题 64](#_Toc459887623)

# 面试有几点需注意：

**面试题目： 根据你的等级和职位变化，入门级到专家级：范围↑、深度↑、方向↑。**

1. 题目类型： 技术视野、项目细节、理论知识题，[**算法**](http://lib.csdn.net/base/31)题，开放性题，案例题。
2. 进行追问： 可以确保问到你开始不懂或面试官开始不懂为止，这样可以大大延展题目的区分度和深度，知道你的实际能力。因为这种关联知识是长时期的学习，绝对不是临时记得住的。
3. 回答问题再棒，面试官（可能是你的直接领导面试），会考虑我要不要这个人做我的同事？所以态度很重要。（感觉更像是相亲）
4. 资深的工程师能把 absolute 和 relative 弄混，这样的人不要也罢，因为团队需要的你这个人具有可以依靠的才能（靠谱）。

# [前端开发](http://lib.csdn.net/base/22)面试知识点大纲：

HTML&CSS：

对Web标准的理解、浏览器内核差异、兼容性、hack、CSS基本功：布局、盒子模型、选择器优先级及使用、[HTML5](http://www.cnblogs.com/oooweb/)、[CSS3](http://www.cnblogs.com/oooweb/)、移动端适应

JavaScript：

数据类型、面向对象、继承、闭包、插件、作用域、跨域、原型链、模块化、自定义事件、内存泄漏、事件机制、异步装载回调、模板引擎、Nodejs、JSON、ajax等。

其他：

HTTP、安全、正则、优化、重构、响应式、移动端、团队协作、可维护、SEO、UED、架构、职业生涯

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

2、DOM操作 ——如何添加、移除、移动、复制、创建和查找节点等。

3、事件 —— 如何使用事件，以及IE和标准DOM事件模型之间存在的差别。

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

5、严格模式与混杂模式 —— 如何触发这两种模式，区分它们有何意义。

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

7、块级元素与行内元素 —— 怎么用CSS控制它们、以及如何合理的使用它们

8、浮动元素——怎么使用它们、它们有什么问题以及怎么解决这些问题。

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

10、JSON —— 作用、用途、设计结构。

# HTML

## 1.XHTML和HTML有什么区别

HTML是一种基本的WEB网页设计语言，XHTML是一个基于XML的置标语言

最主要的不同：

XHTML 元素必须被正确地嵌套。

XHTML 元素必须被关闭。

标签名必须用小写字母。

XHTML 文档必须拥有根元素。

## 2.Doctype? 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

<!DOCTYPE> 声明位于文档中的最前面，处于 <html> 标签之前。告知浏览器的解析器， 用什么文档类型 规范来解析这个文档。

严格模式的排版和 JS 运作模式是 以该浏览器支持的最高标准运行。

在混杂模式中，页面以宽松的向后兼容的方式显示。模拟老式浏览器的行为以防止站点无法工作。

DOCTYPE不存在或格式不正确会导致文档以混杂模式呈现。

用于声明文档使用那种规范（html/xhtml）一般为 严格 过度 基于框架的html文档

加入XMl声明可触发，解析方式更改为IE5.5 拥有IE5.5的bug

## 3.常见兼容性问题

\* png24位的图片在iE6浏览器上出现背景，解决方案是做成PNG8.

\* 浏览器默认的margin和padding不同。解决方案是加一个全局的\*{margin:0;padding:0;}来统一。

\* IE6双边距bug:块属性标签float后，又有横行的margin情况下，在ie6显示margin比设置的大。

浮动ie产生的双倍距离 #box{ float:left; width:10px; margin:0 0 0 100px;}

这种情况之下IE会产生20px的距离，解决方案是在float的标签样式控制中加入 ——\_display:inline;将其转化为行内属性。(\_这个符号只有ie6会识别)

渐进识别的方式，从总体中逐渐排除局部。

首先，巧妙的使用“\9”这一标记，将IE游览器从所有情况中分离出来。

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

css

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

\* IE下,可以使用获取常规属性的方法来获取自定义属性, 也可以使用getAttribute()获取自定义属性; Firefox下,只能使用getAttribute()获取自定义属性. 解决方法:统一通过getAttribute()获取自定义属性. \* IE下,even对象有x,y属性,但是没有pageX,pageY属性; Firefox下,event对象有pageX,pageY属性,但是没有x,y属性. \* 解决方法：（条件注释）缺点是在IE浏览器下可能会增加额外的HTTP请求数。 \* Chrome 中文界面下默认会将小于 12px 的文本强制按照 12px 显示, 可通过加入 CSS 属性 -webkit-text-size-adjust: none; 解决. 超链接访问过后hover样式就不出现了 被点击访问过的超链接样式不在具有hover和active了解决方法是改变CSS属性的排列顺序: L-V-H-A : a:link {} a:visited {} a:hover {} a:active {}

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

\* HTML5 现在已经不是 SGML 的子集，主要是关于图像，位置，存储，多任务等功能的增加。

\* 绘画 canvas

用于媒介回放的 video 和 audio 元素

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

sessionStorage 的数据在浏览器关闭后自动删除

语意化更好的内容元素，比如 article、footer、header、nav、section

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

新的技术webworker, 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；

支持HTML5新标签：

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

可以利用这一特性让这些浏览器支持HTML5新标签，

浏览器支持新标签后，还需要添加标签默认的样式：

\* 当然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

## 5.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。

## 6.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。

## 7.(写)描述一段语义的html代码

（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>

## 8.iframe有那些缺点

\*iframe会阻塞主页面的Onload事件；

\*iframe和主页面共享连接池，而浏览器对相同域的连接有限制，所以会影响页面的并行加载。

使用iframe之前需要考虑这两个缺点。如果需要使用iframe，最好是通过javascript

动态给iframe添加src属性值，这样可以可以绕开以上两个问题。

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

cookie在浏览器和服务器间来回传递。 sessionStorage和localStorage不会

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；

## 10.行内元素有哪些?块级元素有哪些?空(void)元素有那些

块级元素：div p h1 h2 h3 h4 form ul

行内元素: a b br i span input select

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>

## 11. 你知道多少种Doctype文档类型

该标签可声明三种 DTD 类型，分别表示严格版本、过渡版本以及基于框架的 HTML 文档。

HTML 4.01 规定了三种文档类型：Strict、Transitional 以及 Frameset。

XHTML 1.0 规定了三种 XML 文档类型：Strict、Transitional 以及 Frameset。

Standards （标准）模式（也就是严格呈现模式）用于呈现遵循最新标准的网页，而 Quirks

（包容）模式（也就是松散呈现模式或者兼容模式）用于呈现为传统浏览器而设计的网页。

## 12. 浮动元素引起的问题和解决办法

浮动元素引起的问题：

（1）父元素的高度无法被撑开，影响与父元素同级的元素

（2）与浮动元素同级的非浮动元素会跟随其后

（3）若非第一个元素浮动，则该元素之前的元素也需要浮动，否则会影响页面显示的结构

解决方法： 使用CSS中的clear:both;属性来清除元素的浮动可解决2、3问题，对于问题1，添加如下样式，给父元素添加clearfix样式：

.clearfix:after{content: ".";display: block;height: 0;clear: both;visibility: hidden;}

.clearfix{display: inline-block;} /\* for IE/Mac \*/

清除浮动的几种方法： 1，额外标签法，<div style="clear:both;"></div>（缺点：不过这个办法会增加额外的标签使HTML结构看起来不够简洁。）2，使用after伪类

#parent:after{

content:".";

height:0;

visibility:hidden;

display:block;

clear:both;

}

3,浮动外部元素

4,设置`overflow`为`hidden`或者auto

## 13. IE 8以下版本的浏览器中的盒模型有什么不同

IE8以下浏览器的盒模型中定义的元素的宽高不包括内边距和边框

## 14. 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

## 15. 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。

\* 拖拽释放(Drag and drop) API

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

音频、视频API(audio,video)

画布(Canvas) API

地理(Geolocation) API

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

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

新的技术webworker, websocket, Geolocation

\* 移除的元素

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

对可用性产生负面影响的元素：frame，frameset，noframes；

支持HTML5新标签：

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

可以利用这一特性让这些浏览器支持HTML5新标签，

浏览器支持新标签后，还需要添加标签默认的样式：

\* 当然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

## 16.如何实现浏览器内多个标签页之间的通信

调用localstorge、cookies等本地存储方式

## 17.webSocket如何兼容低浏览器

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

## 18.线程与进程的区别

一个程序至少有一个进程,一个进程至少有一个线程.

线程的划分尺度小于进程，使得多线程程序的并发性高。

另外，进程在执行过程中拥有独立的内存单元，而多个线程共享内存，从而极大地提高了程序的运行效率。

线程在执行过程中与进程还是有区别的。每个独立的线程有一个程序运行的入口、顺序执行序列和程序的出口。但是线程不能够独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制。

从逻辑角度来看，多线程的意义在于一个应用程序中，有多个执行部分可以同时执行。但操作系统并没有将多个线程看做多个独立的应用，来实现进程的调度和管理以及资源分配。这就是进程和线程的重要区别。

## 19.你如何对网站的文件和资源进行优化

期待的解决方案包括：

文件合并；文件最小化/文件压缩；使用 CDN 托管；缓存的使用（多个域名来提供缓存）其他

## 20.请说出三种减少页面加载时间的方法

1.优化图片

2.图像格式的选择（GIF：提供的颜色较少，可用在一些对颜色要求不高的地方）

3.优化CSS（压缩合并css，如margin-top,margin-left...)

4.网址后加斜杠（如www.campr.com/目录，会判断这个“目录是什么文件类型，或者是目录。）

5.标明高度和宽度（如果浏览器没有找到这两个参数，它需要一边下载图片一边计算大小，如果图片很多，浏览器需要不断地调整页面。这不但影响速度，也影响浏览体验。

当浏览器知道了高度和宽度参数后，即使图片暂时无法显示，页面上也会腾出图片的空位，然后继续加载后面的内容。从而加载时间快了，浏览体验也更好了。）

6.减少http请求（合并文件，合并图片）。

## 21.你都使用哪些工具来测试代码的性能

Profiler, JSPerf（<http://jsperf.com/nexttick-vs-setzerotimeout-vs-> settimeout）, Dromaeo

## 22.什么是 FOUC（无样式内容闪烁）？你如何来避免 FOUC

FOUC - Flash Of Unstyled Content 文档样式闪烁

<style type="text/css" media="all">@import "../fouc.css";</style>

而引用CSS文件的@import就是造成这个问题的罪魁祸首。IE会先加载整个HTML文档的DOM，然后再去导入外部的CSS文件，因此，在页面DOM加载完成到CSS导入完成中间会有一段时间页面上的内容是没有样式的，这段时间的长短跟网速，电脑速度都有关系。

解决方法简单的出奇，只要在<head>之间加入一个<link>或者<script>元素就可以了。

## 23.null和undefined的区别

null是一个表示"无"的对象，转为数值时为0；undefined是一个表示"无"的原始值，转为数值时为NaN。

当声明的变量还未被初始化时，变量的默认值为undefined。 null用来表示尚未存在的对象，常用来表示函数企图返回一个不存在的对象。

undefined表示"缺少值"，就是此处应该有一个值，但是还没有定义。典型用法是：

（1）变量被声明了，但没有赋值时，就等于undefined。

（2) 调用函数时，应该提供的参数没有提供，该参数等于undefined。

（3）对象没有赋值的属性，该属性的值为undefined。

（4）函数没有返回值时，默认返回undefined。

null表示"没有对象"，即该处不应该有值。典型用法是：

（1） 作为函数的参数，表示该函数的参数不是对象。

（2） 作为对象原型链的终点。

## 24.如何判断当前脚本运行在浏览器还是node环境中

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

## 25、请你谈谈Cookie的弊端

cookie虽然在持久保存客户端数据提供了方便，分担了服务器存储的负担，但还是有很多局限性的。 第一：每个特定的域名下最多生成20个cookie

a.IE6或更低版本最多20个cookie

b.IE7和之后的版本最后可以有50个cookie。

c.Firefox最多50个cookie

d.chrome和Safari没有做硬性限制

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

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

IE 提供了一种存储可以持久化用户数据，叫做uerData，从IE5.0就开始支持。每个数据最多128K，每个域名下最多1M。这个持久化数据放在缓存中，如果缓存没有清理，那么会一直存在。

优点：极高的扩展性和可用性

a.通过良好的编程，控制保存在cookie中的session对象的大小。

b.通过加密和安全传输技术（SSL），减少cookie被破解的可能性。

c.只在cookie中存放不敏感数据，即使被盗也不会有重大损失。

d.控制cookie的生命期，使之不会永远有效。偷盗者很可能拿到一个过期的cookie。

缺点：

a.`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。

b.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。

c.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。

## 26.浏览器本地存储

在较高版本的浏览器中，js提供了sessionStorage和globalStorage。在HTML5中提供了localStorage来取代globalStorage。

html5中的Web Storage包括了两种存储方式：sessionStorage和localStorage。

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。

## 27.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

浏览器的支持除了IE７及以下不支持外，其他标准浏览器都完全支持(ie及FF需在web服务器里运行)，值得一提的是IE总是办好事，例如IE7、 IE6中的UserData其实就是javascript本地存储的解决方案。通过简单的代码封装可以统一到所有的浏览器都支持web storage。

localStorage和sessionStorage都具有相同的操作方法，例如setItem、getItem和removeItem等

# CSS

## 1.介绍一下CSS的盒子模型

（1）有两种， IE 盒子模型、标准 W3C 盒子模型；IE的content部分包含了 border 和 pading;

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

## 2. 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新增伪类有哪些

\* 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）

6.后代选择器（li a）

7.通配符选择器（ \* ）

8.属性选择器（a[rel = "external"]）

9.伪类选择器（a: hover, li: nth - child）

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

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

\* 优先级就近原则，同权重情况下样式定义最近者为准;

\* 载入样式以最后载入的定位为准;

优先级为:

!important > id > class > tag

important 比 内联优先级高

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。

## 3.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

|  |  |
| --- | --- |
| 1  2  3  4 | 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

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | .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%;  } |

## 4.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。象行内元素类型一样显示。

inline-block 象行内元素一样显示，但其内容象块类型元素一样显示。

list-item 象块类型元素一样显示，并添加样式列表标记。

2.

\*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）。

\* inherit 规定从父元素继承 position 属性的值。

## 5.CSS3有哪些新特性

CSS3实现圆角（border-radius:8px），阴影（box-shadow:10px），

对文字加特效（text-shadow、），线性渐变（gradient），旋转（transform）

transform:rotate(9deg) scale(0.85,0.90) translate(0px,-30px) skew(-9deg,0deg);//旋转,缩放,定位,倾斜

增加了更多的CSS选择器 多背景 rgba

## 6.一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。

## 7.为什么要初始化CSS样式

- 因为浏览器的兼容问题，不同浏览器对有些标签的默认值是不同的，如果没对CSS初始化往往会出现浏览器之间的页面显示差异。

- 当然，初始化样式会对SEO有一定的影响，但鱼和熊掌不可兼得，但力求影响最小的情况下初始化。

\*最简单的初始化方法就是： \* {padding: 0; margin: 0;} （不建议）

淘宝的样式初始化：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | body,  h1,  h2,  h3,  h4,  h5,  h6,  hr, p, blockquote, dl, dt, dd, ul, ol, li, pre,  form, fieldset, legend, button, input, textarea, th, td { margin:0;  padding:0;  }  body,  button, input, select, textarea { font:12px/1.5tahoma,  arial,  \5b8b\4f53;  }  h1,  h2,  h3,  h4,  h5,  h6{  font-size:100%;  }  address,  cite, dfn, em, var { font-style:normal;  }  code,  kbd, pre,  samp { font-family:couriernew,  courier,  monospace;  }  small{  font-size:12px; }  ul,  ol { list-style:none;  }  a  { text-decoration:none;  }  a:hover  { text-decoration:underline;  }  sup  { vertical-align:text-top;  }  Sub  { vertical-align:text-bottom;  }  legend  { color:#000;  }  fieldset,  img { border:0;  }  button,  input, select, textarea { font-size:100%;  }  table  { border-collapse:collapse;  border-spacing:0;  } |

## 8.absolute的containing block计算方式跟正常流有什么不同

block-level boxes  
一个 block-level element ('display' 属性值为 'block', 'list-item' 或是 ‘table’) 会生成一个 block-level box，这样的盒子会参与到 block-formatting context (一种布局的方式) 中。  
block formatting context  
在这种布局方式下，盒子们自所在的 containing block 顶部起一个接一个垂直排列，水平方向上撑满整个宽度 (除非内部的盒子自己内部建立了新的 BFC)。  
containing block  
一般来说，盒子本身就为其子孙建立了 containing block，用来计算内部盒子的位置、大小，而对内部的盒子，具体采用哪个 containing block 来计算，需要分情况来讨论：  
  
若此元素为 inline 元素，则 containing block 为能够包含这个元素生成的第一个和最后一个 inline box 的 padding box (除 margin, border 外的区域) 的最小矩形；  
否则则由这个祖先元素的 padding box 构成。  
根元素所在的 containing block 被称为 initial containing block，在我们常用的浏览器环境下，指的是原点与 canvas 重合，大小和 viewport 相同的矩形；  
对于 position 为 static 或 relative 的元素，其 containing block 为祖先元素中最近的 block container box 的 content box (除 margin, border, padding 外的区域)；  
对于 position:fixed 的元素，其 containing block 由 viewport 建立；  
对于 position:absolute 的元素，则是先找到其祖先元素中最近的 position 属性非 static 的元素，然后判断：  
如果都找不到，则为 initial containing block。

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

1、'display'、'position' 和 'float' 的相互关系；2、position跟display、overflow、float下的margin collapse。  
一、'display'、'position' 和 'float' 的相互关系  
首先我们先来看下这3个属性。  
display 属性规定元素应该生成的框的类型。 block 象块类型元素一样显示，none 缺省值。象行内元素类型一样显示， [inline-block](https://www.baidu.com/s?wd=inline-block&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHcvrjTdrH00T1d9Pj0YnvwbPvD1rHR4rAn40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHTdPHckP1mzP1bsnHDsPjD3r0) 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（display 还有很多其他值设置，读者自行查找）。  
 position 属性规定元素的定位类型。 absolute表示生成[绝对定位](https://www.baidu.com/s?wd=%E7%BB%9D%E5%AF%B9%E5%AE%9A%E4%BD%8D&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHcvrjTdrH00T1d9Pj0YnvwbPvD1rHR4rAn40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHTdPHckP1mzP1bsnHDsPjD3r0)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](https://www.baidu.com/s?wd=%E7%BB%9D%E5%AF%B9%E5%AE%9A%E4%BD%8D&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHcvrjTdrH00T1d9Pj0YnvwbPvD1rHR4rAn40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHTdPHckP1mzP1bsnHDsPjD3r0)的元素，相对于浏览器窗口进行定位；relative生成[相对定位](https://www.baidu.com/s?wd=%E7%9B%B8%E5%AF%B9%E5%AE%9A%E4%BD%8D&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHcvrjTdrH00T1d9Pj0YnvwbPvD1rHR4rAn40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHTdPHckP1mzP1bsnHDsPjD3r0)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](https://www.baidu.com/s?wd=z-index&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHcvrjTdrH00T1d9Pj0YnvwbPvD1rHR4rAn40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHTdPHckP1mzP1bsnHDsPjD3r0) 声明）。  
 Float也是是一种布局方式，它定义元素在哪个方向浮动。以往这个属性总应用于图像，使文本围绕在图像周围，不过在 CSS 中，任何元素都可以浮动。浮动元素会生成一个块级框，而不论它本身是何种元素。 在布局过程中也经常会使用它来达到左右并排布局的效果。  
那么这三种布局和框形成的关键特性( display )之间有什么关系呢，请看下面流程图：  
  
转换对应表：  
设定值 计算值  
inline-table table  
inline, run-in, table-row-group, table-column, table-column-group, table-header-group,   
table-footer-group, table-row, table-cell, table-caption, [inline-block](https://www.baidu.com/s?wd=inline-block&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHcvrjTdrH00T1d9Pj0YnvwbPvD1rHR4rAn40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHTdPHckP1mzP1bsnHDsPjD3r0) block  
其他 同设定值  
总的来说，可以把它看作是一个类似优先级的机制， "position:absolute" 和 "position:fixed" 优先级最高，有它存在的时候，浮动不起作用，'display' 的值也需要调整； 其次，元素的 'float' 特性的值不是 "none" 的时候或者它是根元素的时候，调整 'display' 的值； 最后，非根元素，并且非浮动元素，并且非[绝对定位](https://www.baidu.com/s?wd=%E7%BB%9D%E5%AF%B9%E5%AE%9A%E4%BD%8D&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHcvrjTdrH00T1d9Pj0YnvwbPvD1rHR4rAn40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHTdPHckP1mzP1bsnHDsPjD3r0)的元素，'display' 特性值同设置值。  
这从另一个侧面说明了一个问题：浮动或绝对定位的元素，只能是块元素或表格。  
1. 'display' 的值为 'none'  
如果 'display' 的值为 'none'，那么 'position' 和 'float' 不起作用。在这种情况下，元素不产生框。因此浮动和定位无效。  
2. 'position' 的值是 'absolute' 或 'fixed'  
否则，如果 'position' 的值是 'absolute' 或 'fixed'，框就是绝对定位的，'float' 计算后的值应该是 'none'，并且，'display' 会被按照上表设置。 框的位置将由 'top'，'right'，'bottom' 和 'left' 属性和该框的包含块确定。  
也就是说，当元素是绝对定位时，浮动失效，'display' 会被按规则重置。  
示例代码：  
<script type="text/javascript">function getStyle(obj, style) { var \_style = (style == "float") ? "styleFloat" : style; return document.defaultView ? document.defaultView.getComputedStyle(obj, null).getPropertyValue(style) : obj.currentStyle[\_style.replace(/-[a-z]/g, function() { return arguments[0].charAt(1).toUpperCase(); })];}window.onload = function() { document.getElementById("info").innerHTML = "float : " + getStyle(document.getElementById("test"), "float") + "<br/>display : " + getStyle(document.getElementById("test"), "display");}</script><div id="test" style="position:absolute; float:left;display:inline;"></div><div id="info"></div>  
上面代码中有一个既是绝对定位又是浮动的元素，以上代码可取出其 'display' 和 'float' 的计算值。  
IE 中，'float' 值和 'display' 的特性值未发生变化，还是 "float: left; display: inline"。  
其他浏览器中计算后的结果是："float: none; display: block"。  
3. 'float' 的值不是 "none"

## 10.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。）

## 11.css定义的权重

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

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | /\*权重为1\*/  div{  }  /\*权重为10\*/  .class1{  }  /\*权重为100\*/  #id1{  }  /\*权重为100+1=101\*/  #id1 div{  }  /\*权重为10+1=11\*/  .class1 div{  }  /\*权重为10+10+1=21\*/  .class1 .class2 div{  } |

如果权重相同，则最后定义的样式会起作用，但是应该避免这种情况出现

## 12.解释下浮动和它的工作原理？清除浮动的技巧

浮动元素脱离文档流，不占据空间。浮动元素碰到包含它的边框或者浮动元素的边框停留。

使用空标签清除浮动。

这种方法是在所有浮动标签后面添加一个空标签 定义css clear:both. 弊端就是增加了无意义标签。

使用overflow。

给包含浮动元素的父标签添加css属性 overflow:auto; zoom:1; zoom:1用于兼容IE6。

使用after伪对象清除浮动。

该方法只适用于非IE浏览器。具体写法可参照以下示例。使用中需注意以下几点。一、该方法中必须为需要清除浮动元素的伪对象中设置 height:0，否则该元素会比实际高出若干像素。

## 13.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。

## 14.CSS引入的方式有哪些? link和@import的区别是

内联 内嵌 外链 导入

区别 ：同时加载

前者无兼容性，后者CSS2.1以下浏览器不支持

Link 支持使用javascript改变样式，后者不可

(1) link属于HTML标签，而@import是CSS提供的;

(2) 页面被加载的时，link会同时被加载，而@import引用的CSS会等到页面被加载完再加载;

(3) import只在IE5以上才能识别，而link是HTML标签，无兼容问题;

(4) link方式的样式的权重 高于@import的权重.

## 15.CSS选择符有哪些? 哪些属性可以继承?优先级算法如何计算?内联和important哪个优先级高

标签选择符 类选择符 id选择符

继承不如指定 Id>class>标签选择

后者优先级高

## 16.前端页面有哪三层构成，分别是什么?作用是什么

结构层 Html 表示层 CSS 行为层 js

## 17.css的基本语句构成是

选择器{属性1:值1;属性2:值2;……}

18.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游览器从所有情况中分离出来。

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

css

.bb{

background-color:#f1ee18;/\*所有识别\*/

.background-color:#00deff\9; /\*IE6、7、8识别\*/

+background-color:#a200ff;/\*IE6、7识别\*/

\_background-color:#1e0bd1;/\*IE6识别\*/

}

\* IE下,可以使用获取常规属性的方法来获取自定义属性,

也可以使用getAttribute()获取自定义属性;

Firefox下,只能使用getAttribute()获取自定义属性.

解决方法:统一通过getAttribute()获取自定义属性.

\* IE下,even对象有x,y属性,但是没有pageX,pageY属性;

Firefox下,event对象有pageX,pageY属性,但是没有x,y属性.

\* （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 {}

## 19.写出几种IE6 BUG的解决方法

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dislpay: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

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over:hidden,zoom:0.08 line-height:1px）

## 20.标签上title与alt属性的区别是什么

Alt 当图片不显示是 用文字代表。

Title 为该属性提供信息

## 21.描述css reset的作用和用途

Reset重置浏览器的css默认属性 浏览器的品种不同，样式不同，然后重置，让他们统一。

## 22.解释css sprites，如何使用

Css 精灵 把一堆小的图片整合到一张大的图片上，减轻服务器对图片的请求数量

## 23.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

## 24.你如何对网站的文件和资源进行优化?期待的解决方案包括：

文件合并

文件最小化/文件压缩

使用CDN托管

缓存的使用

## 25.清除浮动的几种方式，各自的优缺点

1.使用空标签清除浮动 clear:both（理论上能清楚任何标签，，，增加无意义的标签）

2.使用overflow:auto（空标签元素清除浮动而不得不增加无意代码的弊端,,使用zoom:1用于兼容IE）

3.是用afert伪元素清除浮动(用于非IE浏览器)

SASS

## 26. 什么叫优雅降级和渐进增强

优雅降级：Web站点在所有新式浏览器中都能正常工作，如果用户使用的是老式浏览器，则代码会检查以确认它们是否能正常工作。由于IE独特的盒模型布局问题，针对不同版本的IE的hack实践过优雅降级了,为那些无法支持功能的浏览器增加候选方案，使之在旧式浏览器上以某种形式降级体验却不至于完全失效.

渐进增强：从被所有浏览器支持的基本功能开始，逐步地添加那些只有新式浏览器才支持的功能,向页面增加无害于基础浏览器的额外样式和功能的。当浏览器支持时，它们会自动地呈现出来并发挥作用。

## 27.display:none和visibility:hidden的区别

display:none 隐藏对应的元素，在文档布局中不再给它分配空间，它各边的元素会合拢，就当他从来不存在。

visibility:hidden 隐藏对应的元素，但是在文档布局中仍保留原来的空间。

## 28. position的absolute与fixed共同点与不同点

A：共同点：

1.改变行内元素的呈现方式，display被置为block；2.让元素脱离普通流，不占据空间；3.默认会覆盖到非定位元素上

B不同点：

absolute的”根元素“是可以设置的，而fixed的”根元素“固定为浏览器窗口。当你滚动网页，fixed元素与浏览器窗口之间的距离是不变的。

## 29. 解释下 CSS sprites，以及你要如何在页面或网站中使用它

CSS Sprites其实就是把网页中一些背景图片整合到一张图片文件中，再利用CSS的“background-image”，“background- repeat”，“background-position”的组合进行背景定位，background-position可以用数字能精确的定位出背景图片的位置。这样可以减少很多图片请求的开销，因为请求耗时比较长；请求虽然可以并发，但是也有限制，一般浏览器都是6个。对于未来而言，就不需要这样做了，因为有了`http2`。

# JavaScript

## 1.[JavaScript](http://lib.csdn.net/base/18)原型，原型链,有什么特点

\* 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。

　　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。

　　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)

## 2.eval是做什么的

它的功能是把对应的字符串解析成JS代码并运行；

应该避免使用eval，不安全，非常耗性能（2次，一次解析成js语句，一次执行）。

## 3.写一个通用的事件侦听器函数

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77 | // event(事件)工具集，来源：github.com/markyun   markyun.Event  = {       //  页面加载完成后       readyEvent  : function(fn) {           if  (fn==null) {               fn=document;           }           var  oldonload = window.onload;           if  (typeof window.onload != 'function')  {               window.onload  = fn;           }  else {               window.onload  = function() {                   oldonload();                   fn();               };           }       },       //  视能力分别使用dom0||dom2||IE方式  来绑定事件       //  参数： 操作的元素,事件名称 ,事件处理程序       addEvent  : function(element, type, handler) {           if  (element.addEventListener) {               //事件类型、需要执行的函数、是否捕捉               element.addEventListener(type,  handler, false);           }  else if (element.attachEvent) {               element.attachEvent('on' +  type, function() {                   handler.call(element);               });           }  else {               element['on' +  type] = handler;           }       },       //  移除事件       removeEvent  : function(element, type, handler) {           if  (element.removeEnentListener) {               element.removeEnentListener(type, handler, false);           } else if (element.datachEvent) {               element.detachEvent('on' + type, handler);           } else {               element['on' + type] = null;           }       },       // 阻止事件 (主要是事件冒泡，因为IE不支持事件捕获)       stopPropagation : function(ev) {           if (ev.stopPropagation) {               ev.stopPropagation();           } else {               ev.cancelBubble = true;           }       },       // 取消事件的默认行为       preventDefault : function(event) {           if (event.preventDefault) {               event.preventDefault();           } else {               event.returnValue = false;           }       },       // 获取事件目标       getTarget : function(event) {           return event.target || event.srcElement;       },       // 获取event对象的引用，取到事件的所有信息，确保随时能使用event；       getEvent : function(e) {           var ev = e || window.event;           if (!ev) {               var c = this.getEvent.caller;               while (c) {                   ev = c.arguments[0];                   if (ev && Event == ev.constructor) {                       break;                   }                   c = c.caller;               }           }           return ev;       }   }; |

## 4.[Node.js](http://lib.csdn.net/base/30)的适用场景

高并发、聊天、实时消息推送

## 5.介绍js的基本数据类型

number,string,boolean,object,undefined

## 6.Javascript如何实现继承

通过原型和构造器

## 7.["1", "2", "3"].map(parseInt) 答案是多少

[1, NaN, NaN] 因为 parseInt 需要两个参数 (val, radix) 但 map 传了 3 个 (element, index, array)

## 8.如何创建一个对象? （画出此对象的内存图）

|  |  |
| --- | --- |
| 1  2  3  4  5 | function Person(name, age) {     this.name = name;     this.age = age;     this.sing = function() { alert(this.name) }   } |

## 9.谈谈This对象的理解

this是js的一个关键字，随着函数使用场合不同，this的值会发生变化。

但是有一个总原则，那就是this指的是调用函数的那个对象。

this一般情况下：是全局对象Global。 作为方法调用，那么this就是指这个对象

## 10.事件是，IE与火狐的事件机制有什么区别，如何阻止冒泡

1. 我们在网页中的某个操作（有的操作对应多个事件）。例如：当我们点击一个按钮就会产生一个事件。是可以被 JavaScript 侦测到的行为。

2. 事件处理机制：IE是事件冒泡、火狐是 事件捕获；

3. ev.stopPropagation();

## 12.什么是闭包（closure），为什么要用它

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.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | 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 |

## 13."use strict";是什么意思，使用它的好处和坏处分别是什么

ECMAscript 5添加了第二种运行模式："严格模式"(strict mode)。顾名思义，这种模式使得Javascript在更严格的条件下运行。  
　　设立"严格模式"的目的，主要有以下几个：  
　　1. 消除Javascript语法的一些不合理、不严谨之处，减少一些怪异行为;  
　　2. 消除代码运行的一些不安全之处，保证代码运行的安全;  
　　3. 提高编译器效率，增加运行速度;  
　　4. 为未来新版本的Javascript做好铺垫。  
　　注：经过测试 IE6,7,8,9 均不支持严格模式。  
　　缺点：  
　　现在网站的 JS 都会进行压缩，一些文件用了严格模式，而另一些没有。这时这些本来是严格模式的文件，被 merge 后，这个串就到了文件的中间，不仅没有指示严格模式，反而在压缩后浪费了字节。

## 14.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');

}

## 15.new操作符具体干了什么呢

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

2、属性和方法被加入到 this 引用的对象中。

3、新创建的对象由 this 所引用，并且最后隐式的返回 this 。

var obj = {};

obj.\_\_proto\_\_ = Base.prototype;

Base.call(obj);

## 16.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

## 17.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'}

## 18.js延迟加载的方式有哪些

defer和async、动态创建DOM方式（用得最多）、按需异步载入js

## 19.ajax 是什么? 同步和异步的区别? 如何解决跨域问题

通过异步模式，提升了用户体验

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

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。

Ajax的最大的特点是什么。

Ajax可以实现动态不刷新（局部刷新）

readyState属性 状态 有5个可取值： 0=未初始化 ，1=正在加载 2=以加载，3=交互中，4=完成

ajax的缺点

ajax不支持浏览器back按钮。

安全问题 AJAX暴露了与服务器交互的细节。

对搜索引擎的支持比较弱。

破坏了程序的异常机制。

不容易调试。

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

## 20. 对Node的优点和缺点提出了自己的看法

\*（优点）因为Node是基于事件驱动和无阻塞的，所以非常适合处理并发请求，

因此构建在Node上的代理服务器相比其他技术实现（如Ruby）的服务器表现要好得多。

此外，与Node代理服务器交互的客户端代码是由javascript语言编写的，

因此客户端和服务器端都用同一种语言编写，这是非常美妙的事情。

\*（缺点）Node是一个相对新的开源项目，所以不太稳定，它总是一直在变，

而且缺少足够多的第三方库支持。看起来，就像是Ruby/Rails当年的样子。

## 21. 说出这道题的答案

(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”。

## 22.模块化怎么做

[立即执行函数](http://benalman.com/news/2010/11/immediately-invoked-function-expression/),不暴露私有成员

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | var module1 = (function(){  　　　　var \_count = 0;  　　　　var m1 = function(){  　　　　　　//...  　　　　};  　　　　var m2 = function(){  　　　　　　//...  　　　　};  　　　　return {  　　　　　　m1 : m1,  　　　　　　m2 : m2  　　　　};  　　})(); |

## 24.异步加载的方式有哪些

(1) defer，只支持IE

(2) async：

(3)创建script，插入到DOM中，加载完毕后callBack

## 25.documen.write和 innerHTML的区别

document.write只能重绘整个页面

innerHTML可以重绘页面的一部分

## 26..call() 和 .apply() 的区别

例子中用 add 来替换 sub，add.call(sub,3,1) == add(3,1) ，所以运行结果为：alert(4);

注意：js 中的函数其实是对象，函数名是对 Function 对象的引用。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | function add(a,b)  {      alert(a+b);  }    function sub(a,b)  {      alert(a-b);  }    add.call(sub,3,1); |

## 27.[jQuery](http://lib.csdn.net/base/22)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

## 28.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：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | $.fn.stringifyArray = function(array) {        return JSON.stringify(array)    }      $.fn.parseArray = function(array) {        return JSON.parse(array)    } |

然后调用：

|  |  |
| --- | --- |
| 1 | $("").stringifyArray(array) |

## 29.针对 jQuery 的优化方法

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

\*频繁操作的DOM，先缓存起来再操作。用Jquery的链式调用更好。

比如：var str=$("a").attr("href");

\*for (var i = size; i < arr.length; i++) {}

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

for (var i = size, length = arr.length; i < length; i++) {}

## 30.JavaScript中的作用域与变量声明提升

如何编写高性能的Javascript？

那些操作会造成内存泄漏？

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）

## 31. javascript的typeof返回哪些数据类型

Object number function boolean underfind

## 32.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（== – ===）

## 33.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

## 34.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

## 35.IE和DOM事件流的区别

1.参数不一样

2.事件加不加on

3.this指向问题

## 36.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

## 37.ajax请求的时候get 和post方式的区别

一个在url后面 一个放在虚拟载体里面

有大小限制

安全问题

应用不同 一个是论坛等只需要请求的，一个是类似修改密码的

## 38.call和apply的区别

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

Object.apply(this,arguments)

## 39.ajax请求时，如何解释json数据

使用eval parse 鉴于安全性考虑 使用parse更靠谱

## 40.写一个获取非行间样式的函数

function getStyle(obj,attr,value)

{

if(!value)

{

if(obj.currentStyle)

{

return obj.currentStyle(attr)

{

obj.getComputedStyle(attr,false)

}

}

Else

{

obj.style[attr]=value

}

}

## 41.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！

http://www.webasily.com/?p=78 例子可见此链接

## 42.如何阻止事件冒泡和默认事件

canceBubble return false

## 43.添加 删除 替换 插入到某个接点的方法

obj.appendChidl()

obj.innersetBefore

obj.replaceChild

obj.removeChild

## 44.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

## 45.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

## 49.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 $().ready(function)

## 50.”==”和“===”的不同

前者会自动转换类型

后者不会

## 51.javascript的同源策略

一段脚本只能读取来自于同一来源的窗口和文档的属性，这里的同一来源指的是主机名、协议和端口号的组合

## 52.编写一个数组去重的方法

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

}

if(!result[arr \[i\])  
{  
newArr.push(arr \[i\)  
result[arr \[i\]=1  
}

## 53. 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。

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）

## 54. 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)

## 55. Jquery与jQuery UI 有什么区别

\*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

1. 在一个框架的属性面板中，不能设置下面哪一项。（  C ）
2. 源文件
3. 边框颜色

C．边框宽度

D．滚动条

1. CSS样式表根据所在网页的位置，可分为?(B )

A．行内样式表、内嵌样式表、混合样式表

B．行内样式表、内嵌样式表、外部样式表

C．外部样式表、内嵌样式表、导入样式表

D．外部样式表、混合样式表、导入样式表

1. 对于标签，其中\*代表GET或（ C ）

A. SET

B. PUT

C. POST

D. INPUT

4. HTML代码表示?(D )

A. 创建表格

B. 创建一个滚动菜单

C. 设置每个表单项的内容

D. 创建一个下拉菜单

5. 在CSS中下面哪种方法表示超链接文字在鼠标经过时，超链接文字无下划线?(B )

A. A:link{TEXT-DECORATION: underline }

B. A:hover {TEXT-DECORATION: none}

C. A:active {TEXT-DECORATION: blink }

D. A:visited {TEXT-DECORATION: overline }

6. 下面代码片段，说法正确的是:(B )

.DIV1 { position:absolute;

line-height:22px;

height:58px;

background-color: #FF0000; }

A. Line-height:22px;修饰文本字体大小

B. position:absolute;表示绝对定位，被定位的元素位置固定

C. height:58px; 表示被修饰的元素距离别的元素的距离

D. background-color: #FF0000; 表示被修饰的元素的背景图像

7. 在Dreamweaver 中, 在设置各个框架属性时, 参 数Scroll 是用来设置（ B ）属性的。

A. 是否进行颜色设置

B. 是否出现滚动条

C. 是否设置边框宽度

D. 是否使用默认边框宽度

8. 以下链接到电子邮件的正确格式是(B )

A. 邮箱

B. 邮箱

C. 邮箱

D. 邮箱

9. 若将Dreamweaver中２个横向相邻的单元格合并，则两单元格中文字会（ A ）

A．文字合并

B．左单元格文字丢失

C．右单元格文字丢失

D．系统出错

10. ID为left的DIV标签，用CSS设置DIV的左边为红色实线，下面设置正确的是 (C )

A．style=”border-top: #ff0000 1 solid;”

B．style=”border-left: 1, #ff0000 ,solid;”

C．style=”border-left: 1 #ff0000 solid;”

D．style=”border-right: 1, #ff0000, dashed;”

11. 的意思是?(A )

A. 图像相对于周围的文本左对齐

B. 图像相对于周围的文本右对齐

C. 图像相对于周围的文本底部对齐

D. 图像相对于周围的文本顶部对齐

12. 一个有 3 个框架的 Web 页实际上有几个独立 的HTML 文件?（C ）

A. 2

B. 3

C. 4

D. 5

13. 下列标签可以不成对出现的是（ B）

A．〈HTML〉〈/HTML〉

B．〈P〉 〈/P〉

C．〈TITLE〉〈/TITLE〉

D．〈BODY〉〈/BODY〉

14. 下列那个不是CSS选择符(A )

A．p

B．.td

C．#box

D．shop

15. 在CSS的三种选择符种，哪个优先级最高?( C )

A．标签选择符

B．类选择符号

C．ID选择符

16. 引入样式表的方式有几种?

引入样式表的方式有两种：方式一通过link引入，如<link rel=“stylesheet”

Type=“text/css” href=“css.css”> 方法二 通过import导入 如import css.css

17. 关于Document对象，下列叙述正确的是（AC ）

ADocument对象表示浏览器中显示的HTML文档

B)Document对象的常用属性bgcolor用来设置文档的背景图片

C)Document对象的常用方法getElementById（）用于访问唯一的一个元素

D)Document对象的常用方法getElementByNames（）

18. 在javascript 中，以下哪个变量名是非法的（B ）

A)Name

B)9name

C)Name\_a

D)Name9

19. Javascript可以在以下什么地方执行?(B )

A)服务器端的IE浏览器

B)客户端的IE浏览器

C在服务器端的tomcat容器里

D)在客户端的tomcat容器里

20. 在javascript中，( B )变量在函数外声明，并可从脚本的任意位置访问

A)局部

B)全局

C)typeOf

D)New

21. 关于javascript说法正确的是（B ）。

A)它是面向对象的

B)它是基于对象的

C)它是面向过程的

D)以上说法都不正确

22. 以下哪条语句会产生运行错误(A )

A、var obj = ()

B、var obj = []

C、var obj = {}

D、var obj = //

23. 在javascript中，下拉列表框的（B ）方法可从列表中删除一个元素

A)Add( )

B)Remove( )

C)Focus( )

D)blur（）

24. 请选择结果为真的表达式(C )

A)null instanceof Object

B)null === undefined

C)null == undefined

D)NaN == NaN

25. 在javascript中，String 对象的方法不包括（D ）。

A)charAt()

B)substring()

C)toUpperCase()

D)Length

26. 下拉菜单中，用户更改表单元素Select中的值时，就会调用（A ）事件处理程序。( )

A)onChange

B)onFocus

C)onMouseOver

D)onClick

27. ( C )事件处理程序可用于在用户单击按钮时执行函数

A)onSubmit

B)onChange

C)onClick

D)onBlur

# 初级Javascript：

## 1.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)

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：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6 | if(typeof Array.isArray==="undefined")  {    Array.isArray = function(arg){          return Object.prototype.toString.call(arg)==="[object Array]"      };  } |

## 2.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)

JavaScript



|  |  |
| --- | --- |
| 1 | document.getElementById(“ID”).value |

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

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | 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]);  　　}  } |

## 4.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)

JavaScript



|  |  |
| --- | --- |
| 1  2  3 | var dom = document.getElementById(“ID”);  dom.innerHTML = “xxxx”  dom.style.color = “#000” |

## 5.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

## 6.什么是Ajax和JSON，它们的优缺点

Ajax是异步JavaScript和XML，用于在Web页面中实现异步数据交互。

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）

## 7.看下列代码输出为何，解释原因

JavaScript



|  |  |
| --- | --- |
| 1  2  3 | var a;  alert(typeof a); // undefined  alert(b); // 报错 |

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。注意未申明的变量和声明了未赋值的是不一样的。

## 8.看下列代码,输出什么，解释原因

JavaScript



|  |  |
| --- | --- |
| 1  2 | 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”。

## 9.看下列代码,输出什么，解释原因

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | var undefined;  undefined == null; // true  1 == true;   // true  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？

JavaScript



|  |  |
| --- | --- |
| 1  2  3 | 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String。

## 10.看代码给答案

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5 | var a = new Object();  a.value = 1;  b = a;  b.value = 2;  alert(a.value); |

答案：2（考察引用数据类型细节）

## 11.已知数组var stringArray = [“This”, “is”, “Baidu”, “Campus”]，Alert出”This is Baidu Campus”。

答案：alert(stringArray.join(“”))

## 12.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”。

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | 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;  } |

(考察基础API)

## 13.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]

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | 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;  } |

## 12.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

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | 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); |

## 13.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’);

## 14.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

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | function escapeHtml(str) {  return str.replace(/[<>”&]/g, function(match) {      switch (match) {                     case “<”:                        return “&lt;”;                     case “>”:                        return “&gt;”;                     case “&”:                        return “&amp;”;                     case “\””:                        return “&quot;”;        }    });  } |

## 15.foo = foo||bar ，这行代码是什么意思？为什么要这样写

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。

## 16.看下列代码，将会输出什么(变量声明提升)

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6 | 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：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | 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。

## 17.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。

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | var iArray = [];  funtion 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(); |

## 18.把两个数组合并，并删除第二个元素。

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4 | var array1 = ['a','b','c'];  var bArray = ['d','e','f'];  var cArray = array1.concat(bArray);  cArray.splice(1,1); |

## 19.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

## 20.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：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | function serilizeUrl(url) {      var result = {};      url = url.split("?")[1];      var map = url.split("&");      for(var i = 0, len = map.length; i < len; i++) {          result[map[i].split("=")[0]] = map[i].split("=")[1];      }      return result;  } |

## 21.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。

邮箱的正则匹配：

JavaScript



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

## 22.看下面代码，给出输出结果

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5 | for(var i=1;i<=3;i++){    setTimeout(function(){        console.log(i);    },0);  }; |

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？

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | 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 |

## 23.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：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | 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" |

## 24.Javascript中callee和caller的作用

答案：

caller是返回一个对函数的引用，该函数调用了当前函数；

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）

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | 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];          }     }  } |

# 中级Javascript：

## 1.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

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34 | // 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;          }      } |

## 2.如何消除一个数组里面重复的元素

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | 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 |

## 3.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

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | 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）

答案：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | 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(); |

## 4.下面这个ul，如何点击每一列的时候alert其index?（闭包）

XHTML



|  |  |
| --- | --- |
| 1  2  3  4  5 | <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> |

答案：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | // 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);  } |

## 5.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}\*/

答案：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57 | 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;            } |

## 6.请评价以下代码并给出改进意见。

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | 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);        });     }  } |

评价：

* 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：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | 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;  　　}  } |

## 7.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’**

JavaScript



|  |  |
| --- | --- |
| 1  2  3 | 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。

## 8.定义一个log方法，让它可以代理console.log的方法

可行的方法一：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5 | 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：

JavaScript



|  |  |
| --- | --- |
| 1  2  3 | function log(){      console.log.apply(console, arguments);  }; |

到此，追问apply和call方法的异同。

答案：

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]) 。

## 9.在Javascript中什么是伪数组？如何将伪数组转化为标准数组

答案：

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。

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：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6 | 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);      }; |

## 10.对作用域上下文和this的理解，看下列代码：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | var User = {    count: 1,      getCount: function() {      return this.count;    }  };    console.log(User.getCount());  // what?    var func = User.getCount;  console.log(func());  // what? |

问两处console输出什么？为什么？

答案是1和undefined。

func是在winodw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Uesr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：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | 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()); |

## 11.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。

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

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42 | /\*  \* 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 === 'onreadystatechange' && 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：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | 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              }          });      }  }; |

## 12.（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

## 13.说出以下函数的作用是？空白区域应该填写什么

## JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | //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

# 其他问题

你遇到过比较难的技术问题是？你是如何解决的？

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）

\* Sublime Text 、Eclipse、Notepad、Firebug、HttpWatch、Yslow。

\* 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）

页面重构怎么操作？

列举IE 与其他浏览器不一样的特性？

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）

WEB应用从服务器主动推送Data到客户端有那些方式？

html5 websoket

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)

你有哪些性能优化的方法？

（看雅虎14条性能优化原则）。

（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。

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 1、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

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）

查找浏览器缓存

DNS解析、查找该域名对应的IP地址、重定向（301）、发出第二个GET请求

进行HTTP协议会话

客户端发送报头(请求报头)

服务器回馈报头(响应报头)

html文档开始下载

文档树建立，根据标记请求所需指定MIME类型的文件

文件显示

[

浏览器这边做的工作大致分为以下几步：

加载：根据请求的URL进行域名解析，向服务器发起请求，接收文件（HTML、JS、CSS、图象等）。

解析：对加载到的资源（HTML、JS、CSS等）进行语法解析，建议相应的内部数据结构（比如HTML的DOM树，JS的（对象）属性表，CSS的样式规则等等）

}

除了前端以外还了解什么其它技术么？你最最厉害的技能是什么？

你常用的开发工具是什么，为什么？

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；

做好的页面结构，页面重构和用户体验；

处理hack，兼容、写出优美的代码格式；

针对服务器的优化、拥抱最新前端技术。

加班的看法？

加班就像借钱，原则应当是------救急不救穷

平时如何管理你的项目？

先期团队必须确定好全局样式（globe.css），编码模式(utf-8) 等

编写习惯必须一致（例如都是采用继承式的写法，单样式都写成一行）；标注样式编写人，各模块都及时标注（标注关键样式调用的地方）；页面进行标注（例如 页面 模块 开始和结束）；CSS跟HTML 分文件夹并行存放，命名都得统一（例如style.css）JS 分文件夹存放 命民以该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

如何设计突发大规模并发[架构](http://lib.csdn.net/base/16)？

说说最近最流行的一些东西吧？常去哪些网站？

Node.js、Mongodb、npm、MVVM、MEAN、three.js

移动端（[Android](http://lib.csdn.net/base/15) [iOS](http://lib.csdn.net/base/1)）怎么做好用户体验?

清晰的视觉纵线、信息的分组、极致的减法、

利用选择代替输入、标签及文字的排布方式、

依靠明文确认密码、合理的键盘利用、

你在现在的团队处于什么样的角色，起到了什么明显的作用？

你认为怎样才是全端工程师（Full Stack developer）？

介绍一个你最得意的作品吧？

你的优点是什么？缺点是什么？

如何管理前端团队?

最近在学什么？能谈谈你未来3，5年给自己的规划吗？

想问公司的问题？

问公司问题：

目前关注哪些最新的Web前端技术（未来的发展方向）？

前端团队如何工作的（实现一个产品的流程）？

公司的薪资结构是什么样子的？