前端面试题及答案

* + 1. 前言

小伙伴们，这里给大家总结了一些优质的前端面试题（多数源于网络），大家看了之后要用心钻研其中的原理，重要知识需要系统学习，透彻理解，形成自己的知识链。万不可投机取巧，只求面试过关是不对的喲~！

* + 1. 前端开发面试知识点大纲

前端开发面试知识点大纲：

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、CSS3、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

作为一名前端工程师，无论工作年头长短都应该必须掌握的知识点：

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。

* + 1. 试题部分
       1. 如何创建一个Javascript Object?

可以有两种方式创建一个Javascript Object, 代码如下： 第一种方法：var obj = new Object(); 第二种方法：var ob = {};

* + - 1. 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

* + - 1. 什么是语义化的HTML?

直观的认识标签 对于搜索引擎的抓取有好处

* + - 1. 清除浮动的几种方式，各自的优缺点？

1、父级div定义height

• 优点：简单，代码少，容易掌握

• 缺点：只适合高度固定的布局，要给出精确的高度，如果高度和父级div不一样时，会产生问题

2、结尾处加空div标签clear:both

• 优点：简单，代码少，浏览器支持好，不容易出现怪问题

• 缺点：不少初学者不理解原理；如果页面浮动布局多，就要增加很多空div，让人感觉很不爽

3、父级div定义伪类:after和zoom

• 优点：浏览器支持好，不容易出现怪问题（目前：大型网站都有使用，如：腾迅，网易，新浪等等）

• 缺点：代码多，不少初学者不理解原理，要两句代码结合使用，才能让主流浏览器都支持

4、父级div定义overflow:hidden

• 优点：简单，代码少，浏览器支持好

• 缺点：不能和position配合使用，因为超出的尺寸的会被隐藏

5、父级div定义overflow:auto

• 优点：简单，代码少，浏览器支持好

• 缺点：内部宽高超过父级div时，会出现滚动条。

6、父级div也一起浮动

• 优点：没有优点

• 缺点：会产生新的浮动问题。

7、父级div定义display:table

• 优点：没有优点

• 缺点：会产生新的未知问题

8、结尾处加br标签clear:both

• 原理：父级div定义zoom:1来解决IE浮动问题，结尾处加br标签clear:both

• 建议：不推荐使用，只作了解

* + - 1. javascript的typeof返回哪些数据类型

Object number function boolean underfind

* + - 1. 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（== – ===）

* + - 1. 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

* + - 1. 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

* + - 1. 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

* + - 1. ajax请求的时候get 和post方式的区别

1、 get是把参数数据队列加到提交表单的ACTION属性所指的URL中，值和表单内各个字段一一对应，在URL中可以看到。post是通过HTTP post机制，将表单内各个字段与其内容放置在HTML HEADER内一起传送到ACTION属性所指的URL地址。用户看不到这个过程。

2、 对于get方式，服务器端用Request.QueryString获取变量的值，对于post方式，服务器端用Request.Form获取提交的数据。两种方式的参数都可以用Request来获得。

3、get传送的数据量较小，不能大于2KB。post传送的数据量较大，一般被默认为不受限制。但理论上，因服务器的不同而异.

4、get安全性非常低，post安全性较高。

5、 <form method="get" action="a.asp?b=b">跟<form method="get" action="a.asp">是一样的，也就是说，method为get时action页面后边带的参数列表会被忽视；而<form method="post" action="a.asp?b=b">跟<form method="post" action="a.asp">是不一样的。

* + - 1. call和apply的区别

apply和call两者在作用上是相同的，一般用于修改this指向，调用一个对象的一个方法，以另一个对象替换当前对象。

简单示例：

function add(a,b)  
{  
alert(a+b);  
}  
function reduce(a,b)  
{  
alert(a-b);  
}  
add.call(reduce,1,3) //将add方法运用到reduce,结果为4

不同点只有一个，call有多个参数， apply只有2个参数，它将call的参数（arg1,arg2…）放在一个数组中作为apply的第二参数

Object.call(this, arg1,arg2 arg3)  
Object.apply(this, [arg1,arg2])

* + - 1. 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，它的最大用处有两个，一个是前面提到的可以读取函数内部的变量，另一个就是让这些变量的值始终保持在内存中。

简单示例：

function f1(){

　　　　n=999;

　　　　function f2(){  
　　　　　　alert(n);  
　　　　}

　　　　return f2;

　　}

　　var result=f1();

　　result(); // 999

* + - 1. 如何阻止事件冒泡和默认事件

canceBubble

return false

* + - 1. 添加 删除 替换 插入到某个接点的方法

obj.appendChidl(); obj.innersetBefore(); obj.replaceChild(); obj.removeChild();

* + - 1. 解释jsonp的原理，以及为什么不是真正的ajax

jsonp主要是通过script可以链接远程url来实现跨域请求的。如：

<script src="http://jsonp.js?callback=xxx"></script>

callback定义了一个函数名，而远程服务端通过调用指定的函数并传入参数来实现传递参数。

JSONP的缺点则是：它只支持GET请求而不支持POST等其它类型的HTTP请求；它只支持跨域HTTP请求这种情况，不能解决不同域的两个页面之间如何进行JavaScript调用的问题。  
jsonp依靠动态创建script标签，利用callback回执返回的值, 根本没有使用ajax XmlHttpRequest，所以不是真正的ajax

* + - 1. 使用最简洁的方法，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  
}

* + - 1. <!DOCTYPE html>中DOCTYPE有什么作用，去掉会有什么影响？

文档模式主要有以下两个作用：

1、告诉浏览器使用什么样的html或xhtml规范来解析html文档

2、对浏览器的渲染模式产生影响；不同的渲染模式会影响到浏览器对于 CSS 代码甚至 JavaScript 脚本的解析，所以Doctype是非常关键的，尤其是在 IE 系列浏览器中，由DOCTYPE 所决定的 HTML 页面的渲染模式至关重要。

要建立符合标准的网页，DOCTYPE声明是必不可少的关键组成部分；除非你的XHTML确定了一个正确的DOCTYPE，否则你的标识和CSS都不会生效。

* + - 1. HTML行内元素有那些，块级元素有那些？画出CSS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 – 举中对齐块  
\* dir – 目录列表  
\* div – 常用块级容易，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）。

* + - 1. 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哪个优先级更高？

包含选择符（又称后代选择符） E1 E2{}

子对象选择符 E1>E2{}

ID选择符 #ID{}

类选择符 E.className{}

选择符分组 E1,E2,E3{}

不可继承的：display、margin、border、padding、background、height、min-height、max-height、width、min-width、max-width、overflow、position、left、right、top、bottom、z-index、float、clear、table-layout、vertical-align、page-break-after、page-bread-before和unicode-bidi。

所有元素可继承：visibility和cursor。

内联元素可继承：letter-spacing、word-spacing、white-space、line-height、color、font、font-family、font-size、font-style、font-variant、font-weight、text-decoration、text-transform、direction。

终端块状元素可继承：text-indent和text-align。

列表元素可继承：list-style、list-style-type、list-style-position、list-style-image。

优先级算法计算的问题：

在多个选择符应用于同一个元素上那么Specificity值高的最终获得优先级。

选择符Specificity值列表：规则：

1. 行内样式优先级Specificity值为1,0,0,0，高于外部定义。

如：<div style=”color: red”>sjweb</div>

外部定义指经由<link>或<style>标签定义的规则；

2.!important声明的Specificity值最高；

3.Specificity值一样的情况下，按CSS代码中出现的顺序决定，后者CSS样式居上；

4.由继续而得到的样式没有specificity的计算，它低于一切其他规则(比如全局选择符\*定义的规则)。

算法：当遇到多个选择符同时出现时候

按选择符得到的Specificity值逐位相加，

{数位之间没有进制 比如说： 0,0,0,5 + 0,0,0,5 =0,0,0,10 而不是 0,0, 1, 0}

就得到最终计算得的specificity，

然后在比较取舍时按照从左到右的顺序逐位比较。

!Important > 行内样式 > id > class > 标签

* + - 1. HTML5储存类型有哪些？

HTML 5 支持本地存储，在之前版本中是通过 Cookie 实现的。HTML5 本地存储速度快而且安全。

有两种不同的对象可用来存储数据：

localStorage 适用于长期存储数据，浏览器关闭后数据不丢失

sessionStorage 存储的数据在浏览器关闭后自动删除

* + - 1. HTTP状态码有哪些，分别代表什么意思？

|  |  |  |
| --- | --- | --- |
| 状态代码 | 状态信息 | 含义 |
| 100 | Continue | 初始的请求已经接受，客户应当继续发送请求的其余部分。（HTTP 1.1新） |
| 101 | Switching Protocols | 服务器将遵从客户的请求转换到另外一种协议（HTTP 1.1新） |
| 200 | OK | 一切正常，对GET和POST请求的应答文档跟在后面。 |
| 201 | Created | 服务器已经创建了文档，Location头给出了它的URL。 |
| 202 | Accepted | 已经接受请求，但处理尚未完成。 |
| 203 | Non-Authoritative Information | 文档已经正常地返回，但一些应答头可能不正确，因为使用的是文档的拷贝（HTTP 1.1新）。 |
| 204 | No Content | 没有新文档，浏览器应该继续显示原来的文档。如果用户定期地刷新页面，而Servlet可以确定用户文档足够新，这个状态代码是很有用的。 |
| 205 | Reset Content | 没有新的内容，但浏览器应该重置它所显示的内容。用来强制浏览器清除表单输入内容（HTTP 1.1新）。 |
| 206 | Partial Content | 客户发送了一个带有Range头的GET请求，服务器完成了它（HTTP 1.1新）。 |
| 300 | Multiple Choices | 客户请求的文档可以在多个位置找到，这些位置已经在返回的文档内列出。如果服务器要提出优先选择，则应该在Location应答头指明。 |
| 301 | Moved Permanently | 客户请求的文档在其他地方，新的URL在Location头中给出，浏览器应该自动地访问新的URL。 |
| 302 | Found | 类似于301，但新的URL应该被视为临时性的替代，而不是永久性的。注意，在HTTP1.0中对应的状态信息是“Moved Temporatily”。出现该状态代码时，浏览器能够自动访问新的URL，因此它是一个很有用的状态代码。注意这个状态代码有时候可以和301替换使用。例如，如果浏览器错误地请求http://host/~user（缺少了后面的斜杠），有的服务器 返回301，有的则返回302。严格地说，我们只能假定只有当原来的请求是GET时浏览器才会自动重定向。请参见307。 |
| 303 | See Other | 类似于301/302，不同之处在于，如果原来的请求是POST，Location头指定的重定向目标文档应该通过GET提取（HTTP 1.1新）。 |
| 304 | Not Modified | 客户端有缓冲的文档并发出了一个条件性的请求（一般是提供If-Modified-Since头表示客户只想比指定日期更新的文档）。服务器告 诉客户，原来缓冲的文档还可以继续使用。 |
| 305 | Use Proxy | 客户请求的文档应该通过Location头所指明的代理服务器提取（HTTP 1.1新）。 |
| 307 | Temporary Redirect | 和302 （Found）相同。许多浏览器会错误地响应302应答进行重定向，即使原来的请求是POST，即使它实际上只能在POST请求的应答是303时才能重定 向。由于这个原因，HTTP 1.1新增了307，以便更加清除地区分几个状态代码：当出现303应答时，浏览器可以跟随重定向的GET和POST请求；如果是307应答，则浏览器只 能跟随对GET请求的重定向。（HTTP 1.1新） |
| 400 | Bad Request | 请求出现语法错误。 |
| 401 | Unauthorized | 客户试图未经授权访问受密码保护的页面。应答中会包含一个WWW-Authenticate头，浏览器据此显示用户名字/密码对话框，然后在填 写合适的Authorization头后再次发出请求。 |
| 403 | Forbidden | 资源不可用。服务器理解客户的请求，但拒绝处理它。通常由于服务器上文件或目录的权限设置导致。 |
| 404 | Not Found | 无法找到指定位置的资源。这也是一个常用的应答。 |
| 405 | Method Not Allowed | 请求方法（GET、POST、HEAD、DELETE、PUT、TRACE等）对指定的资源不适用。（HTTP 1.1新） |
| 406 | Not Acceptable | 指定的资源已经找到，但它的MIME类型和客户在Accpet头中所指定的不兼容（HTTP 1.1新）。 |
| 407 | Proxy Authentication Required | 类似于401，表示客户必须先经过代理服务器的授权。（HTTP 1.1新） |
| 408 | Request Timeout | 在服务器许可的等待时间内，客户一直没有发出任何请求。客户可以在以后重复同一请求。（HTTP 1.1新） |
| 409 | Conflict | 通常和PUT请求有关。由于请求和资源的当前状态相冲突，因此请求不能成功。（HTTP 1.1新） |
| 410 | Gone | 所请求的文档已经不再可用，而且服务器不知道应该重定向到哪一个地址。它和404的不同在于，返回407表示文档永久地离开了指定的位置，而 404表示由于未知的原因文档不可用。（HTTP 1.1新） |
| 411 | Length Required | 服务器不能处理请求，除非客户发送一个Content-Length头。（HTTP 1.1新） |
| 412 | Precondition Failed | 请求头中指定的一些前提条件失败（HTTP 1.1新）。 |
| 413 | Request Entity Too Large | 目标文档的大小超过服务器当前愿意处理的大小。如果服务器认为自己能够稍后再处理该请求，则应该提供一个Retry-After头（HTTP 1.1新）。 |
| 414 | Request URI Too Long | URI太长（HTTP 1.1新）。 |
| 416 | Requested Range Not Satisfiable | 服务器不能满足客户在请求中指定的Range头。（HTTP 1.1新） |
| 500 | Internal Server Error | 服务器遇到了意料不到的情况，不能完成客户的请求。 |
| 501 | Not Implemented | 服务器不支持实现请求所需要的功能。例如，客户发出了一个服务器不支持的PUT请求。 |
| 502 | Bad Gateway | 服务器作为网关或者代理时，为了完成请求访问下一个服务器，但该服务器返回了非法的应答。 |
| 503 | Service Unavailable | 服务器由于维护或者负载过重未能应答。例如，Servlet可能在数据库连接池已满的情况下返回503。服务器返回503时可以提供一个 Retry-After头。 |
| 504 | Gateway Timeout | 由作为代理或网关的服务器使用，表示不能及时地从远程服务器获得应答。（HTTP 1.1新） |
| 505 | HTTP Version Not Supported | 服务器不支持请求中所指明的HTTP版本。（HTTP 1.1新） |

* + - 1. CSS3新增伪类选择器有哪些？

:first-of-type p:first-of-type 选择属于其父元素的首个 <p> 元素的每个 <p> 元素。

:last-of-type p:last-of-type 选择属于其父元素的最后 <p> 元素的每个 <p> 元素。

:only-of-type p:only-of-type 选择属于其父元素唯一的 <p> 元素的每个 <p> 元素。

:only-child p:only-child 选择属于其父元素的唯一子元素的每个 <p> 元素。

:nth-child(n) 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。

:nth-last-child(n) p:nth-last-child(2) 同上，从最后一个子元素开始计数。

:nth-of-type(n) p:nth-of-type(2) 选择属于其父元素第二个 <p> 元素的每个 <p> 元素。

:nth-last-of-type(n) p:nth-last-of-type(2) 同上，但是从最后一个子元素开始计数。

:last-child p:last-child 选择属于其父元素最后一个子元素每个 <p> 元素。

:root :root 选择文档的根元素。

:empty p:empty 选择没有子元素的每个 <p> 元素（包括文本节点）。

:target #news:target 选择当前活动的 #news 元素。

:enabled input:enabled 选择每个启用的 <input> 元素。

:disabled input:disabled 选择每个禁用的 <input> 元素

:checked input:checked 选择每个被选中的 <input> 元素。

:not(selector) :not(p) 选择非 <p> 元素的每个元素。

::selection ::selection 选择被用户选取的元素部分。

* + - 1. 用CSS3编写：div旋转90度，0.5秒后变圆，0.5秒后放大2倍。

div{

width:50px;

height:50px;

border:1px solid red;

border-top-width: 3px;

animation: myDiv 2s;

}

@keyframes myDiv {

0%{

transform: rotate(90deg);

}

50%{

border-radius: 50%;

transform: rotate(90deg) scale(1);

}

100%{

transform: rotate(90deg) scale(2);

}

}

* + - 1. 事件绑定和普通事件有什么区别？

事件绑定就是针对dom元素的事件，绑定在dom元素上  
普通事件即为非针对dom元素的事件

* + - 1. 什么是事件委托，如何实现事件委托，使用事件委托有什么好处？

事件委托概念：通俗的讲，事件就是onclick，onmouseover，onmouseout，等就是事件，委托呢，就是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。

使用事件委托技术能让你避免对特定的每个节点添加事件监听器；相反，事件监听器是被添加到它们的父元素上。事件监听器会分析从子元素冒泡上来的事件，找到是哪个子元素的事件。

好处：1、提高性能。2、可以再已有父级上为后续通过js往html中添加的DOM添加事件

实现方式：

假定我们有一个UL元素，它有几个子元素：

<ul id="parent-list">

<li id="post-1">Item 1</li>

<li id="post-2">Item 2</li>

<li id="post-3">Item 3</li>

<li id="post-4">Item 4</li>

<li id="post-5">Item 5</li>

<li id="post-6">Item 6</li>

</ul>

当子元素的事件冒泡到父ul元素时，你可以检查事件对象的target属性，捕获真正被点击的节点元素的引用。下面是一段很简单的JavaScript代码，演示了事件委托的过程：

// 找到父元素，添加监听器...

document.getElementById("parent-list").addEventListener("click",function(e) {

// e.target是被点击的元素!

// 如果被点击的是li元素

if(e.target && e.target.nodeName == "LI") {

// 找到目标，输出ID!

console.log("List item ",e.target.id.replace("post-")," was clicked!");

}

});

* + - 1. typeof(null)=\_\_\_\_\_\_

答案为object

* + - 1. NaN \* 6=\_\_\_\_\_\_

答案为NaN

* + - 1. JS中调用某个函数之前，如何取得该函数最多可以传递多少个参数？该函数被调用时，如果知道传了多少个参数过来？

假设函数名为fun，那个fun.length就是它最多能接受的参数个数；

在fun函数里面，arguments就是用数组装着调用时传过来的所有参数，因此arguments.length就是已经传递过来的参数个数；

* + - 1. JS如何得到HTTP的请求头信息和返回的头信息？

Javascript中跟response header有关的就两个方法：

getResponseHeader 从响应信息中获取指定的http头 语法  
strValue =  
oXMLHttpRequest.getResponseHeader(bstrHeader);  
getAllResponseHeaders 获取响应的所有http头 语法  
strValue =  
oXMLHttpRequest.getAllResponseHeaders();

需要注意的是，通常，在IE下不能完整的获取header报头数据，只能取到如下header数据：

X-Powered-By:  
X-UA-Compatible:  
Keep-Alive:  
Transfer-Encoding:  
Content-Type:  
比如你要获取时间戳，在IE下必须做些特殊处理，需要在后端设置一下，关闭缓存：

header( 'Cache-Control: no-store'); // 关闭缓存

* + - 1. 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。

* + - 1. link 和@import 的区别是?

（1）、link属于XHTML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XHTML标签，无兼容问题;

（4）、link方式的样式的权重 高于@import的权重.

* + - 1. 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%;

• }

* + - 1. 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 {}

* + - 1. 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

• 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；

• 浏览器兼容问题：

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。

2、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]-->

* + - 1. JavaScript原型，原型链 ?

我们创建的每个函数都有一个prototype属性，这个属性是一个指针，指向一个对象，这个对象的用途是包含可以由特定类型的所有实例共享的属性和方法。那么，prototype就是通过调用构造函数而创建的那个对象实例的原型对象。

使用原型的好处是可以让对象实例共享它所包含的属性和方法。也就是说，不必在构造函数中添加定义对象信息，而是可以直接将这些信息添加到原型中。使用构造函数的主要问题就是每个方法都要在每个实例中创建一遍。

在JavaScript中,一共有两种类型的值,原始值和对象值。每个对象都有一个内部属性 prototype ,我们通常称之为原型。原型的值可以是一个对象,也可以是null。如果它的值是一个对象，则这个对象也一定有自己的原型。这样就形成了一条线性的链，我们称之为原型链。

* + - 1. 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.

\*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。

* + - 1. 页面重构怎么操作？

编写 CSS、让页面结构更合理化，提升用户体验，实现良好的页面效果和提升性能。

* + - 1. 语义化的理解？

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。

* + - 1. 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：

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; }

* + - 1. (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。

* + - 1. iframe有那些缺点？

框架的优点：

重载页面时不需要重载整个页面，只需要重载页面中的一个框架页(减少了数据的传输，增加了网页下载速度)

方便制作导航栏

框架的缺点：

会产生很多页面，不容易管理

不容易打印

浏览器的后退按钮无效

代码复杂,无法被一些搜索引擎索引到

多数小型的移动设备（PDA 手机）无法完全显示框架

多框架的页面会增加服务器的http请求

由于上面诸多缺点，因此不符合标准网页设计的理念,已经被标准网页设计抛弃

提示: 目前框架的所有优点完全可以使用Ajax实现，因此已经没有必要使用框架了。

* + - 1. 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；

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）。

* + - 1. 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。

* + - 1. Node.js的适用场景

高并发、聊天、实时消息推送

* + - 1. 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)

* + - 1. js的常见内置对象类有哪些？

Date,Array,Math、Number、Boolean、String、Array、RegExp、Function...

* + - 1. 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。

但是总有一个原则，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

* + - 1. 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();

* + - 1. 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');

}

* + - 1. JSON 是什么

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'}

* + - 1. 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

* + - 1. ajax 是什么?有什么优缺点? ajax的交互模型?同步和异步的区别?如何解决跨域问题?

AJAX即“Asynchronous Javascript And XML”（异步JavaScript和XML），是指一种创建交互式网页应用的网页开发技术。

优点：

1. 通过异步模式，实现动态不刷新（局部刷新），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。

ajax的缺点

1、ajax不支持浏览器back按钮。

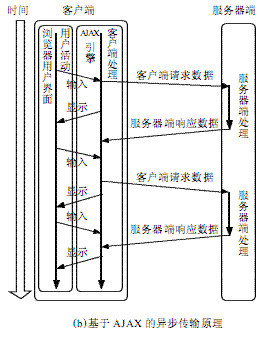
2、安全问题 AJAX暴露了与服务器交互的细节。

3、对搜索引擎的支持比较弱。

4、破坏了程序的异常机制。

5、不容易调试。

AJAX交互模型



同步需要等待返回结果才能继续，异步不必等待

同步：脚本会停留并等待服务器发送回复然后再继续

异步：脚本允许页面继续其进程并处理可能的回复

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

* + - 1. 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

var module1 = (function(){

　　　　var \_count = 0;

　　　　var m1 = function(){

　　　　　　//...

　　　　};

　　　　var m2 = function(){

　　　　　　//...

　　　　};

　　　　return {

　　　　　　m1 : m1,

　　　　　　m2 : m2

　　　　};

　　})();

* + - 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。

* + - 1. 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“6”。

* + - 1. 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

* + - 1. 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)

* + - 1. JavaScript中的作用域与变量声明提升？

变量声明提升，就是把下面的变量和方法的声明提到上面。

比如下面这两段代码的意义其实是一样的，第二段是第一段代码的声明提升的详细表述

第一段代码：

var v='Hello World';   
(function(){   
alert(v);   
var v='I love you';   
})();

第二段代码：

var v='Hello World';   
(function(){

var v;  
alert(v);   
v='I love you';   
})();

下面这个网址有详细讲解，希望大家去深入研究下：

http://blog.csdn.net/sunxing007/article/details/9034253

* + - 1. 前端开发的优化问题

(1) 减少http请求次数：css spirit,data uri

　　(2) JS，CSS源码压缩

　　(3) 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

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

　　(5) 用setTimeout来避免页面失去响应

　　(6) 用hash-table来优化查找

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

　　(8) 少用全局变量

　　(9) 缓存DOM节点查找的结果

　　(10) 避免使用CSS Expression

　　(11) 图片预载

　　(12) 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

* + - 1. 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）

详细版：

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

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

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

4、进行HTTP协议会话，客户端发送报头(请求报头);

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

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

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

8、浏览器开始下载html文档(响应报头，状态码200)，同时使用缓存;

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

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

简洁版：

浏览器根据请求的URL交给DNS域名解析，找到真实IP，向服务器发起请求；

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

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

载入解析到的资源文件，渲染页面，完成。

这个过程可以分为四个步骤：

1、当发送一个URL请求时，不管这个URL是web页面的URL还是web页面的每个资源的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，这样就可以使得浏览器获得请求对应的IP地址了。

2、浏览器通过与远程web服务器TCP三次握手协商来建立一个TCP/IP链接。三个握手包含一个同步报文，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进行传递，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，最后由由客户端发出该请求已经被接受的报文。

3、一旦TCP/IP链接建立，浏览器会通过该链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状态码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。

* + - 1. 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）

* + - 1. 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、npmM、MVVM

新的东西，需要去了解下，不一定要懂，但至少应该去官网看看，自己写写demo

* + - 1. 判断字符串是否是这样组成的，第一个必须是字母，后面可以是字母、数字、下划线，总长度为5-20

var reg = /^[a-zA-Z][a-zA-Z\_0-9]{4,19}$/;

　　reg.test("a1a\_\_a1a\_\_a1a\_\_a1a\_\_");

* + - 1. 截取字符串abcdefg的efg

var str = "abcdefg";

　　if (/efg/.test(str)) {

　　var efg = str.substr(str.indexOf("efg"), 3);

　　alert(efg);

　　}

* + - 1. 判断一个字符串中出现次数最多的字符，统计这个次数

//将字符串的字符保存在一个hash table中，key是字符，value是这个字符出现的次数

　　var str = "abcdefgaddda";

　　var obj = {};

　　for (var i = 0, l = str.length; i < l; i++) {

　　var key = str[i];

　　if (!obj[key]) {

　　obj[key] = 1;

　　} else {

　　obj[key]++;

　　}

　　}

　　/\*遍历这个hash table，获取value最大的key和value\*/

　　var max = -1;

　　var max\_key = "";

　　var key;

　　for (key in obj) {

　　if (max < obj[key]) {

　　max = obj[key];

　　max\_key = key;

　　}

　　}

　　alert("max:"+max+" max\_key:"+max\_key);

* + - 1. IE与FF脚本兼容性问题

(1) window.event：

　　表示当前的事件对象，IE有这个对象，FF没有，FF通过给事件处理函数传递事件对象

　　(2) 获取事件源

　　IE用srcElement获取事件源，而FF用target获取事件源

　　(3) 添加，去除事件

　　IE：element.attachEvent(“onclick”, function) element.detachEvent(“onclick”, function)

　　FF：element.addEventListener(“click”, function, true) element.removeEventListener(“click”, function, true)

　　(4) 获取标签的自定义属性

　　IE：div1.value或div1[“value”]

　　FF：可用div1.getAttribute(“value”)

　　(5) document.getElementByName()和document.all[name]

　　IE;document.getElementByName()和document.all[name]均不能获取div元素

　　FF：可以

　　(6) input.type的属性

　　IE：input.type只读

　　FF：input.type可读写

　　(7) innerText textContent outerHTML

　　IE：支持innerText, outerHTML

　　FF：支持textContent

　　(8) 是否可用id代替HTML元素

　　IE：可以用id来代替HTML元素

　　FF：不可以

　　这里只列出了常见的，还有不少，更多的介绍可以参看JavaScript在IE浏览器和Firefox浏览器中的差异总结

* + - 1. 规避javascript多人开发函数重名问题

(1) 可以开发前规定命名规范，根据不同开发人员开发的功能在函数前加前缀

(2) 将每个开发人员的函数封装到类中，调用的时候就调用类的函数，即使函数重名只要类名不重复就ok

* + - 1. javascript面向对象中继承实现

javascript面向对象中的继承实现一般都使用到了构造函数和Prototype原型链，简单的代码如下：

　　function Animal(name) {

　　this.name = name;

　　}

　　Animal.prototype.getName = function() {alert(this.name)}

　　function Dog() {};

　　Dog.prototype = new Animal("Buddy");

　　Dog.prototype.constructor = Dog;

　　var dog = new Dog();

* + - 1. 编写一个方法 求一个字符串的字节长度

一个英文字符占用一个字节，一个中文字符占用两个字节

　　function GetBytes(str){

var len = str.length;

var bytes = len;

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

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

}

return bytes;

}

alert(GetBytes("你好,as"));

* + - 1. 如何显示/隐藏一个DOM元素?

el.style.display = "";

　　el.style.display = "none";

　　el是要操作的DOM元素

* + - 1. 如何控制alert中的换行

使用\n

alert("p\np");

* + - 1. 多浏览器检测通过什么?

(1) navigator.userAgent

(2) 不同浏览器的特性，如addEventListener

* + - 1. 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

* + - 1. null，undefined 的区别？

null和undefined基本是同义的，只有一些细微的差别

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。

Object.getPrototypeOf(Object.prototype)

// null

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。

var i;

i // undefined

function f(x){console.log(x)}

f() // undefined

var o = new Object();

o.p // undefined

var x = f();

x // undefined