HTML

Doctype作用？严格模式与混杂模式如何区分？它们有何意义?

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

2、严格模式和混杂模式的如何区分？他们有什么意义？

标准模式(严格模式)的排版和JS运作模式都是以该浏览器支持的最高标准运行。

在兼容模式（混杂模式或怪异模式）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。

意义：随着标准一致性越来越重要，浏览器开发商不得不面临一个艰难的抉择：逐渐遵循w3c的标准是前进的方向。但是改变现有的css，完全去遵循标准，会使许多旧网站或多或少的收到破坏，如果浏览器突然以正确的方式解析现存的css，陈旧的网站的显示必然会受到影响。所以，所有的浏览器都需要提供两种模式，混杂模式服务于旧世规则，严格模式服务于标准规则。

HTML5 为什么只需要写 <!DOCTYPE HTML>？

**html5**不基于SGML,因此不需要对DTD进行引用，但是需要doctype来规范浏览器的行为（让浏览器按照他们应该的方式来运行）

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

行内元素有哪些？块级元素有哪些？ 空(void)元素有那些？

行内元素：a、b、span、img、input、strong、select、label、em、button、textarea

块级元素：div、ul、li、dl、dt、dd、p、h1-h6、blockquote

空元素：即系没有内容的HTML元素，例如：br、meta、hr、link、input、img

页面导入样式时，使用link和@import有什么区别？

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

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

区别3：link是XHTML标签，无兼容问题；@import是在CSS2.1提出的，低版本(IE5及以下)的浏览器不支持。

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

介绍一下你对浏览器内核的理解？

主要分成两个部分：

渲染引擎(Render Engine)

和JS引擎。

渲染引擎：负责取得网页的内容(html,xml和图像等)，整理讯息(例如假如css)，以及计算网页的显示方式，然后输出到显示器或打印机。浏览器的内核的不同对于网页的语法解释会有不同，所以渲染的效果也不同。所有网页浏览器、电子邮件客户端以及它需要编辑、显示网络内容的应用程序都需要内核。

JS引擎：解析和执行JavaScript来实现网页的动态效果。

最开始渲染引擎和JS引擎并没有区分的很明确，后来JS引擎越来越独立，内核就倾向与只指渲染引擎。

常见的浏览器内核有哪些？

Trident内核：IE，360，搜过浏览器；

Gecko内核：Netscape6及以上版本，

Presto内核：Opera

Blink内核：Opera；

Webkit内核：Safari，Chrome

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？

第一行的代码html比较长 h5的短

新特性：

1）.拖拽释放（drag and drop）API

2）.语意化更好的内容标签（header，nav，footer，aside，article，section）

3）.音频，视频API（audio，video）

4）.画布（canvas）API

5).地理（Geolocation）API

6).本地离线存储localStorage长期存储数据，浏览器关闭后数据不会丢失；

7）.sessionStorage的数据在浏览器关闭后自动删除。

8）.表单控件，calendar,date,time,email，URL，search

9）.新的技术webworker，websocket，geolocation

移除的元素：

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

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

支持HTML5新标签：

1）IE8/IE7/IE6支持通过document.createElement方法产生的标签，可以利用这一特性让这些浏览器支持HTML5新标签，浏览器支持新标签后，还需要添加标签默认的样式。

如何区分 HTML 和 HTML5:

DOCTYPE声明新增的结构元素，功能元素。

简述一下你对HTML语义化的理解？

(1)HTML 语义化让页面的内容结构化，结构更清晰，便于对浏览器、搜索引擎解析;

(2)即使在没有样式 CSS 的情况下也能以一种文档格式显示，并且是容易阅读的;

(3)搜索引擎的爬虫也依赖于 HTML 标记来确定上下文和各个关键字的权重，有利于 SEO;

(4)使阅读源代码的人更容易将网站分块，便于阅读、维护和理解。

HTML5的离线储存怎么使用，工作原理能不能解释一下？

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

原理：HTML5的离线存储是基于一个新建的.appcache文件的缓存机制(不是存储技术)，通过这个文件上的解析清单离线存储资源，这些资源就会像cookie一样被存储了下来。之后当网络在处于离线状态下时，浏览器会通过被离线存储的数据进行页面展示

如何使用

页面头部像下面一样加入一个manifest的属性；

在cache.manifest文件的编写离线存储的资源

在离线状态时，操作window.applicationCache进行需求实现

浏览器是怎么对HTML5的离线储存资源进行管理和加载的呢？

1在线的情况下，浏览器发现html头部有manifest属性，它会请求manifest文件，如果是第一次访问app，那么浏览器就会根据manifest文件的内容下载相应的资源并且进行离线存储。如果已经访问过app并且资源已经离线存储了，那么浏览器就会使用离线的资源加载页面，然后浏览器会对比新的manifest文件与旧的manifest文件，如果文件没有发生改变，就不做任何操作，如果文件改变了，那么就会重新下载文件中的资源并进行离线存储。

2离线的情况下，浏览器就直接使用离线存储的资源。

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

相同点：都存储在客户端

不同点：

1.存储大小

cookie数据大小不能超过4k。

sessionStorage和localStorage 虽然也有存储大小的限制，但比cookie大得多，可以达到5M或更大。

2.有效时间

localStorage    存储持久数据，浏览器关闭后数据不丢失除非主动删除数据；

sessionStorage  数据在当前浏览器窗口关闭后自动删除。

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

3. 数据与服务器之间的交互方式

cookie的数据会自动的传递到服务器，服务器端也可以写cookie到客户端

sessionStorage和localStorage不会自动把数据发给服务器，仅在本地保存。

iframe有那些缺点？

1.iframe会阻塞主页面的onload事件

2.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。

3.iframe框架结构很迷惑，如果嵌套多个iframe页面的货出现多个滚动条，用户体验差；

4.代码复杂，不容易被搜索引擎搜索到，所以iframe不利于搜索引擎优化，很多搜索引擎爬虫还不能很好的处理iframe的内容；

5.很多移动设备无法完全显示框架，设备兼容性差；

6.iframe框架页面会增加服务器的http请求，对于大型网站是不可取的

  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。

Label的作用是什么？是怎么用的？（加 for 或 包裹）

label标签来定义表单控制间的关系,当用户选择该标签时，浏览器会自动将焦点转到和标签相关的表单控件上。

<label for="Name">Number:</label> <input type=“text“name="Name" id="Name"/>

<label>Date:<input type="text" name="B" /></label>

HTML5的form如何关闭自动完成功能？

HTML的输入框可以拥有自动完成的功能，当你往输入框输入内容的时候，浏览器会从你以前的同名输入框的历史记录中查找出类似的内容并列在输入框下面，这样就不用全部输入进去了，直接选择列表中的项目就可以了。  
  
但有时候我们希望关闭输入框的自动完成功能，例如当用户输入内容的时候，我们希望使用AJAX技术从数据库搜索并列举而不是在用户的历史记录中搜索。  
  
关闭输入框的自动完成功能有3种方法：  
  
1、在IE的Internet选项菜单里的内容--自动完成里面设置  
  
2、设置Form的autocomplete为"on"或者"off"来开启或者关闭自动完成功能  
  
3、设置输入框的autocomplete为"on"或者"off"来开启或者关闭该输入框的自动完成功能

如何实现浏览器内多个标签页之间的通信? (阿里)

**方法一：使用localStorage**

使用localStorage.setItem(key,value);添加内容

使用storage事件监听添加、修改、删除的动作

**方法二、使用cookie+setInterval**

webSocket如何兼容低浏览器？(阿里)  
Adobe Flash Socket 、 ActiveX HTMLFile (IE) 、 基于 multipart 编码发送 XHR 、 基于长轮询的 XHR

页面可见性（Page Visibility）API 可以有哪些用途？

页面可见性： 就是对于用户来说，页面是显示还是隐藏, 所谓显示的页面，就是我们正在看的页面；隐藏的页面，就是我们没有看的页面。 因为，我们一次可以打开好多标签页面来回切换着，始终只有一个页面在我们眼前，其他页面就是隐藏的，还有一种就是.........，(把浏览器最小化，所有的页面就都不可见了)。

API 很简单，document.hidden 就返回一个布尔值，如果是true, 表示页面可见，false 则表示，页面隐藏。  不同页面之间来回切换，触发visibilitychange事件。 还有一个document.visibilityState, 表示页面所处的状态，取值：visible, hidden 等四个。

document.addEventListener("visibilitychange", function(){

if(document.hidden){

document.title ="hidden";

}else {

document.title = "visibile";

}

})

我们打开这个页面，然后再打开另一个页面，来回点击这两个页面，当我们看到这个页面时，标题显示visiable ,当我们看另一个页面时，标题显示hidden;

动画，视频，音频都可以在页面显示时打开，在页面隐藏时关闭。

如何在页面上实现一个圆形的可点击区域？

方案一：

定义一个客户端图像映射。图像映射（image-map）指带有可点击区域的一幅图像。

<img src="task6.jpg" width="1366" height="768" border="0" usemap="#Map" />

<map name="Map" id="Map">

<area shape="circle" coords="100,100,50" href="https://www.baidu.com" target="\_blank" />

</map>

方案二：

<style>

.disc{

width:100px;

height:100px;

background-color:dimgray;

border-radius: 50%;

cursor: pointer;

position: absolute;

left:50px;

top:50px;

line-height: 100px;

text-align: center;

color: white;

}

</style>

<div class="disc">点击区域</div>

方案三：

<script>

document.onclick = function(e){

var r = 50; //圆的半径

var x1 = 100, y1 = 100;

var x2 = e.clientX,

y2 = e.clientY;

var len=Math.abs(Math.sqrt(Math.pow(x2-x1,2)+Math.pow(y2-y1,2)));

if(len<=50){

console.log("Inner");

}else{

console.log("Outer");

}

}

</script>

实现不使用 border 画出1px高的线，在不同浏览器的Quirksmode和CSSCompat模式下都能保持同一效果。

解析：<div style=”height:1px;overflow:hidden;background:red”></div>

网页验证码是干嘛的，是为了解决什么安全问题？

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

title与h1的区别、b与strong的区别、i与em的区别？

title与h1的区别

　　定义：

　　　　title是网站标题，

　　　　h1是文章主题

　　作用：

　　　　title概括网站信息，可以直接告诉搜索引擎和用户这 个网站是关于什么主题和内容的，是显示在网页Tab栏里的；

　　　　h1突出文章主题，面对用户，更突出其视觉效果，指向 页面主体信息，是显示在网页中的。

b与strong的区别

　　定义：

　　　　b(bold)是实体标签，用来给文字加粗，

　　　　strong是逻辑标签，作用是加强字符语气。

　　区别：

　　　　b标签只是加粗的样式，没有实际含义，常用来表达无强调或着重意味的粗体文字，比如文章摘要中的关键词、 评测文章中的产品名称、文章的导言；

　　　　strong表示标签内字符重要，用以强调，其默认格式是加粗，但是可以通 过CSS添加样式，使用别的样式强调。

　　建议：为了符合CSS3的规范，b应尽量少用而改用strong

i与em的区别

　　定义：

　　　　i(italic)是实体标签，用来使字符倾斜，

　　　　em(emphasis)是逻辑标签，作用是强调文本内容

　　区别：

　　　　i标签只是斜体的样式，没有实际含义，常用来表达无强调或着重意味的斜体，比如生物学名、术语、外来语；

　　　　em表示标签内字符重要，用以强调，其默认格式是斜体，但是可以通过CSS添加样式。

　　建议：为了符合CSS3的规 范，i应尽量少用而改用em

CSS

介绍一下标准的CSS的盒子模型？与低版本IE的盒子模型有什么不同的？

这是一个很基础的前端问题，在考察前端基础功底的时候，特别是在校招的情况下会经常被问到。

标准的盒模型主要有两种：标准(W3C)盒模型，怪异(IE)盒模型 。两者的区别就在于元素宽度计算方式不一样。盒子模型构成：内容(content)、内填充(padding)、 边框(border)、外边距(margin)。IE8及其以下版本浏览器，未声明 DOCTYPE，内容宽高会包含内填充和边框，被称为怪异盒模型(IE盒模型)。

在标准盒模型中，元素宽度 = width + padding + border + margin。

怪异(IE)盒模型中，元素宽度 = width + margin

标准浏览器通过设置 css3 的 box-sizing: border-box 属性，触发“怪异模式”解析计算宽高

CSS选择符有哪些？哪些属性可以继承？

\*    1.id选择器（ # myid）  
   2.类选择器（.myclassname）  
   3.标签选择器（div, h1, p）  
   4.相邻选择器（h1 + p）  
   5.子选择器（ul > li）  
   6.后代选择器（li a）  
   7.通配符选择器（ \* ）  
   8.属性选择器（a[rel = "external"]）  
   9.伪类选择器（a:hover, li:nth-child）  
  \*   可继承的样式： font-size font-family color, UL LI DL DD DT;  
  \*   不可继承的样式：border padding margin width height ;

CSS优先级算法如何计算？

 \*   优先级就近原则，同权重情况下样式定义最近者为准;  
 \*   载入样式以最后载入的定位为准;  
  优先级为:  
   同权重: 内联样式表（标签内部）> 嵌入样式表（当前文件中）> 外部样式表（外部文件中）。  
   !important >  id > class > tag    important 比 内联优先级高

CSS3新增伪类有那些？

   :after          在元素之前添加内容，也可以用来做清除浮动。

   :before        在元素之后添加内容。

   :enabled        选择可用的表单元素。

   :disabled       控制表单控件的禁用状态。

    :checked        单选框或复选框被选中。

如何居中div？如何居中一个浮动元素？如何让绝对定位的div居中？

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Document</title>

    <style>

        body { height: 4000px; }

        div { width: 500px; height: 320px; background: pink; position: fixed; top: 50%; left: 50%; margin:-160px 0 0 -250px; }

    </style>

</head>

<body>

    <div></div>

</body>

</html>

重点是下面的样式：position: fixed; top: 50%; left: 50%; margin:-160px 0 0 -250px;

-160px和-250px分别是div元素高度和宽度的一半。负号你懂的，你相反方向移动。

display有哪些值？说明他们的作用。

display 的属性值有：none|inline | block | inline-block | list-item | run-in | table | inline-table | table-row-group | table-header-group | table-footer-group | table-row | table-column-group | table-column | table-cell | table-caption | inherit  
  
其中常用的的有none、inline、block、inline-block。分别的意思是：

1、none： 元素不会显示，而且改元素现实的空间也不会保留。但有另外一个 visibility: hidden， 是保留元素的空间的。

2、inline： display的默认属性。将元素显示为内联元素，元素前后没有换行符。我们知道内联元素是无法设置宽高的，所以一旦将元素的display 属性设为 inline， 设置属性height和width是没有用的。此时影响它的高度一般是内部元素的高度（font-size）和padding。

3、block： 将元素将显示为块级元素，元素前后会带有换行符。设置为block后，元素可以设置width和height了。元素独占一行。

4、inline-block：行内块元素。这个属性值融合了inline 和 block 的特性，即是它既是内联元素，又可以设置width和height。

position的值relative和absolute定位原点是？

relative（相对定位）：定位原点是元素本身所在位置；  
absolute（绝对定位）：定位原点是离自己这一级元素最近的一级position设置为absolute或者relative的父元素的左上角为原点的

CSS3有哪些新特性？

1. CSS3实现圆角（border-radius），阴影（box-shadow），边框图片border-image

2. 对文字加特效（text-shadow、），强制文本换行（word-wrap），线性渐变（linear-gradient）

3.旋转,缩放,定位,倾斜：transform:rotate(90deg) scale(0.85,0.90) translate(0px,-30px) skew(-9deg,0deg);

4. 增加了更多的CSS选择器、多背景、rgba()；

5. 在CSS3中唯一引入的伪元素是 ::selection ；

6. 媒体查询(@media)，多栏布局（flex）

请解释一下CSS3的Flexbox（弹性盒布局模型）,以及适用场景？

设为Flex布局以后（ display: flex;），子元素的float、clear和vertical-align属性将失效。

采用Flex布局的元素，称为Flex容器（flex container），简称”容器”。

它的所有子元素自动成为容器成员，称为Flex项目（flex item），简称”项目”。

容器默认存在两根轴：水平的主轴（main axis）和垂直的交叉轴（cross axis）。

主轴的开始位置（与边框的交叉点）叫做main start，结束位置叫做main end；交叉轴的开始位置叫做cross start，结束位置叫做cross end。

项目默认沿主轴排列。单个项目占据的主轴空间叫做main size，占据的交叉轴空间叫做cross size。

**容器的属性**

以下6个属性设置在容器上（justify-content和align-items、flex-wrap:wrap最常用的）。

* justify-content：定义了项目在主轴上的对齐方式。它可能取5个值
  + center： 居中
  + flex-start（默认值）：左对齐
  + flex-end：右对齐
  + space-between：两端对齐，项目之间的间隔都相等。
  + space-around：每个项目两侧的间隔相等。所以，项目之间的间隔比项目与边框的间隔大一倍。
* align-items：属性定义项目在交叉轴上如何对齐。它可能取5个值。
  + center：交叉轴的中点对齐。
  + flex-start：交叉轴的起点对齐。
  + flex-end：交叉轴的终点对齐。
  + baseline: 项目的第一行文字的基线对齐。
  + stretch（默认值）：如果项目未设置高度或设为auto，将占满整个容器的高度。
* flex-flow：flex-flow属性是flex-direction属性和flex-wrap属性的简写形式，默认值为row nowrap。
* flex-direction：属性决定主轴的方向（即项目的排列方向）；
  + row（默认值）：主轴为水平方向，起点在左端。
  + row-reverse：主轴为水平方向，起点在右端。
  + column：主轴为垂直方向，起点在上沿。
  + column-reverse：主轴为垂直方向，起点在下沿。
* flex-wrap：默认情况下，项目都排在一条线（又称”轴线”）上。flex-wrap属性定义，如果一条轴线排不下，如何换行。
  + 它可能取三个值。
  + （1）nowrap（默认）：不换行。
  + （2）wrap：换行，第一行在上方。【这个属性经常用】
  + （3）wrap-reverse：换行，第一行在下方。
* align-content：属性定义了多根轴线的对齐方式。如果项目只有一根轴线，该属性不起作用
  + flex-start：与交叉轴的起点对齐。
  + flex-end：与交叉轴的终点对齐。
  + center：与交叉轴的中点对齐。
  + space-between：与交叉轴两端对齐，轴线之间的间隔平均分布。
  + space-around：每根轴线两侧的间隔都相等。所以，轴线之间的间隔比轴线与边框的间隔大一倍。
  + stretch（默认值）：轴线占满整个交叉轴。

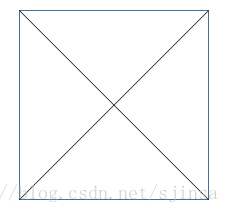
**项目的属性总结**

以下6个属性设置在项目上。（align-self、flex、order）

* **align-self**：
  + 属性允许单个项目有与其他项目不一样的对齐方式，可覆盖align-items属性。默认值为auto，表示继承父元素的align-items属性，如果没有父元素，则等同于stretch；该属性可能取6个值，除了auto，其他都与align-items属性完全一致。
  + auto / flex-start / flex-end / center / baseline / stretch;
* **flex**：
  + 属性是flex-grow, flex-shrink 和 flex-basis的简写，默认值为0 1 auto。后两个属性可选。该属性有两个快捷值：auto (1 1 auto) 和 none (0 0 auto)。建议优先使用这个属性，而不是单独写三个分离的属性，因为浏览器会推算相关值。
* **order**：
  + 属性定义项目的排列顺序。数值越小，排列越靠前，默认为0。
* **flex-grow**：
  + 属性定义项目的放大比例，默认为0，即如果存在剩余空间，也不放大。如果所有项目的flex-grow属性都为1，则它们将等分剩余空间（如果有的话）。如果一个项目的flex-grow属性为2，其他项目都为1，则前者占据的剩余空间将比其他项多一倍。
* **flex-shrink**：
  + flex-shrink属性定义了项目的缩小比例，默认为1，即如果空间不足，该项目将缩小。如果所有项目的flex-shrink属性都为1，当空间不足时，都将等比例缩小。如果一个项目的flex-shrink属性为0，其他项目都为1，则空间不足时，前者不缩小。负值对该属性无效。
* **flex-basis**：
  + flex-basis属性定义了在分配多余空间之前，项目占据的主轴空间（main size）。浏览器根据这个属性，计算主轴是否有多余空间。它的默认值为auto，即项目的本来大小。它可以设为跟width或height属性一样的值（比如350px），则项目将占据固定空间。

用纯CSS创建一个三角形的原理是什么？

1.采用的是均分原理

非常有意思的是盒子都是一个矩形或正方形，从形状的中心，向4个角上下左右划分4个部  


2.代码的实现

第一步 保证元素是块级元素

第二步 设置元素的边框

第三步 不需要显示的边框使用透明色

.square{

width:0;

height:0;

margin:0 auto;

border:6px solid transparent;

border-top: 6px solid red;

}

第二种写法：

.square{

width:0;

height:0;

margin:0 auto;

border-width:6px;

border-color:red transparent transparent transparent;

border-style:solid dashed dashed dashed;*//为了兼容IE6，把没有的边框都设置为虚线*

}

一个满屏 品 字布局 如何设计?

需要用到技术

1.     元素水平居中对齐

1)        使用margin对齐(推荐)

2)        使用left：50%

3)        使用text-align

2.  元素对相对窗口定位

1)        使用filxed（推荐）

2)        使用absolute定位

3)        使用html和body的width和height填?这个窗口

3.     元素左右定位

1)        使用float左右浮动

2)        使用绝对定位进行左右定位（推荐）

具体实现的代码

html代码

<div class="main">

<div class="wrapper-up">

<div class="div-square-up"></div>

</div>

<div class="wrapper-down">

<div class="div-square-left"></div>

<div class="div-square-right"></div>

</div>

</div>

CSS样式

body{

height: 1200px;

}

.main {

position: fixed;

left: 0;

top: 0;

height: 100%;

width: 100%;

}

.wrapper-up {

height: 50%;

}

.wrapper-down {

height: 50%;

position: relative;

}

.div-square-up {

width: 50%;

margin: 0 auto;

border: 2px solid red;

height: 96%;

box-sizing: border-box;

}

.div-square-left {

position: absolute;

left: 0;

width: 48%;

height: 100%;

box-sizing: border-box;

border: 2px solid red;

}

.div-square-right {

position: absolute;

right: 0;

width: 48%;

height: 100%;

border: 2px solid red;

box-sizing: border-box;

}

常见兼容性问题？

1. 不同浏览器的标签默认外补丁margin和内补丁padding不同

发生概率：100%

解决方案：使用CSS通配符\*，设置内外补丁为0

\*{ margin: 0; padding: 0;}

2.  块属性标签float之后，又有横向的margin值，在IE6中显示会比设置的大（IE6双边距bug）

  发生概率：90%

   解决方案：在float标签样式控制中加入display:inline;

3. 设置较小的高度标签（一般小于10px），在IE6，IE7，遨游中超出自己设置的高度

发生概率：60%

解决方案：给超出高度的标签设置overflow:hidden;或者设置行高line-height小于你设置的高度。

4. 行内标签设置display:block;后又采用float布局，再设置横向margin值时，在IE6中显示会比设置的大（IE6双边距bug）

发生概率：20%

解决方案：在display:block;后面加上display:inline;display:table;

5. 图片默认有间距

发生概率：20%

解决方案：使用float为img布局

6. 标签最低高度设置min-height不兼容

发生概率：5%

解决方案：例如要设置一个标签的最小高度为200px

{ min-height: 200px;

 height: auto!important;

height: 200px;

overflow: visible;}

7. 透明度兼容设置

发生概率：主要看你要写的东西设不设透明度

解决方案：一句话

transparent\_class {

    filter:alpha(opacity=50);

       -moz-opacity:0.5;

       -khtml-opacity: 0.5;

       opacity: 0.5;

 }

**opacity:0.5;** This is the “most important” one because it is the currentstandard in CSS. This will work in most versions of Firefox, Safari, andOpera.This would be all you need if all browsers supported current standards. Which,of course, they don’t.

**filter:alpha(opacity=50);** This one you need for IE.

**-moz-opacity:0.5;** You need this one to support way old school versions of theMozilla browsers like Netscape Navigator.

**-khtml-opacity:0.5;** This is for way old versions of Safari (1.x) when therendering engine it was using was still referred to as KTHML, asopposed to thecurrent [WebKit](http://webkit.org/) .

8. Box Model的bug

描述：给一个元素设置了高度和宽度的同时，还为其设置margin和padding的值，会改变该元素的实际大小。

解决办法：在需要加 margin和padding的div内部加一个div,在这个div里设置margin和padding值。

9. IE6中的列表li楼梯状bug

描述：通常在li中的元素（比如a）设置了浮动float，但li本身不浮动。

解决办法：

ul li{float:left;}

或 ul li{display:inline;}

10.li空白间距

描述：在IE下，会增加li和li之间的垂直间距

解决办法：给li里的a显式的添加宽度或者高度

li a{width:20px;}

或者

li a{display:block;float:left;clear:left;}

或者

li {display:inline;}

li a{display:block;}

或者

在每个列表li上设置一个实线的底边，颜色和li的背景色相同

11.overflow：auto;和position:relative的碰撞

描述：此bug只出现在IE6和IE7中，有两个块级元素，父元素设置了overflow：auto;子元素设置了position:relative;且高度大于父元素，在IE6-7中子元素不会被隐藏而是溢出。

解决方案：给父元素也设置position:relative;

12.浮动层的错位

描述：当内容超出外包容器定义的宽度时会导致浮动层错位问题。在Firefox、IE7、IE8及其他标准浏览器里,超出的内容仅仅只是超出边缘;但在IE6中容器会忽视定义的width值,宽度会错误地随内容宽度增长而增长。如果在这个浮动元素之后还跟着一个浮动元素,那么就会导致错位问题。

解决方案：overflow：hidden;

13.IE6克隆文本的bug

描述：若你的代码结构如下

<!--这是注释-->

     <div>

       ……

   </div>

<!--这是注释-->

很有可能在IE6网页上出现一段空白文本

解决方案：

使用条件注释

删除所有注释

在注释前面的那个浮动元素加上 display：inline；

14.IE的图片缩放

描述：图片在IE下缩放有时会影响其质量

解决方案：img{ -mg-interpolation-mode:bicubic;}

15.IE6下png图片的透明bug

描述：使用透明图片,使用png24或png32图片在IE6下面显示图片会有一层淡蓝色的背景。

解决方案：

.img{

background:url('http://shenmo.wanmei.com/images/logo/sm\_logo\_202x104.png');

\_background:0;

\_filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(src='http://shenmo.wanmei.com/images/logo/sm\_logo\_202x104.png',sizingMethod='scale');

}

img{filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(src='http://shenmo.wanmei.com/images/logo/sm\_logo\_202x104.png',sizingMethod='scale');}

或

<imgsrc="test.png" width="247" height="216"style="filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(src='http://shenmo.wanmei.com/images/logo/sm\_logo\_202x104.png',sizingMethod='scale');" alt="" />

16.<iframe>透明背景bug

   描述：在IE浏览器中，<iframe>框架不会自动把背景设为透明

   解决方案：

     <iframesrc="content.html"allowTransparency="true"></iframe>

   在iframe调用的content.html页面中设置

   body{background-color: transparent;}

16.禁用IE默认的垂直滚动条

解决方案：

html{

   overflow:auto;

}

li与li之间有看不见的空白间隔是什么原因引起的？有什么解决办法？

浏览器的默认行为是把inline元素间的空白字符（空格换行tab）渲染成一个空格，也就是我们上面的代码<li>换行后会产生换行字符，而它会变成一个空格，当然空格就占用一个字符的宽度。

解决方案：

方法一：既然是因为<li>换行导致的，那就可以将<li>代码全部写在一排，如下

<div class="wrap">

<h3>li标签空白测试</h3>

<ul>

<li class="part1"></li><li class="part2"></li><li class="part3"></li><li class="part4"></li>

</ul>

</div>

再刷新页面看就没有空白了，就是这么神奇~

方法二：我们为了代码美观以及方便修改，很多时候我们不可能将<li>全部写在一排，那怎么办？既然是空格占一个字符的宽度，那我们索性就将<ul>内的字符尺寸直接设为0，将下面样式放入样式表，问题解决。

.wrap ul{font-size:0px;}

1

但随着而来的就是<ul>中的其他文字就不见了，因为其尺寸被设为0px了，我们只好将他们重新设定字符尺寸。

方法三：本来以为方法二能够完全解决问题，但经测试，将li父级标签字符设置为0在Safari浏览器依然出现间隔空白；既然设置字符大小为0不行，那咱就将间隔消除了，将下面代码替换方法二的代码，目前测试完美解决。同样随来而来的问题是li内的字符间隔也被设置了，我们需要将li内的字符间隔设为默认。

.wrap ul{letter-spacing: -5px;}

1

之后记得设置li内字符间隔

.wrap ul li{letter-spacing: normal;}

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 ？

浏览器兼容性问题

出现浏览器兼容性问题的主要原因，是不同浏览器对同一段代码的有不同解析，造成的页面显示效果不统一。

在大多数情况下，我们的需求是，无论用户在什么浏览器上查看我们的网站或登录我们的系统，显示效果都应该是一致的。所以浏览器兼容性问题是我们web前端开发人员经常会碰到和必须解决的问题。

什么是css hack

不同的浏览器对CSS的解析结果是不同的，因此会导致相同的CSS输出的页面效果不同，这就需要CSS Hack来解决浏览器局部的兼容性问题。而这个针对不同的浏览器写不同的CSS 代码的过程，就叫CSS Hack。

常见的兼容性问题

1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。

2png24位的图片在iE6浏览器上出现背景，解决方案是做成PNG8.

3Chrome 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。 （可通过加入 CSS 属性transform: scale(0.75,0.75); 解决）

4.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。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。

5在ie6，ie7中元素高度超出自己设置高度。原因是IE8以前的浏览器中会给元素设置默认的行高的高度导致的。解决方案是加上overflow:hidden或设置line-height为更小的高度。

6min-height在IE6下不起作用。解决方案是添加 height:auto !important;height:xxpx;其中xx就是min-height设置的值。

7透明性IE用filter:Alpha(Opacity=60)，而其他主流浏览器用 opacity:0.6;

8 a(有href属性)标签嵌套下的img标签，在IE下会带有边框。解决办法是加上a img{border:none;}样式。

9 input边框问题。去掉input边框一般用border:none;就可以，但由于IE6在解析input样式时的BUG(优先级问题)，在IE6下无效。

10 ie6的默认CSS样式，涉及到border的有border-style:inset;border-width:2px;浏览器根据自己的内核解析规则，先解析自身的默认CS再 解析开发者书写的CSS，达到渲染标签的目的。IE6对INPUT的渲染存在bug，border:none;不被解析，当有border-width或border-colo r设置的时候才会令IE6去解析border-style:none;。解决方案是用:border:0或border:0 none;或border:none:border-color:transparent;，推荐用第三种方案。

为什么要初始化CSS样式。

因为浏览器的兼容性，每个浏览器设计时都有自己的特性，因此在解析标签的时候就产生一些差异，因此显示页面的时候就会出现微小的差异。

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

lock-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。

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

visibility有如下属性值：

属性值 属性值描述

visible 默认值。元素是可见的。

hidden 元素是不可见的，相当于display：hidden；，但此时仍占用页面空间

collapse 当在表格元素中使用时，此值可删除一行或一列，但是它不会影响表格的布局。被行或列占据的空间会留给其他内容使用。如果此值被用在其他的元素上，会呈现为 “hidden”。

inherit 规定应该从父元素继承 visibility 属性的值。

visibility的第三种值collapse：

对于一般的元素，它的表现跟display:hidden是一样的。

但例外的是，如果这个元素是table相关的元素，例如table行，table group，table列，table column group，它的表现却跟display: none一样，也就是说，它们占用的空间也会释放。

在不同浏览器下的区别：

在谷歌浏览器里，使用collapse值和使用hidden值没有什么区别。

在火狐浏览器、Opera和IE11里，使用collapse值的效果就如它的字面意思：table的行会消失，它的下面一行会补充它的位置。

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

1如果元素的显示为无，那么元素不被渲染，position，float不起作用，  
2如果元素拥有position：absolute；或者position：fixed；属性那么元素将为绝对定位，float不起作用。

3如果元素float属性不是none，元素会脱离文档流，根据float属性值来显示。  
4有浮动，绝对定位，串联块属性的元素，边距不会和垂直方向上的其他元素边距折叠。

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

（W3C CSS 2.1 规范中的一个概念,它是一个独立容器，决定了元素如何对其内容进行定位,以及与其他元素的关系和相互作用。）一个页面是由很多个 Box 组成的,元素的类型display 属性,决定了这个 Box 的类型。

不同类型的 Box,会参与不同的 Formatting Context（决定如何渲染文档的容器）,因此Box内的元素会以不同的方式渲染,也就是说BFC内部的元素和外部的元素不会互相影响。

CSS权重优先级是如何计算的？

关于CSS权重，需要以套计算公式来计算。

标签选择器 计算权重公式

继承或\* 0，0，0，0

每个元素（标签选择器） 0，0，0，1

每个类，伪类 0，0，1，0

每个ID 0，1，0，1

每个行内样式style 1，0，0，0

!important 重要的 ∞无穷大

CSS的权重继承或\*是最小的（没有权重），!important权重是最大的。

值从左到右，左面的值越大，权重越高，类似我们学过的个十百千万。

CSS除了有权重外，还有权重叠加，在布局中我们经常会使用交集选择器或后代选择器等（.nav .name）等，会出现多个选择器，这时候就会出现权重叠加。权重叠加是这样来计算的：

div ul li------>0，0，0，3；

.nav ul li------> 0，0，1，2；

a:hover------->0，0，1，1；

.nav a------->0，0，1，1

要注意的是这个数位之间是没有进制的，比如：0，0，0，5+0，0，0，5=0，0，0，10，而不是0，0，1，0，因此不会出现10个div能赶上一个类的情况。

继承的权重是0。

如果出现了权重相同的情况，则会根据层叠性的就近原则。

请解释一下为什么会出现浮动和什么时候需要清除浮动？清除浮动的方式

#### 为什么要清除浮动？

当元素有浮动属性时，高度会坍塌，高度的坍塌会对其父元素或后面的元素产生影响，出现布局错乱的现象，可以通过清除浮动的方法来解决。

#### 浮动的影响是怎么产生的？

浮动是将块元素独占一行的属性取消，允许其它元素与它在同一行，其核心就是使这个块从原来的文档流模式中分离出来，它后面 [大专栏](https://www.dazhuanlan.com/)  [为何需要清除浮动，以及清除浮动的方式](https://www.dazhuanlan.com/2020/01/31/5e339fdcd1396/)的对象就视它不存在，浮动元素的高度不能撑起父元素的高度，也就出现了高度坍塌。

#### 清除浮动的方式

1、给父级div定义高度height  
2、父级div设置overflow:hidden;  
3、利用伪对象after方法，作用于浮动元素的父级

.clearFix:after{

clear:both;

display:block;

height:0;

content:"";

}

.clearFix{zoom:1;}

zoom:1的原理：当设置了zoom的值之后，所设置的元素就会扩大或缩小，高度宽度就会重新计算，这里一旦改变zoom值是其实也会发生重新渲染。  
4、底部加一个空的div

<div class="clear"></div>

.clear{clear:both;}

移动端的布局用过媒体查询吗？  
**通过媒体查询可以为不同大小和尺寸的媒体定义不同的css，适应相应的设备的显示。**

使用 CSS 预处理器吗？喜欢那个？

流行是SASS和LESS等 这些预处理器最终得到的还是css文件。所以他们的目的就是一个 提高开发效率。我感觉CSS还有很多东西要学，所以对sass和less仅限于听说过的层次，主要是学习CSS基础的了，如果项目需要的话，

我相信只要CSS基础扎实，用sass或者less都可以很容易上手；

CSS优化、提高性能的方法有哪些？

1.发布前压缩CSS，减少数据传输量。

2.合并属性，如margine-left:5px;margine-top:10px 这个就可以合并成一条的。

3.合理设计CSS 布局，注意复用样式，减少渲染上花的时间。class和ID的选择，少用\*这种全局匹配，合理设置基本样式（如设置table{}）提高复用。

4.减少低效代码的使用，如滤镜，express表达式，!import引入。

5.其它技术，如CSS SPRITE等。

浏览器是怎样解析CSS选择器的？

CSS选择器的解析是从右向左解析的。若从左向右的匹配，发现不符合规则，需要进行回溯，会损失很多性能。若从右向左匹配，先找到所有的最右节点，对于每一个节点，向上寻找其父节点直到找到根元素或满足条件的匹配规则，则结束这个分支的遍历。两种匹配规则的性能差别很大，是因为从右向左的匹配在第一步就筛选掉了大量的不符合条件的最右节点（叶子节点），而从左向右的匹配规则的性能都浪费在了失败的查找上面。  
而在 CSS 解析完毕后，需要将解析的结果与 DOM Tree 的内容一起进行分析建立一棵 Render Tree，最终用来进行绘图。在建立 Render Tree 时（WebKit 中的「Attachment」过程），浏览器就要为每个 DOM Tree 中的元素根据 CSS 的解析结果（Style Rules）来确定生成怎样的 Render Tree。

在网页中的应该使用奇数还是偶数的字体？为什么呢？

## 一、浏览器的原因

其一是为了迁就ie6，万恶的ie6会把定义为13px的字渲染成14px，

还有一个原因是，偶数宽的汉字，比如12px宽的汉字，去掉1像素的间距，填充了像素的实际宽是11px，这样汉字的中竖线左右是平分的，以“中”这个字为例，在12像素时，竖线在中间，左右各5像素，显得均衡。

其二像谷歌一些比较流行的浏览器一般会有个默认的最小字体，而且对奇数字体渲染的不太好看

## 二、实际应用

偶数字号相对更容易和 web 设计的其他部分构成比例关系。比如：当我用了 14 px 的正文字号，我可能会在一些地方用 14 × 0.5 = 7 px 的 margin，在另一些地方用 14 × 1.5 = 21 px 的标题字号。

Windows 自带的点阵宋体（中易宋体）从 Vista 开始只提供 12、14、16 px 这三个大小的点阵，而 13、15、17 px 时用的是小一号的点阵（即每个字占的空间大了 1 px，但点阵没变），于是略显稀疏。

margin和padding分别适合什么场景使用？

抽离样式模块怎么写，说出思路，有无实践经验？[阿里航旅的面试题]

margin:

     需要在border外侧添加空白时；

     空白处不需要背景（色）时；

    上下相连的两个盒子之间的空白，需要相互抵消时。

padding：

    需要在border内测添加空白时；

    空白处需要背景（色）时；

    上下相连的两个盒子之间的空白，希望等于两者之和时。

把常用的css样式单独做成css文件……通用的和业务相关的分离出来，通用的做成样式模块儿共享，业务相关的，放进业务相关的库里面做成对应功能的模块儿。

元素竖向的百分比设定是相对于容器的高度吗？

   相对于父容器的宽度

全屏滚动的原理是什么？用到了CSS的那些属性？

原理：有点类似于轮播，整体的元素一直排列下去，假设有5个需要展示的全屏页面，那么高度是500%，只是展示100%，剩下的可以通过transform进行y轴定位，也可以通过margin-top实现

overflow：hidden；transition：all 1000ms ease；

什么是响应式设计？响应式设计的基本原理是什么？如何兼容低版本的IE？

响应式网站设计(Responsive Web design)的理念是：集中创建页面的图片排版大小，可以智能地根据用户行为以及 使用的设备环境（系统平台、屏幕尺寸、屏幕定向等）进行相对应的布局，无论用户正在使用笔记本还是iPad，我们 的页面都应该能够自动切换分辨率、图片尺寸及相关脚本功能等，以适应不同设备。

响应式Web设计不仅仅是关于屏幕分辨率自适应以及自动缩放的图片等等，它更像是一种对于设计的全新思维模式, 如我们需要兼容不同屏幕分辨率、清晰度以及屏幕定向方式竖屏(portrait)、横屏(landscape)，怎样才能做到让一种 设计方案满足所有情况？

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

视差滚动（Parallax Scrolling）就是这样的效果之一。这种技术通过在网页向下滚动的时候，控   
制背景的移动速度比前景的移动速度慢来创建出令人惊叹的3D效果。  
原理：（1）CSS3实现  
优点：开发时间短、性能和开发效率比较好，缺点是不能兼容到低版本的浏览器  
（2）jquery实现  
通过控制不同层滚动速度，计算每一层的时间，控制滚动效果。  
优点：能兼容到各个版本的，效果可控性好  
缺点：开发起来对制作者要求高  
（3）插件实现方式  
例如：parallax-scrolling，兼容性十分好

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

如何修改chrome记住密码后自动填充表单的黄色背景 ？

（1）单冒号(:)用于CSS3伪类，双冒号(::)用于CSS3伪元素  
在css2之前用的是单冒号，之后css3使用时双冒号。目前除了IE外不兼容双冒号，其他的浏览器兼容双   
冒号，建议还是使用单冒号。  
（2）：：before就是以一个子元素的存在，定义在元素主体内容之前的一个伪元素。并不存在与dom之   
中，只存在在页面之中。同理，after是在主体内容之后显示的。

你对line-height是如何理解的？

行高是指一行文字的高度，具体说是两行文字间基线的距离。  
css中起高度作用的因该是height和line-height，一个没有定义height属性，最终其表现作用一定是   
line-height。  
单行文本垂直居中：把line-height值设置为height一样大小的值可以实现单行文字的垂直居中，其实   
也可以把height删除。  
多行文本垂直居中：需要设置display属性为inline-block。

设置元素浮动后，该元素的display值是多少？（自动变成display:block）

display：block  
IE出现双边框的原因：浮动元素的浮动方向与margin的方向一致会出现双边框。  
解决bug：（1）给浮动元素添加一个display：inline  
（2）给IE6写一个hack，其值为正常值的一半。

怎么让Chrome支持小于12px 的文字？

方法一：首先取消浏览器自动调整功能。  
.classstyle{ -webkit-text-size-adjust:none; font-size:9px; } （现在无效）  
方法二：现在可以使用css3里的一个属性：transform：scale（）  
p{font-size:10px;-webkit-transform:scale(0.8);}//0.8是缩放比例

让页面里的字体变清晰，变细用CSS怎么做？

（-webkit-font-smoothing: antialiased;）

-webkit-font-smoothing在window系统下没有起作用，但是在IOS设备上起作用  
-webkit-font-smoothing：antialiased是最佳的，灰度平滑。

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

在css规范中这么描述的，让一种字体表示为斜体（oblique），如果没有这样样式，就可以使用   
italic。oblique是一种倾斜的文字，不是斜体。

position:fixed;在android下无效怎么处理？

在head头中加入<meta name="viewport" content="width=device-width, initial-scale=1.0,   
maximum-scale=1.0, minimum-scale=1.0, user-scalable=no"/>

如果需要手动写动画，你认为最小时间间隔是多久，为什么？（阿里）

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

display:inline-block 什么时候会显示间隙？(携程)

（1）有空格时候会有间隙 解决：移除空格  
（2）margin正值的时候 解决：margin使用负值  
（3）使用font-size时候 解决：font-size:0、letter-spacing、word-spacing

overflow: scroll时不能平滑滚动的问题怎么处理？

（1）高度尺寸不确定的时候，使用：overflow-y：scroll;  
（2）高度尺寸确定的，要么没有滚动条，要么直接出现，不会出现跳动。  
（3）css3计算calc和vw单位巧妙实现滚动条出现页面不跳动：  
.wrap-outer {  
margin-left: calc(100vw - 100%);  
}  
或.wrap-outer {  
padding-left: calc(100vw - 100%);  
}  
首先，.wrap-outer指的是居中定宽主体的父级，如果没有，创建一个  
然后，calc是css3的计算  
100vw是浏览器的内部宽度，而100%是可用宽度，不含滚动条  
calc（100vw-100%）是浏览器的滚动条的宽度

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

（1）height：calc（100%-100px）  
（2）absolute positioning：外层position：relative；  
百分百自适应元素 position: absolute; top: 100px; bottom: 0; left: 0

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

（1）png是便携式网络图片（Portable Network Graphics）是一种无损数据压缩位图文件格式，   
优点是：压缩比高，色彩好。 大多数地方都可以用。  
（2）jpg是一种针对相片使用的一种失真压缩方法，是一种破坏性的压缩，在色调及颜色平滑变化做的   
不错。在www上，被用来储存和传输照片的格式。  
（3）gif是一种位图文件格式，以8位色重现真色彩的图像。可以实现动画效果时候

webp格式  
是谷歌在2010年推出的图片格式，压缩率只有jpg的2/3，大小比png小了45%，缺点是压缩的时间更久了   
。兼容性不好，目前谷歌和opera支持。

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

Cookie隔离问题，同一个网页,多个RemoteWebDriver会共享同一个Cookie。比如想要并行登陆并执   
行操作，这样是不行的。

style标签写在body后与body前有什么区别？

页面加载自上而下 当然是先加载样式。

JavaScript

介绍JavaScript的基本数据类型。

1. Number 数字类型  
2. String 字符串类型   
3. Boolean 布尔类型  
4. Function 函数  
5. Object 对象  
6. Null  
7. Undefined 没有定义类型

说说写JavaScript的基本规范？

1. 命名规范  
   1-1，javascript文件：js后缀为.js中，尽量减少html中的js代码。因为存在js代码会明显增加文件大   
   小，且不能对其进行缓存和压缩。  
   1-2，缩进单位是4个空格。避免使用tab键来缩进。因为tab始终没有统一tab长短标准，空格会增加文   
   件的大小，但是可以忽略。  
   1-3,每行长度不超过80个字符。  
   1-4。注释，让注释有意义  
   1-5，变量声明。所有变量必须在使用前通过var进行申明。  
   1-6，函数声明。所有函数在使用前进行声明。  
   1-7，命名。大小写字母，10个数字和下划线组成。

JavaScript原型，原型链 ? 有什么特点？

每个对象都会在其内部初始化一个属性，就是prototype(原型)，当我们访问一个对象的属性时，

如果这个对象内部不存在这个属性，那么他就会去prototype里找这个属性，这个prototype又会有自己的prototype，

于是就这样一直找下去，也就是我们平时所说的原型链的概念。

关系：instance.constructor.prototype = instance.\_\_proto\_\_

特点：

JavaScript对象是通过引用来传递的，我们创建的每个新对象实体中并没有一份属于自己的原型副本。当我们修改原型时，与之相关的对象也会继承这一改变。

当我们需要一个属性的时，Javascript引擎会先看当前对象中是否有这个属性， 如果没有的话，

就会查找他的Prototype对象是否有这个属性，如此递推下去，一直检索到 Object 内建对象。

JavaScript有几种类型的值？（堆：原始数据类型和 栈：引用数据类型），你能画一下他们的内存图吗？

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

Javascript如何实现继承？

：构造继承，原型继承，（实例继承，拷贝继承）。构造函数继承可以将构造函数的属性拷贝给实例   
（＊.call(this,[])）。但是缺点是无法实现函数复用。原型继承可以实现函数复用，但是所有实例共   
享一个属性，任意一个实例改变原型属性都会改变其它实例的属性值。推荐采用构造函数继承传递属性   
（拷贝），原型继承传递方法。

Javascript创建对象的几种方式？

1.对象字面量的方式，var obj={};{}是字面量，可以立即求值；2.用function来模拟无参的构造   
函数，再定义属性；3.用function模拟构造函数，利用this；function Rectangle(w,h)   
{this.width=w;this.height=h;} var rect=new Rectangle(4,8); 4.利用工厂方式（内置对象   
Object，本质是方法），var obj=new Object();；5.利用原型方式来创建；6.混合方式来创建。

Javascript作用链域?

当代码在一个环境中执行时，会创建变量对象的一个作用域链。如果是个函数，则将其活动对象作   
为变量对象。活动对象在最开始只包含一个arguments对象。而下一个变量对象则来自下一个包含环境   
。如此一直延续到全局执行环境，这种组织形式即为作用域链。内部函数可访问外部变量，外部变量无   
法访问内部函数。注意：js没有块级作用域，若要形成块级作用域，可通过（function（）｛｝）（）   
；立即执行的形式实现。

谈谈This对象的理解。

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

eval是做什么的？

计算某个字符串，并执行其中的js代码。eval（string），这样避免使用eval，不安全，耗性能。   
由json字符串转换成json对象可以使用eval，var obj=eval('('+ str +')')

什么是window对象? 什么是document对象?

window对象代表浏览器中打开的一个窗口。document对象代表整个html文档。实际上，document对   
象是window对象的对象属性。

null，undefined的区别？

null表示一个对象被定义了，但存放了空指针。undefined表示这个值不存在。typeof(null)--   
object;typeof(undefined)--undefined

写一个通用的事件侦听器函数(机试题)。

// 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;  
}  
};

[“1”, “2”, “3”].map(parseInt) 答案是多少？

为何返回不是 [1,2,3] 却是 [1,NaN,NaN]？原因：parseInt接收的是两个参数，map传递的是3个参   
数。

关于事件，IE与火狐的事件机制有什么区别？

IE为事件冒泡，Firefox同时支持事件捕获和冒泡事件。但并非所有浏览器都支持事件捕获。   
jQuery中使用event.stopPropagation()方法可阻止冒泡;（旧ie的方法 ev.cancelBubble = true;）

如何阻止冒泡？

什么是闭包（closure），为什么要用它？

闭包指的是一个函数可以访问另一个函数作用域中变量的函数。常见的构造方法，是在一个函数内   
部定义另外一个函数。内部函数可以引用外层的参数和变量；参数和变量不会被垃圾回收机制回收。注   
意，闭包的原理是作用域链，所以闭包访问的上级作用域中的变量是个对象，其值为其运算结束后的最   
后一个值。除非用立即执行函数来解决。

javascript 代码中的”use strict”;是什么意思 ?

使用它区别是什么？

除了正常模式运行外，ECMAscript添加了第二种运行模式：“严格模式”。  
区别：（1）消除js不合理，不严谨地方，减少怪异行为  
（2）消除代码运行的不安全之处，  
（3）提高编译器的效率，增加运行速度  
（4）为未来的js新版本做铺垫。

如何判断一个对象是否属于某个类？

使用instanceof 即if(a instanceof Person){alert('yes');}

new操作符具体干了什么呢?

：1、创建一个空对象，并且 this 变量引用该对象，同时还继承了该函数的原型。p.\_proto\_ =   
Base.prototype;  
2、属性和方法被加入到 this 引用的对象中。Base.call(p/this)  
3、新创建的对象由 this 所引用，并且最后隐式的返回 this 。

用原生JavaScript的实现过什么功能吗？

（1）显示和隐藏  
jq：  
$(el).show();  
$(el).hide();

js：  
el.style.display = '';  
el.style.display = 'none';

（2）淡入淡出  
jq：$(el).fadeIn();　　  
js：function fadeIn(el) {  
var opacity = 0;  
  
el.style.opacity = 0;  
el.style.filter = '';  
  
var last = +new Date();  
var tick = function() {  
opacity += (new Date() - last) / 400;  
el.style.opacity = opacity;  
el.style.filter = 'alpha(opacity=' + (100 \* opacity)|0 + ')';  
  
last = +new Date();  
  
if (opacity < 1) {  
(window.requestAnimationFrame && requestAnimationFrame(tick)) || setTimeout(tick,   
16);  
}  
};  
  
tick();  
}  
  
fadeIn(el);

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？

Object.hasOwnProperty(proName)：是用来判断一个对象是否有你给出名称的属性或对象。不过需   
要注意的是，此方法无法检查该对象的原型链中是否具有该属性，该属性必须是对象本身的一个成员。  
判断对象是否有某个特定的属性。必须用字符串指定该属性。（例如，o.hasOwnProperty("name")）。

对JSON的了解？

JSON中对象通过“{}”来标识，一个“{}”代表一个对象，如{“AreaId”:”123”}，对象的值是   
键值对的形式（key：value）。  
是js的一个严格的子集，一种轻量级的数据交换格式，类似于xml。数据格式简单，易于读写，占用带   
宽小。两个函数：JSON.parse(str)<检查代码 str－－js对象>;JSON.stringify(obj);eval('('＋json   
＋')')<不会检查代码，会直接执行>

[].forEach.call($$("

"),function(a){ a.style.outline="1px solid #"+(~~(Math.random()

(1<<24))).toString(16) }) 能解释一下这段代码的意思吗？

获取页面中所有的元素，然后遍历每个元素，为元素的style属性增加一个颜色边框。

js延迟加载的方式有哪些？

js的延迟加载有助与提高页面的加载速度。  
（1）使用setTimeout延迟方法的加载时间 $(function (){  
setTimeout('A()', 1000); //延迟1秒  
})  
（2）让js最后加载  
引入外部js脚本文件时，如果放入html的head中,则页面加载前该js脚本就会被加载入页面，而放入   
body中，则会按照页面从上倒下的加载顺序来运行javascript的代码~~~ 所以我们可以把js外部引入的   
文件放到页面底部，来让js最后引入，从而加快页面加载速度。  
（3）上述方法2也会偶尔让你收到Google页面速度测试工具的“延迟加载javascript”警告。所以这里   
的解决方案将是来自Google帮助页面的推荐方案。  
//这些代码应被放置在</body>标签前(接近HTML文件底部)  
<script type="text/javascript">  
function downloadJSAtOnload() {  
var element = document.createElement("script");  
element.src = "defer.js";  
document.body.appendChild(element);  
}  
if (window.addEventListener)  
window.addEventListener("load", downloadJSAtOnload, false);  
else if (window.attachEvent)  
window.attachEvent("onload", downloadJSAtOnload);  
else window.onload = downloadJSAtOnload;  
</script>  
使用此段代码的步骤：   
1）.复制上面代码  
2）.粘贴代码到HTML的标签前 (靠近HTML文件底部)  
3）.修改“defer.js”为你的外部JS文件名  
4）.确保你文件路径是正确的。例如：如果你仅输入“defer.js”，那么“defer.js”文件一定与HTML   
文件在同一文件夹下。

Ajax 是什么? 如何创建一个Ajax？

（1）ajax的全称：Asynchronous Javascript And XML。所谓异步，在这里简单地解释就是：向服   
务器发送请求的时候，我们不必等待结果，而是可以同时做其他的事情，等到有了结果它自己会根据设   
定进行后续操作，与此同时，页面是不会发生整页刷新的，提高了用户体验。  
（2）(1)创建XMLHttpRequest对象,也就是创建一个异步调用对象 （考虑IE6的兼容性，new   
XMLHttpRequest（））;  
(2)创建一个新的HTTP请求,并指定该HTTP请求的方法、URL及验证信息 （xhr.open   
("get"),"example.php",true）;  
(3)设置响应HTTP请求状态变化的函数  
(4)发送HTTP请求  
(5)获取异步调用返回的数据  
(6)使用JavaScript和DOM实现局部刷新  
注：http请求过程：1、建立tcp连接。2、web浏览器向web服务器发送请求指令。3、web浏览器发送请   
求头信息。4、web服务器应答。5、web服务器发送应答头信息。6、web服务器向浏览器发送数据。7、   
web服务器关闭tcp连接。

同步和异步的区别?

同步的概念是os中：不同进程协同完成某项工作而先后次序调整（通过阻塞、唤醒等方式），同步   
强调的是顺序性，谁先谁后。。异步不存在顺序性。同步：浏览器访问服务器，用户看到页面刷新，重   
新发请求，等请求完，页面刷新，新内容出现，用户看到新内容之后进行下一步操作。异步：浏览器访   
问服务器请求，用户正常操作，浏览器在后端进行请求。等请求完，也买你不刷新，新内容也会出现，   
永辉看到新内容。

如何解决跨域问题?

（1）同源策略：在同一个域名下或者同一域名不同文件夹，只有这两个是允许通信的。  
特别注意两点：  
第一，如果是协议和端口造成的跨域问题“前台”是无能为力的，  
第二：在跨域问题上，域仅仅是通过“URL的首部”来识别而不会去尝试判断相同的ip地址对应着两个   
域或两个域是否在同一个ip上。  
（2）参考，写的很不错：http://www.cnblogs.com/JChen666/p/3399951.html

页面编码和被请求的资源编码如果不一致如何处理？

若请求的资源编码，如外引js文件编码与页面编码不同。可根据外引资源编码方式定义为   
charset="utf-8"或"gbk"。

模块化开发怎么做？

模块化开发指的是在解决某一个复杂问题或者一系列问题时，依照一种分类的思维吧问题进行系统   
性的分解以之解决。模块化是一种处理复杂系统分解为代码结构更合理，可维护性更高的可管理的模块   
方式。对于软件行业：系统被分解为一组高内聚，低耦合的模块。  
（1）定义封装的模块  
（2）定义新模块对其他模块的依赖  
（3）可对其他模块的引入支持  
在JavaScript中出现了一些非传统模块开发方式的规范 CommonJS的模块规范，AMD（Asynchronous   
Module Definition），CMD（Common Module Definition）等

AMD是异步模块定义，所有的模块将被异步加载，模块加载不影响后边语句运行。

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

参考这个<http://blog.chinaunix.net/uid-26672038-id-4112229.html>  
解释：AMD 是 RequireJS 在推广过程中对模块定义的规范化产出。  
CMD 是 SeaJS 在推广过程中对模块定义的规范化产出。  
区别：  
1. 对于依赖的模块，AMD 是提前执行，CMD 是延迟执行。不过 RequireJS 从 2.0 开始，也改成   
可以延迟执行（根据写法不同，处理方式不同）。CMD 推崇 as lazy as possible.  
2. CMD 推崇依赖就近，AMD 推崇依赖前置。  
3. AMD 的 API 默认是一个当多个用，CMD 的 API 严格区分，推崇职责单一。

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

1，概念

requireJS是基于AMD模块加载规范，使用回调函数来解决模块加载的问题。

2，原理

requireJS是使用创建script元素，通过指定script元素的src属性来实现加载模块的。

3，特点

1. 实现js文件的异步加载，避免网页失去响应

2，管理模块之间的依赖，便于代码的编写和维护

4，项目优化

r.js 是基于requirejs模块化的基础上进一步的压缩和打包成一个js，请求数大大减少，便于优化

让你自己设计实现一个requireJS，你会怎么做？

核心是实现 js 的加载模块，维护 js 的依赖关系，控制好文件加载的先后顺序

谈一谈你对ECMAScript6的了解？

ES6新的语法糖，类，模块化等新特性

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

在我看来 ES6 新添加的 class 只是为了补充 js 中缺少的一些面向对象语言的特性，但本质上来说它只是一种语法糖，不是一个新的东西，其背后还是原型继承的思想。通过加入 class 可以有利于我们更好的组织代码。

在 class 中添加的方法，其实是添加在类的原型上的。

异步加载的方式有哪些？

1. onload时的异步加载

window.addEventLitener("load",loadHandler);

dunction loadHandler(){

var gs = document.createElement("script");

....

}

将script方法放在一个函数里面，然后在window的onload()方法里面执行

2. $(document).ready()

引入jquery

$(document).ready(function(){})

1

3.async

<script type="text/javascript" src="xxx.js" async></script>

1

async属性是HTML5新增属性，兼容性问题

async属性规定一旦脚本可用，则会异步执行

async属性仅适用于外部脚本

不能保证脚本按顺序执行

将在onload事件之前完成

4. defer

<script type="text/javascript" defer></script>

1

defer规定是否对脚本执行进行延迟，直到页面加载

兼容所有浏览器

如果脚本不会改变文档内容，可将defer属性加入到<script>标签中，以便加快处理文档的速度

可以确保所有设置了defer属性的脚本按顺序执行

5. es6模块type="module"属性

<script type="module" src="xxx.js"></script>

1

浏览器对于带有type="module"的<script>，都是异步加载，不会造成堵塞浏览器，即等整个页面渲染完，再执行模块脚本，等同于打开的<script>标签的defer属性

documen.write和 innerHTML的区别?

document.write和innerHTML

主要区别：document.write是直接将内容写入页面的内容流，会导致页面全部重绘，innerHTML将内容写入某个DOM节点，不会导致页面全部重绘

DOM操作——怎样添加、移除、移动、复制、创建和查找节点?

（1）创建新节点

         createDocumentFragment()    //创建一个DOM片段

         createElement()   //创建一个具体的元素

         createTextNode()   //创建一个文本节点

（2）添加、移除、替换、插入

         appendChild()

         removeChild()

         replaceChild()

         insertBefore() //在已有的子节点前插入一个新的子节点

（3）查找

         getElementsByTagName()    //通过标签名称

        getElementsByName()    //通过元素的Name属性的值(IE容错能力较强，会得到一个数组，其中包括id等于name值的)

        getElementById()    //通过元素Id，唯一性

.call() 和 .apply() 的含义和区别？

apply：调用一个对象的一个方法，用另一个对象替换当前对象。例如：B.apply(A, arguments);即A对象应用B对象的方法。

call：调用一个对象的一个方法，用另一个对象替换当前对象。例如：B.call(A, args1,args2);即A对象调用B对象的方法。

call 与 apply 的相同点：

方法的含义是一样的，即方法功能是一样的；

第一个参数的作用是一样的；

call 与 apply 的不同点：两者传入的列表形式不一样

call可以传入多个参数；

apply只能传入两个参数，所以其第二个参数往往是作为数组形式传入

数组和对象有哪些原生方法，列举一下？

Array.concat( ) 连接数组  
Array.join( ) 将数组元素连接起来以构建一个字符串  
Array.length 数组的大小  
Array.pop( ) 删除并返回数组的最后一个元素  
Array.push( ) 给数组添加元素  
Array.reverse( ) 颠倒数组中元素的顺序  
Array.shift( ) 将元素移出数组  
Array.slice( ) 返回数组的一部分  
Array.sort( ) 对数组元素进行排序  
Array.splice( ) 插入、删除或替换数组的元素  
Array.toLocaleString( ) 把数组转换成局部字符串  
Array.toString( ) 将数组转换成一个字符串  
Array.unshift( ) 在数组头部插入一个元素  
  
Object.hasOwnProperty( ) 检查属性是否被继承  
Object.isPrototypeOf( ) 一个对象是否是另一个对象的原型  
Object.propertyIsEnumerable( ) 是否可以通过for/in循环看到属性  
Object.toLocaleString( ) 返回对象的本地字符串表示  
Object.toString( ) 定义一个对象的字符串表示  
Object.valueOf( ) 指定对象的原始值

JS 怎么实现一个类。怎么实例化这个类

1. 构造函数法（this + prototype） -- 用 new 关键字 生成实例对象
   * 缺点：用到了 this 和 prototype，编写复杂，可读性差

function Mobile(name, price){

this.name = name;

this.price = price;

}

Mobile.prototype.sell = function(){

alert(this.name + "，售价 $" + this.price);

}

var iPhone7 = new Mobile("iPhone7", 1000);

iPhone7.sell();

1. Object.create 法 -- 用 Object.create() 生成实例对象
   * 缺点：不能实现私有属性和私有方法，实例对象之间也不能共享数据

var Person = {

firstname: "Mark",

lastname: "Yun",

age: 25,

introduce: function(){

alert('I am ' + Person.firstname + ' ' + Person.lastname);

}

};

var person = Object.create(Person);

person.introduce();

// Object.create 要求 IE9+，低版本浏览器可以自行部署：

if (!Object.create) {

　 Object.create = function (o) {

　　　 function F() {}

　　　 F.prototype = o;

　　　 return new F();

　　};

　}

1. 极简主义法（消除 this 和 prototype） -- 调用 createNew() 得到实例对象
   * 优点：容易理解，结构清晰优雅，符合传统的"面向对象编程"的构造

var Cat = {

age: 3, // 共享数据 -- 定义在类对象内，createNew() 外

createNew: function () {

var cat = {};

// var cat = Animal.createNew(); // 继承 Animal 类

cat.name = "小咪";

var sound = "喵喵喵"; // 私有属性--定义在 createNew() 内，输出对象外

cat.makeSound = function () {

alert(sound); // 暴露私有属性

};

cat.changeAge = function(num){

Cat.age = num; // 修改共享数据

};

return cat; // 输出对象

}

};

var cat = Cat.createNew();

cat.makeSound();

1. ES6 语法糖 class -- 用 new 关键字 生成实例对象

class Point {

constructor(x, y) {

this.x = x;

this.y = y;

}

toString() {

return '(' + this.x + ', ' + this.y + ')';

}

}

var point = new Point(2, 3);

### Javascript 如何实现继承？

1. 构造函数绑定：使用 call 或 apply 方法，将父对象的构造函数绑定在子对象上

function Cat(name,color){

　Animal.apply(this, arguments);

　this.name = name;

　this.color = color;

}

1. 实例继承：将子对象的 prototype 指向父对象的一个实例

Cat.prototype = new Animal();

Cat.prototype.constructor = Cat;

1. 拷贝继承：如果把父对象的所有属性和方法，拷贝进子对象

function extend(Child, Parent) {

　　　var p = Parent.prototype;

　　　var c = Child.prototype;

　　　for (var i in p) {

　　　 c[i] = p[i];

　　　}

　　　c.uber = p;

}

1. 原型继承：将子对象的 prototype 指向父对象的 prototype

function extend(Child, Parent) {

var F = function(){};

　F.prototype = Parent.prototype;

　Child.prototype = new F();

　Child.prototype.constructor = Child;

　Child.uber = Parent.prototype;

}

1. ES6 语法糖 extends：class ColorPoint extends Point {}

class ColorPoint extends Point {

constructor(x, y, color) {

super(x, y); // 调用父类的constructor(x, y)

this.color = color;

}

toString() {

return this.color + ' ' + super.toString(); // 调用父类的toString()

}

}

JavaScript中的作用域与变量声明提升？

avaScript 作用域：

* 在 Java、C 等语言中，作用域为 for 语句、if 语句或{}内的一块区域，称为作用域；
* 而在 JavaScript 中，作用域为 function(){}内的区域，称为函数作用域。

JavaScript 变量声明提升：

* 在 JavaScript 中，函数声明与变量声明经常被 JavaScript 引擎隐式地提升到当前作用域的顶部。
* 声明语句中的赋值部分并不会被提升，只有名称被提升
* 函数声明的优先级高于变量，如果变量名跟函数名相同且未赋值，则函数声明会覆盖变量声明
* 如果函数有多个同名参数，那么最后一个参数（即使没有定义）会覆盖前面的同名参数

如何编写高性能的Javascript？

遵循严格模式：”use strict”;

将js脚本放在页面底部，加快渲染页面;

将js脚本成组打包，减少请求;

使用非阻塞方式下载js脚本;

尽量使用局部变量来保存全局变量;

尽量减少使用闭包;

使用window对象属性方法时省略window;

尽量减少对象成员嵌套;

缓存DOM节点的访问;

通过避免使用eval和function()构造器;

给setTimeout()和setInterval()传递函数而不是字符作为参数;

尽量使用直接量创建对象和数组; 最小化重绘(repaint)和回流(reflow);

那些操作会造成内存泄漏？

内存泄漏指不用拥有或需要任何对象（数据）之后，它们仍然存在于内存中。

提示：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），或对该对象的唯一引用是循环的，那么该对象占用的内存立即被回收。

如果setTimeout的第一个参数使用字符串而非函数，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是，就会产生一个循环）等会造成内存泄漏。

JQuery的源码看过吗？能不能简单概况一下它的实现原理？

jQuery.fn的init方法返回的this指的是什么对象？为什么要返回this？

jquery中如何将数组转化为json字符串，然后再转化回来？

jQuery 的属性拷贝(extend)的实现原理是什么，如何实现深拷贝？

语法

$.extend( target [, object1 ] [, objectN ] )

指示是否深度合并

$.extend( [deep ], target, object1 [, objectN ] )

警告: 不支持第一个参数传递 false 。

参数

描述

deep

可选。 Boolean类型 指示是否深度合并对象，默认为false。如果该值为true，且多个对象的某个同名属性也都是对象，则该"属性对象"的属性也将进行合并。

target

Object类型 目标对象，其他对象的成员属性将被附加到该对象上。

object1

可选。 Object类型 第一个被合并的对象。

objectN

可选。 Object类型 第N个被合并的对象。

二、深拷贝代码（把extend函数的第一个参数设置为true）：

var p = {

"id":"007",

"name":"刘德华",

"wife":{

"id":"008",

"name":"刘德的妻子",

"address":{

"city":"北京",

"area":"海淀区"

}

}

}

//深度拷贝的方式（把extend函数的第一个参数设置为true）,把p的属性合并到jQuery（$）上

$.extend(true,$,p);

p.wife.name="张三疯";

p.wife.address.city="香港";

console.log($.id);

console.log($.wife.name);

console.log($.wife.address.city);

console.log(p);

三、浅拷贝代码（extend函数里不传入第一个参数，默认为false）：

//二、浅拷贝的方式（extend函数里不传入第一个参数，默认为false）,把p的属性合并到jQuery（$）上，但是共享引用类型的属性wife和address

var p = {

"id":"007",

"name":"刘德华",

"wife":{

"id":"008",

"name":"刘德的妻子",

"address":{

"city":"北京",

"area":"海淀区"

}

}

}

$.extend($,p);//或者 $.extend(p);

p.wife.name="张三疯";

p.wife.address.city="香港";

console.log($.id);

console.log($.wife.name);

console.log($.wife.address.city);

console.log(p);

jquery.extend 与 jquery.fn.extend的区别？

jQuery.extend()是为jQuery类添加类方法（静态方法），需要通过jQuery类来调用（直接使用 $.xxx 调用）；

jQuery.fn.extend()是为jQuery类添加成员函数（实例方法），所有jQuery实例都可以直接调用（需要使用 $().xxx 调用）。

jQuery 的队列是如何实现的？队列可以用在哪些地方？

jQuery 核心中，有一组队列控制方法，这组方法由 queue()/dequeue()/clearQueue() 三个方法组成，它对需要连续按序执行的函数的控制可以说是简明自如，主要应用于 animate () 方法，ajax 以及其他要按时间顺序执行的事件中。

先解释一下这组方法各自的含义。

queue(name,[callback]): 当只传入一个参数时，它返回并指向第一个匹配元素的队列（将是一个函数数组，队列名默认是 fx); 当有两个参数传入时，第一个参数还是默认为 fx 的的队列名，第二个参数又分两种情况，当第二个参数是一个函数时，它将在匹配的元素的队列最后添加一个函数。当第二个参数是一个函数数组时，它将匹配元素的队列用新的一个队列来代替（函数数组）. 可能，这个理解起来有点晕。

dequeue(name): 这个好理解，就是从队列最前端移除一个队列函数，并执行它。

clearQueue([queueName]): 这是 1.4 新增的方法。清空对象上尚未执行的所有队列。参数可选，默认为 fx. 但个人觉得这个方法没多大用，用 queue() 方法传入两个参数的第二种参数即可实现 clearQueue 方法。

现在，我们要实现这样一个效果，有标有 1 至 7 的数字方块，要求这七个方块自左到右依次下落；写好以后

；如果此时，你想调换一个某个的执行顺序，比如，你想让 5 落下后再开始下落 3, 或者新加 8 至 15 八个方块，怎么办？

重写吗？在里面小心冀冀的改吗？显然，我们需要另外一种列简明便捷的方法来实现这个效果，那就是 jQuery 队列控制方法。；

这样一来，看起来是不是简明多了。如何实现？

1. 新建一个数组，把动画函数依次放进去（这样更改顺序，新加动画是不是方便多了？);
2. 用 queue 将这组动画函数数组加入到 slideList 队列中；
3. 用 dequeue 取出 slideList 队列中第一个函数，并执行它；
4. 初始执行第一个函数。

至于 clearQueue() 方法，就不多说了，演示中停止按钮调用的就是 clearQueue() 方法，当然你还可以用 queue() 方法直接将现在的函数列队替换成 [] 空数组实现（个人比较推荐空数组替换。, 更直观）.

谈一下Jquery中的bind(),live(),delegate(),on()的区别？

bind(type,[data],fn) 为每个匹配元素的特定事件绑定事件处理函数

1. $("a").bind("click",function(){alert("ok");});

live(type,[data],fn) 给所有匹配的元素附加一个事件处理函数，即使这个元素是以后再添加进来的

1. $("a").live("click",function(){alert("ok");});

delegate(selector,[type],[data],fn) 指定的元素（属于被选元素的子元素）添加一个或多个事件处理程序，并规定当这些事件发生时运行的函数

1. $("#container").delegate("a","click",function(){alert("ok");})

**on(events,[selector],[data],fn)** 在选择元素上绑定一个或多个事件的事件处理函数

## 差别

* 1、.bind() 是直接绑定在元素上
* 2、.live() 则是通过冒泡的方式来绑定到元素上的。更适合列表类型的，绑定到 document DOM 节点上。和。bind() 的优势是支持动态数据。
* 3、.delegate() 则是更精确的小范围使用事件代理，性能优于。live()
* 4、.on() 则是最新的 1.9 版本整合了之前的三种方式的新事件绑定机制

JQuery一个对象可以同时绑定多个事件，这是如何实现的？

1. $ele.on('eventName', handle1);
2. $ele.on('eventName', handle2);
3. $ele.on('eventName', handle3);

其实 $ele 元素的 eventName 事件有一个处理函数数组 监听一次就往里面放一个 handle，数组是先进后出型的

也就是栈， 然后触发事件的时候一次执行

上面的监听相当于

1. $ele.eventHandle['eventName'] = [];
2. $ele.eventHandle['eventName'].push(handle1);
3. $ele.eventHandle['eventName'].push(handle2);
4. $ele.eventHandle['eventName'].push(handle3);

然后 $ele.trigger('eventName') 触发的时候， 从栈里面取出处理函数执行

1. while($ele.eventHandle['eventName'].length) {
2. handle = $ele.eventHandle['eventName'].pop();
3. handle();
4. }

最先监听的最后执行；

是否知道自定义事件。jQuery里的fire函数是什么意思，什么时候用？

* 事件即“发布 / 订阅”模式，自定义事件即“消息发布”，事件的监听即“订阅订阅”
* JS 原生支持自定义事件，示例：
  1. document.createEvent(type); // 创建事件
  2. event.initEvent(eventType, canBubble, prevent); // 初始化事件
  3. target.addEventListener('dataavailable', handler, false); // 监听事件
  4. target.dispatchEvent(e); // 触发事件

jQuery 里的 fire 函数用于调用 jQuery 自定义事件列表中的事件

1 种是把那个函数放到抄 ready 函数外面。

第 2 种是袭在 ready 函数里面加上 window.deleteStu = deleteStu; 即 zd:

1. $(function(){
2. function deleteStu(id){
3. alert(id);
4. }
5. window.deleteStu = deleteStu;
6. }

jQuery 是通过哪个方法和 Sizzle 选择器结合的？（jQuery.fn.find()进入Sizzle）

通过创建一个div元素，检测被传入的fn是否被当前浏览器支持  
function assert( fn ) {  
var div = document.createElement("div");  
　　//此处用try-catch的原因是：被传入的fn很有可能是会报错的。因为fn中用的方法或属性很可能不被当前浏览器所支持。  
try {  
　　　　 // 尝试执行被传入的方法，并将结果返回，如果出错则走catch，直接返回false（当前方法不被支持）  
return !!fn( div );  
} catch (e) {  
return false;  
} finally {  
// 如果被创建的div存在parentNode，则将当前div删除。  
　　　　 // 这里div看上去不可能有parentNode，因为感觉它没有被插入DOM树。但由于被传入的函数fn很有可能会将这个div插入DOM树，所有必须执行此操作。  
if ( div.parentNode ) {  
div.parentNode.removeChild( div );  
}

针对 jQuery性能的优化方法？

show slide animate 等频繁修改 dom 很耗性能，可采用 jquery.transit 插件等

使用单个 id 或 class 选择器当然也是优化点咯，元素选择器是真的会卡

每次调用 $() 其实都是生成一个超大的对象，还是尽量存为变量吧

用 jquery 写事件委托是最爽的，优点也是非常明显的

Jquery与jQuery UI有啥区别？

1）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

2）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；

3）jQuery本身注重于后台，没有漂亮的界面，而jQuery UI则补充了前者的不足，他提供了华丽的展示界面，使人更容易接受。既有强大的后台，又有华丽的前台。jQuery UI是jQuery插件，只不过专指由jQuery官方维护的UI方向的插件

JQuery的源码看过吗？能不能简单说一下它的实现原理？

## 1、闭包机制；

1. // 以下截取自 jquery 源码片段
2. (function( window, undefined ) {
3. /\* 源码内容 \*/
4. })( window );

上面这一小段代码来自于 1.9.0 当中 jquery 的源码，它是一个无污染的 JS 插件的标准写法，专业名词叫闭包。可以把它简单的看做是一个函数，与普通函数不同的是，这个函数没有名字，而且会立即执行

我们将里面的变量变成了局域变量，这不仅可以提高运行速度，更重要的是我们在引用 jquery 的 JS 文件时，不会因为 jquery 当中的变量太多，而与其它的 JS 框架的变量命名产生冲突；闭包中的变量声明没有污染到外面的全局变量；;

## 2、作为全局的一个属性；

1. window.jQuery = window.$ = jQuery;

这一句话将我们在闭包当中定义的 jQuery 对象导出为全局变量 jQuery 和 $，因此我们才可以在外部直接使用 jQuery 和 $。

window 是默认的 JS 上下文环境，因此将对象绑定到 window 上面，就相当于变成了传统意义上的全局变量，

## 3、最核心的功能，就是选择器；

首先我们进入 jquery 源码中，可以很容易的找到 jquery 对象的声明，看过以后会发现，原来我们的 jquery 对象就是 init 对象。

1. jQuery = function( selector, context ) {
2. return new jQuery.fn.init( selector, context, rootjQuery );
3. }

这里出现了 jQuery.fn 这样一个东西，它的由来可以在 jquery 的源码中找到，它其实代表的就是 jQuery 对象的原型。

1. jQuery.fn = jQuery.prototype;
2. jQuery.fn.init.prototype = jQuery.fn;

这两句话，第一句把 jQuery 对象的原型赋给了 fn 属性，第二句把 jQuery 对象的原型又赋给了 init 对象的原型。也就是说，init 对象和 jQuery 具有相同的原型，因此我们在上面返回的 init 对象，就与 jQuery 对象有一样的属性和方法。

我们不打算深究 init 这个方法的逻辑以及实现，但是我们需要知道的是，jQuery 其实就是将 DOM 对象加了一层包裹，而寻找某个或者若干个 DOM 对象是由 sizzle 选择器负责的，

jQuery 对象有很多的属性和方法；对于属性来说，我们最需要关注的只有一个属性，就是 [0] 属性，[0] 其实就是原生的 DOM 对象。

很多时候，我们在 jQuery 和 DOM 对象之间切换时需要用到 [0] 这个属性。

从截图也可以看出，jQuery 对象其实主要就是把原生的 DOM 对象存在了 [0] 的位置，并给它加了一系列简便的方法。

这个索引 0 的属性我们可以从一小段代码简单的看一下它的由来，下面是 init 方法中的一小段对 DOMElement 对象作为选择器的源码。

1. // Handle $(DOMElement)
2. if ( selector.nodeType ) {
3. /\* 可以看到，这里将 DOM 对象赋给了 jQuery 对象的 [0] 这个位置 \*/
4. this.context = this[0] = selector;
5. this.length = 1;
6. return this;
7. }

这一小段代码可以在 jquery 源码中找到，它是处理传入的选择参数是一个 DOM 对象的情况。

可以看到，里面很明显的将 jQuery 对象索引 0 的位置以及 context 属性，都赋予了 DOM 对象

。代码不仅说明了这一点，也同时说明了，我们使用 $(DOMElement) 可以将一个 DOM 对象转换为 jQuery 对象，从而通过转换获得 jQuery 对象的简便方法。

## 4、ready 方法

$(function(){}) 或者是 ready 方法；

实现类似 jquery 的 ready 方法的效果我们是可以简单做到的，它的实现原理就是，维护一个函数数组，然后不停的判断 DOM 是否加载完毕，倘若加载完毕就触发所有数组中的函数。

遵循着这一思想，LZ 拿出很久之前写的一个小例子，来给各位看一下。

1. (function( window, undefined ) {
2. var jQuery = {
3. isReady:false,// 文档加载是否完成的标识
4. readyList:[],// 函数序列
5. //onload 事件实现
6. ready : function(fn){
7. // 如果是函数，加入到函数序列
8. if(fn && typeof fn == 'function' ){
9. jQuery.readyList.push(fn);
10. }
11. // 文档加载完成，执行函数序列。
12. if(jQuery.isReady){
13. for(var i = 0;i < jQuery.readyList.length ;i++){
14. fn = jQuery.readyList[i];
15. jQuery.callback(fn);
16. }
17. return jQuery;
18. }
19. },
20. // 回调
21. callback : function(fn){
22. fn.call(document,jQuery);
23. }
24. };
25. // 导出对象
26. window.$ = window.jQuery = jQuery;
27. // 判断加载是否完成
28. var top = false;
29. try {
30. top = window.frameElement == null && document.documentElement;
31. } catch(e) {}
32. if ( top && top.doScroll ) {
33. (function doScrollCheck() {
34. try {
35. top.doScroll("left");
36. jQuery.isReady = true;
37. jQuery.ready();
38. } catch(e) {
39. setTimeout( doScrollCheck, 50 );
40. }
41. })();
42. }
43. }(window));

这段代码是 LZ 从之前的例子摘出来的，它的实现逻辑非常简单，但是可以达到 jQuery 的 ready 方法的效果，

## 5、extend 方法

简单说两个 extend 方法的常用方式。

* 1、使用 jQuery.fn.extend 可以扩展 jQuery 对象，使用 jQuery.extend 可以扩展 jQuery，前者类似于给类添加普通方法，后者类似于给类添加静态方法。
* 2、两个 extend 方法如果有两个 object 类型的参数，则会将后面的参数对象属性扩展到第一个参数对象上面，扩展时可以再添加一个 boolean 参数控制是否深度拷贝。

jquery 中如何将数组转化为json字符串，然后再转化回来？

## jquery 中如何将数组转化为 json 字符串，然后再转化回来？

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

然后调用：

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

jQuery和Zepto的区别？各自的使用场景？

1. jQuery 主要目标是PC的网页中，兼容全部主流浏览器。在移动设备方面，单独推出 `jQuery Mobile
2. Zepto从一开始就定位移动设备，相对更轻量级。它的API 基本兼容jQuery`，但对PC浏览器兼容不理想

针对 jQuery 的优化方法？

1.使用选择器

尽量选用id选择器和元素选择器:因为遇到这些选择器的时候,jquery会自动调用浏览器的原生方法比如(document.getElementById()),所以他们的执行速度快.

根据图中可以看出ID选择器执行速度最快,其次是元素选择器,然后是class.所以优化jquery尽量使用id选择器和元素选择器.

2.尽量不要使用 ".live()" 方法

这是jquery1.3.1版本之后新增加的方法,这个方法的功能就是为新增DOM元素绑定事件.但对于效率来说,这个方法比较占用资源

3.使用data()

尽量使用data()方法储存变量,因为这是jquery的内部函数 data()来存储状态.

Zepto的点透问题如何解决？

方案一：来得很直接github上有个fastclick可以完美解决https://github.com/ftlabs/fastclick

引入fastclick.js，因为fastclick源码不依赖其他库所以你可以在原生的js前直接加上

 window.addEventListener( "load", function() {

    FastClick.attach( document.body );

 }, false );

或者有zepto或者jqm的js里面加上

 $(function() {

     FastClick.attach(document.body);

});

当然require的话就这样：

var FastClick = require(‘fastclick‘);

FastClick.attach(document.body, options);

方案二：用touchend代替tap事件并阻止掉touchend的默认行为preventDefault()

$("#cbFinish").on("touchend", function (event) {

    //很多处理比如隐藏什么的

     event.preventDefault();

});

方案三：延迟一定的时间(300ms+)来处理事件

$("#cbFinish").on("tap", function (event) {

     setTimeout(function(){

     //很多处理比如隐藏什么的

     },320);

 });

理论上上面的方法可以完美的解决tap的点透问题，如果真的倔强到不行，用click

jQueryUI如何自定义组件?

1. 通过向 $.widget() 传递组件名称和一个原型对象来完成
2. $.widget(“ns.widgetName”, [baseWidget], widgetPrototype);

需求：实现一个页面操作不会整页刷新的网站，并且能在浏览器前进、后退时正确响应。给出你的技术实现方案？

通过使用 pushState + ajax 实现浏览器无刷新前进后退，当一次 ajax 调用成功后我们将一条 state 记录加入到 history 对象中。一条 state 记录包含了 url、title 和 content 属性，在 popstate 事件中可以获取到这个 state 对象，我们可 以使用 content 来传递数据。最后我们通过对 window.onpopstate 事件监听来响应浏览器的前进后退操作。

使用 pushState 来实现有两个问题，一个是打开首页时没有记录，我们可以使用 replaceState 来将首页的记录替换，另一个问 题是当一个页面刷新的时候，仍然会向服务器端请求数据，因此如果请求的 url 需要后端的配合将其重定向到一个页面。

如何判断当前脚本运行在浏览器还是node环境中？（阿里）

通过判断Global对象是否为window，如果不为window，则当前脚本运行在node.js环境中。

​​​​​​​<script>

this === window ? console.log('browser') : console.log('node');

</script>

移动端最小触控区域是多大？

44\*44px

jQuery 的 slideUp动画 ，如果目标元素是被外部事件驱动, 当鼠标快速地连续触发外部元素事件, 动画会滞后的反复执行，该如何处理呢?

1. 在触发元素上的事件设置为延迟处理：使用 JS 原生 setTimeout 方法
2. 在触发元素的事件时预先停止所有的动画，再执行相应的动画事件：$(’.tab’).stop().slideUp();

把 Script 标签 放在页面的最底部的body封闭之前 和封闭之后有什么区别？浏览器会如何解析它们？

html标签只包含head 和body两个标签，解析时，所有标签都会解析进这两个标签里边。body之前的任何位置都会解析进head里边，之后的都会解析进body里边。

移动端的点击事件的有延迟，时间是多久，为什么会有？ 怎么解决这个延时？（click 有 300ms 延迟,为了实现safari的双击事件的设计，浏览器要知道你是不是要双击操作。）

移动端 click 事件会有 300ms 的延时，原因是移动端屏幕双击会缩放(double tap to zoom) 页面。

解决方案：

### 1. 禁用缩放： 浏览器禁用默认的双击缩放行为并且去掉300ms 的点击延迟。

 <meta name="viewport" content="user-scalable=no">

### 2.利用touch事件自己封装这个事件解决300ms 延迟。

原理就是：

当我们手指触摸屏幕，记录当前触摸时间

当我们手指离开屏幕， 用离开的时间减去触摸的时间

如果时间小于150ms，并且没有滑动过屏幕， 那么我们就定义为点击

代码如下:

1. *//封装tap，解决click 300ms 延时*
2. function tap (obj, callback) {
3. var isMove = false;
4. var startTime = 0; *// 记录触摸时候的时间变量*
5. obj.addEventListener('touchstart', function (e) {
6. startTime = Date.now(); *// 记录触摸时间*
7. });
8. obj.addEventListener('touchmove', function (e) {
9. isMove = true; *// 看看是否有滑动，有滑动算拖拽，不算点击*
10. });
11. obj.addEventListener('touchend', function (e) {
12. if (!isMove && (Date.now() - startTime) < 150) { *// 如果手指触摸和离开时间小于150ms 算点击*
13. callback && callback(); *// 执行回调函数*
14. }
15. isMove = false; *// 取反 重置*
16. startTime = 0;
17. });
18. }
19. *//调用*
20. tap(div, function(){ *// 执行代码 });*

### 3、使用插件：fastclick 插件解决300ms 延迟。

GitHub官网地址： <https://github.com/ftlabs/fastclick>

FastClick 是 FT Labs 专门为解决移动端浏览器 300 毫秒点击延迟问题所开发的一个轻量级的库。简而言之，FastClick 在检测到touchend事件的时候，会通过 DOM 自定义事件立即触发一个模拟click事件，并把浏览器在 300 毫秒之后真正触发的click事件阻止掉。使用方法如下。

第一步：在页面中引入fastclick.js文件。

第二步：在js文件中添加以下代码      在 window load 事件之后，在body上调用FastClick.attach()即可。

1. window.addEventListener(function(){
2. FastClick.attach( document.body );
3. },false );

 如果你项目使用了JQuery，就将上面的代码改写成：

1. $(function() {
2. FastClick.attach(document.body);
3. });

### 4、使用 touchend 事件处理

不同于 click 和 touchstart,touchend 没有这个 300ms 的延迟,而是即时触发,你完全可以在 WebGL 或 canvase 游戏中进行实践,但是在web 页面中单单使用 touchend 并不一定靠谱.

1、如果用户在两个不同元素之间触发了 touchstart 和touchend，那么 click 不会被触发 .

2、如果用户触发了 touchstart,但是在touchend之前是一个长长的 touchmove 滚动, 那么 click 也不会被触发.

3、在站点上任然应该保留 click 事件以兼容那些非触摸设备,这是你就要考虑事件的重复触发问题.

在此处输入内容已有同行为我们封装了部分解决方法：

   1、FastClick 来至于FT实验室的一个插件，仅仅只有10kb，但是能解决上面的2-4步.

   2、Tappy 来自于Filament Group，仅仅1kb，概念上类似于上面的指针事件.问题：当你为多个元素进行这些事件绑定时，有可能出现性能的损耗.是否有完美的解决方案呢？

是否需要解决 300ms 在于自己的判断，300ms被设计出来是有特定的用途，上面的meta属性中进行了设定，在chrome和firefox下能起作用，希望其他厂商也能尽 快提供这类支持。touch-action: manipulation 这个css属性能把风险降到最低，虽然现在只有微软支持，但是作为W3C规范和HTML5test的一部分，因此也许我们并不需要等待太久。

知道各种JS框架(Angular, Backbone, Ember, React, Meteor, Knockout…)么? 能讲出他们各自的优点和缺点么?

Underscore 对哪些 JS 原生对象进行了扩展以及提供了哪些好用的函数方法？

Node.js的适用场景？

1.实时应用：如在线聊天，实时通知推送等等（如 socket.io）

2.分布式应用：通过高效的并行 I/O 使用已有的数据

3.工具类应用：海量的工具，小到前端压缩部署（如 grunt），大到桌面图形界面应用程

序

4.游戏类应用：游戏领域对实时和并发有很高的要求（如网易的 pomelo 框架）

5.利用稳定接口提升 Web 渲染能力

6.前后端编程语言环境统一：前端开发人员可以非常快速地切入到服务器端的开发（如著

名的纯 Javascript 全栈式 MEAN 架构）

(如果会用node)知道route, middleware, cluster, nodemon, pm2, server-side rendering么?

解释一下 Backbone 的 MVC 实现方式？

什么是“前端路由”?什么时候适合使用“前端路由”? “前端路由”有哪些优点和缺点?

知道什么是webkit么? 知道怎么用浏览器的各种工具来调试和debug代码么?

如何测试前端代码么? 知道BDD, TDD, Unit Test么? 知道怎么测试你的前端工程么(mocha, sinon, jasmin, qUnit..)?

前端templating(Mustache, underscore, handlebars)是干嘛的, 怎么用?

简述一下 Handlebars 的基本用法？

简述一下 Handlerbars 的对模板的基本处理流程， 如何编译的？如何缓存的？

用js实现千位分隔符?(来源：前端农民工，提示：正则+replace)

检测浏览器版本版本有哪些方式？

我们给一个dom同时绑定两个点击事件，一个用捕获，一个用冒泡，你来说下会执行几次事件，然后会先执行冒泡还是捕获

其他问题

**绑定在被点击元素的事件是按照代码顺序发生，其他元素通过冒泡或者捕获“感知”的事件，按照W3C的标准，先发生捕获事件，后发生冒泡事件。所有事件的顺序是：其他元素捕获阶段事件 -> 本元素代码顺序事件 -> 其他元素冒泡阶段事件 。**

原来公司工作流程是怎么样的，如何与其他人协作的？如何夸部门合作的？

你遇到过比较难的技术问题是？你是如何解决的？

设计模式 知道什么是singleton, factory, strategy, decrator么?

常使用的库有哪些？常用的前端开发工具？开发过什么应用或组件？

页面重构怎么操作？

页面重构就是根据原有页面内容和结构的基础上，通过div+css写出符合web标准的页面结构。

具体实现要达到以下三点：

功能不全页面的重构页面功能不符合用户体验、用户交互结构完整，可通过标准验证，

代码重构：代码质量、SEO优化、页面性能、更好的语义化、浏览器兼容、CSS优化

充分考虑到页面在站点中的“作用和重要性”，并对其进行有针对性的优化

列举IE与其他浏览器不一样的特性？

a. IE的排版引擎是Trident （又称为MSHTML）

b. Trident内核曾经几乎与W3C标准脱节（2005年）

c. Trident内核的大量 Bug等安全性问题没有得到及时解决

d. JS方面，有很多独立的方法，例如绑定事件的attachEvent、创建事件的createEventObject等

e. CSS方面，也有自己独有的处理方式，例如设置透明，低版本IE中使用滤镜的方式

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）  
泽尔德曼（Zeldman·J.），傅捷，王宗义，祝军 著  
内容简介  
《网站重构：应用Web标准进行设计（第2版）》与第1版保持同样的写作风格，作者用幽默诙谐语言，把复杂的事情讲得简单明了，引人入胜。书的核心依然是指导我们如何建立一个显示的速度更快、能访问的用户更多、设计和维护成本更低的网站。  
畅销书作者、网页设计师、Web标准的“传教士”Jeffrey Zeldman已经对他震撼业界的经典书籍《网站重构：应用Web标准进行设计（第1版）》进行了升级更新。新版（现在原版书是全彩的）增加了在第1版出版之后各浏览器的发展动态以及为适应标准化设计所做的最新改进。

什么叫优雅降级和渐进增强？

渐进增强 progressive enhancement：  
针对低版本浏览器进行构建页面，保证最基本的功能，然后再针对高级浏览器进行效果、交互等改进和追加功能达到更好的用户体验。

优雅降级 graceful degradation：  
一开始就构建完整的功能，然后再针对低版本浏览器进行兼容。

区别：

a. 优雅降级是从复杂的现状开始，并试图减少用户体验的供给

b. 渐进增强则是从一个非常基础的，能够起作用的版本开始，并不断扩充，以适应未来环境的需要

c. 降级（功能衰减）意味着往回看；而渐进增强则意味着朝前看，同时保证其根基处于安全地带

是否了解公钥加密和私钥加密。

在现代密码体制中加密和解密是采用不同的密钥（公开密钥），也就是非对称密钥密码系统，每个通信方均需要两个密钥，即公钥和私钥，这两把密钥可以互为加解密。公钥是公开的，不需要保密，而私钥是由个人自己持有，并且必须妥善保管和注意保密。公钥密码体制的核心思想是：加密和解密采用不同的密钥。这是公钥密码体制和传统的对称密码体制最大的区别。对于传统对称密码而言，密文的安全性完全依赖于 密钥的保密性，一旦密钥泄漏，将毫无保密性可言。但是公钥密码体制彻底改变了这一状况。在公钥密码体制中，公钥是公开的，只有私钥是需要保密的。知道公钥和密码算法要推测出私钥在计算上是不可行的。这样，只要私钥是安全的，那么加密就是可信的。

显然，对称密码和公钥密码都需要保证密钥的安全，不同之处在于密钥的管理和分发上面。在对称密码中，必须要有一种可靠的手段将加密密钥（同时也是解密密钥）告诉给解密方；而在公钥密码体制中，这是不需要的。解密方只需要保证自己的私钥的保密性即可，对于公钥，无论是对加密方而言还是对密码分析者而言都是公开的，故无需考虑采用可靠的通道进行密码分发。这使得密钥管理和密钥分发的难度大大降低了。

公钥私钥的原则：

一个公钥对应一个私钥。

密钥对中，让大家都知道的是公钥，不告诉大家，只有自己知道的，是私钥。

如果用其中一个密钥加密数据，则只有对应的那个密钥才可以解密。

如果用其中一个密钥可以进行解密数据，则该数据必然是对应的那个密钥进行的加密。

非对称密钥密码的主要应用就是公钥加密和公钥认证，而公钥加密的过程和公钥认证的过程是不一样的，下面我就详细讲解一下两者的区别。

WEB应用从服务器主动推送Data到客户端有那些方式？

Javascript数据推送

Commet：基于HTTP长连接的服务器推送技术

基于WebSocket的推送方案

SSE（Server-Send Event）：服务器推送数据新方式

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，

 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，

\*（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，

你有用过哪些前端性能优化的方法？

减少全局变量压缩css,js,图片减少dom操作

http状态码有那些？分别代表是什么意思？

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）

部分地区用户反应网站很卡，请问有哪些可能性的原因，以及解决方法？

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

除了前端以外还了解什么其它技术么？你最最厉害的技能是什么？

你用的得心应手用的熟练地编辑器&开发环境是什么样子？

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。

你怎么看待Web App 、hybrid App、Native App？

你移动端前端开发的理解？（和 Web 前端开发的主要区别是什么？）

你对加班的看法？

平时如何管理你的项目？

说说最近最流行的一些东西吧？常去哪些网站？

如何设计突发大规模并发架构？

说说最近最流行的一些东西吧？常去哪些网站？

是否了解开源的工具 bower、npm、yeoman、grunt、gulp，一个 npm 的包里的 package.json 具备的必要的字段都有哪些？（名称、版本号，依赖）

每个模块的代码结构都应该比较简单，且每个模块之间的关系也应该非常清晰，随着功能和迭代次数越来越多，你会如何去保持这个状态的？

Git知道branch, diff, merge么?

如何设计突发大规模并发架构？

当团队人手不足，把功能代码写完已经需要加班的情况下，你会做前端代码的测试吗？

说说最近最流行的一些东西吧？平时常去哪些网站？

知道什么是SEO并且怎么优化么? 知道各种meta data的含义么?

移动端（Android IOS）怎么做好用户体验?

简单描述一下你做过的移动APP项目研发流程？

你在现在的团队处于什么样的角色，起到了什么明显的作用？

你认为怎样才是全端工程师（Full Stack developer）？

介绍一个你最得意的作品吧？

你有自己的技术博客吗，用了哪些技术？

对前端安全有什么看法？

是否了解Web注入攻击，说下原理，最常见的两种攻击（XSS 和 CSRF）了解到什么程度？

项目中遇到国哪些印象深刻的技术难题，具体是什么问题，怎么解决？。

最近在学什么东西？

你的优点是什么？缺点是什么？

如何管理前端团队?

最近在学什么？能谈谈你未来3，5年给自己的规划吗？

有趣的问题

A、B两人分别在两座岛上。B生病了，A有B所需要的药。C有一艘小船和一个可以上锁的箱子。C愿意在A和B之间运东西，但东西只能放在箱子里。只要箱子没被上锁，C都会偷走箱子里的东西，不管箱子里有什么。如果A和B各自有一把锁和只能开自己那把锁的钥匙，A应该如何把东西安全递交给B？

答案：A把药放进箱子，用自己的锁把箱子锁上。B拿到箱子后，再在箱子上加一把自己的锁。

箱子运回A后，A取下自己的锁。箱子再运到B手中时，B取下自己的锁，获得药物。

Amazon主页的左上角有一个商品分类浏览的下拉菜单 没有延迟，而且子菜单也不会在不应该的时候消失。它是怎样做到这一点的呢？

答案是通过探测鼠标移动的方向和轨迹，具体查看Khan Academy工程师 Ben Kamens 写的 jQuery插件