## 滴滴面试

时间：2点开始，3.03结束

地点：文思海辉南楼五层大厅

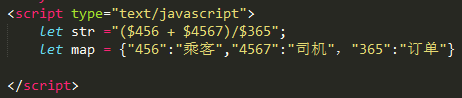
面试题目：

先做一下自我介绍

1.你做的项目主要涉及了那些工具？你的项目中遇到的问题有什么？展示你的作品(最好是多个作品，并且带着UI设计图)，某个功能怎么实现的，用了什么工具，你觉得你做的好的地方是什么？

2.了解ES6么？用过么？

3.



，怎么替换？例如将str中的“$456”替换成“乘客”，输出的结果应该是（乘客+司机）/订单。

提示：先把“$”后面的数字提取出来，排序，然后使用str的substr和replace从大到小替换。

4.js的基本数据类型有哪些？

5.ajax的基本过程，怎么声明？怎么请求，代码怎么写（手写），经过了那些过程？get和post有什么区别？post的机制？

6.数组的原生方法有哪些？怎么用push、pop、shift、unshift来模拟栈和队列？

7.js的基本类型怎么复制引用，object怎么引用？js中copy的原理是什么，怎么解除引用的？（深拷贝与浅拷贝问题）

8.事件流的捕捉和冒泡机制，怎么阻止向上冒泡？什么是事件委托，和冒泡有什么关系？

9.熟悉http的状态码么？列举几个，怎么用？

10．用过grunt、gulp、webpack么？怎么用？

11.你什么时候能入职？能做多久？什么时候毕业？

12．怎么理解地图下钻的功能？

13看过那些前端的书，什么时候开始接触前端。

面试经验：在介绍自己项目的过程中，最好详细的说做了什么，怎么做的，比如你写一个div布局，难点在哪里？怎么自适应？设计一个闪烁的小灯，怎么闪烁的，用了CSS3的box-shadow功能？等，说的越详细越好，让对方知道你有开发经验，他知道你做过，就不会深入这方面的问题。

说话要慢，不要紧张。要是有个点不懂，可以让面试官体系一下，但是不要放弃，能说多少是多少。

## 腾讯地图前端笔试题:

1用CSS实现上下左右居中

<style type="text/css">

body{

margin:0;/\*上右下左\*/

padding:0;

width:100%;

height:100%;

}

/\*上下左右居中，ok\*/

.box{

width: 100px;

height: 100px;

position:absolute;

left:0;

right:0;

top:0;

bottom:0;

margin:auto;

background-color: #3FAD72;

}

/\*上下居中\*/

. .box1{

position: absolute;

top: 50%;

height: 200px;

transform: translateY(-50%);/\*不知道高度的情况下可以用这一行代码\*/

/\*margin-top: -100px; negative half of the height 知道高度的情况下，可以替换上面哪一行\*/

background-color: #3FAD72;

} /\*左右居中\*/

.box2{

position: absolute;

left: 50%;

width: 200px;

margin-left: -100px; /\* negative half of the height \*/

background-color: #3FAD72;

}

</style>

2用观察者模式实现事件listener和trigger

https://www.cnblogs.com/aoguren/p/4680692.html

3介绍几种JS的跨域机制和原理

<http://blog.csdn.net/kongjiea/article/details/44201021>

4，找出字符串中出现次数最多的字符，JS写个函数

<script type="text/javascript">

function charNum(){

var str = "zhaochucichuzuiduodezifu";

var o = {};

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

var char = str.charAt(i);

if (o[char]) //char就是对象o的一个属性，o[char]是属性值，o[char]控制出现的次数

o[char]++; //次数加1

else

o[char] = 1; //若第一次出现，次数记为1

}

var max = 0;

var maxChar = null;

for (var key in o) {

if (max < o[key]) {

max = o[key]; //max始终储存次数最大的那个

maxChar = key; //那么对应的字符就是当前的key

}

}

console.log("最多的字符是" + maxChar);

console.log("出现的次数是" + max);

}

</script>

5用JS实现isprime，写个函数

<script type="text/javascript">

function isPrime(){

var num = 100;

for(var i=0;i<Math.sqrt(num);i++){

if(num%i == 0){

console.log('false');

return false;

}

}

console.log('true');

return true;

}

</script>

面试题：（一个小时）

6.threejs组成一个场景需要上面元素？

7.threejs中怎样让一个物体不受光照的影响。

方法1：设置环境光为白色，emmissive也为白色。

方法2：设置其他光线与物体的夹角为0.，怎么计算法向量？怎么计算夹角？

8.webGL中，生成一个模型，比如立方体，八个顶点各自都是三个面公用的，怎么渲染？

9.讲讲片元着色器和和顶点着色器，以及一个object的渲染过程。

10.怎么把屏幕坐标系转换为webgl坐标系？世界坐标系、本地坐标系、视觉坐标系之间的转换方法？还有什么坐标系？

11.threejs的raycaster的原理是什么，除了intersect[ ]数组之外还会返回那些值？

12.怎么优化3D页面的加载或者渲染速度？加载不出来怎么办？

13.讲一下自己了解的js设计模式。

14.webgl怎么生成一个模型？具体怎么实现？怎么定义坐标？index？着色器？

15.3D裁切是什么？

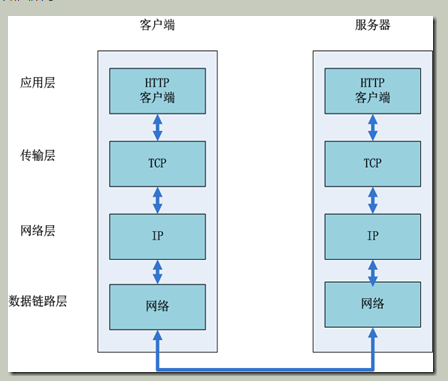
16.一个三维立方体其实都是由三角形拼起来的。

## 美团历年面试题目总结：

### 1.浏览器为什么不实现文件操作

因为浏览器不能访问本地资源的限制，现在的浏览器为了安全起见，一般都默认不能加载本地文件。

### 2.从键盘输入URL到网页呈现发生了什么



（web服务器、DNS、驱动、协议、CDN、代理、渲染引擎、DOMTree+CSSRule、JS引擎）

用户输入url，例如http://[www.baidu.com](http://www.baidu.com/)。其中http为协议，[www.baidu.com](http://www.baidu.com/)为网络地址，及指出需要的资源在那台计算机上。一般网络地址可以为域名或IP地址，此处为域名。使用域名是为了方便记忆，但是为了让计算机理解这个地址还需要把它解析为IP地址。

应用层DNS解析域名：客户端先检查本地是否有对应的IP地址，若找到则返回响应的IP地址。若没找到则请求上级DNS服务器，直至找到或到根节点。

应用层客户端发送HTTP请求：HTTP请求包括请求报头和请求主体两个部分，其中请求报头包含了至关重要的信息，包括请求的方法（GET / POST）、目标url、遵循的协议（http / https / ftp…），返回的信息是否需要缓存，以及客户端是否发送cookie等。

传输层TCP传输报文

 网络层IP协议查询MAC地址：IP协议的作用是把TCP分割好的各种数据包传送给接收方。而要保证确实能传到接收方还需要接收方的MAC地址，也就是物理地址。IP地址和MAC地址是一一对应的关系，一个网络设备的IP地址可以更换，但是MAC地址一般是固定不变的。ARP协议可以将IP地址解析成对应的MAC地址。当通信的双方不在同一个局域网时，需要多次中转才能到达最终的目标，在中转的过程中需要通过下一个中转站的MAC地址来搜索下一个中转目标。

数据到达数据链路层：在找到对方的MAC地址后，就将数据发送到数据链路层传输。这时，客户端发送请求的阶段结束

服务器接收数据：接收端的服务器在链路层接收到数据包，再层层向上直到应用层。这过程中包括在运输层通过TCP协议讲分段的数据包重新组成原来的HTTP请求报文。

服务器响应请求：服务接收到客户端发送的HTTP请求后，查找客户端请求的资源，并返回响应报文，响应报文中包括一个重要的信息——状态码。状态码由三位数字组成，其中比较常见的是200 OK表示请求成功。301表示永久重定向，即请求的资源已经永久转移到新的位置。在返回301状态码的同时，响应报文也会附带重定向的url，客户端接收到后将http请求的url做相应的改变再重新发送。404 not found 表示客户端请求的资源找不到。

服务器返回相应文件：请求成功后，服务器会返回相应的HTML文件。接下来就到了页面的渲染阶段了。

页面渲染

   现代浏览器渲染页面的过程是这样的： 解析HTML以构建DOM树 –> 构建渲染树 –> 布局渲染树 –> 绘制渲染树。

   DOM树是由HTML文件中的标签排列组成，渲染树是在DOM树中加入CSS或HTML中的style样式而形成。渲染树只包含需要显示在页面中的DOM元素，像<head>元素或display属性值为none的元素都不在渲染树中。

   在浏览器还没接收到完整的HTML文件时，它就开始渲染页面了，在遇到外部链入的脚本标签或样式标签或图片时，会再次发送HTTP请求重复上述的步骤。在收到CSS文件后会对已经渲染的页面重新渲染，加入它们应有的样式，图片文件加载完立刻显示在相应位置。在这一过程中可能会触发页面的重绘或重排。

### 3.事件模型

事件流：包括事件捕捉、事件触发、事件冒泡三个阶段。

事件处理程序（模型）：

HTML事件处理程序：<input onclick=function>

DOM0级事件处理程序: document.getElementById('click').onclick = function(event){ }

DOM2级事件处理程序:

var click = document.getElementById('inner');

click.addEventListener('click',function(){

alert('click one');

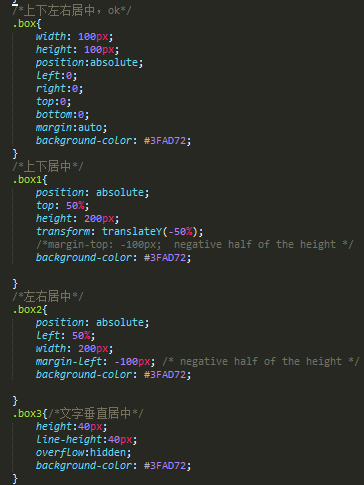
},false);

//要处理的事件名、处理函数、true在捕获阶段调用事件处理函数，false在冒泡阶段调用

### 4.CSS3新特性

* border-radius
* box-shadow
* border-image
* background-size
* background-origin
* text-shadow
* word-wrap
* @font-face
* 2D转换：translate()
  + rotate()
  + scale()
  + skew()
  + matrix()
  + 
* 3D转换
* Transition
* @keyframes
* animation: myfirst 5s linear 2s infinite alternate;
* column-count
* column-gap
* column-rule
* resize
* box-sizing
* outline-offset

### 5.CSS绝对居中



### 6.TCP/IP一定要三次握手吗

1. 主机1向主机2发送连接请求（connect request：CR），并携带一个主机1的初始序列号（CR（seq = x））
2. 主机2收到主机1的请求，返回一个确认请求（acknowledge：ACK），携带自己的初始序列号（SYN （ACK = x + 1, seq = y）） 表示告诉主机1，你可以发送序列号x + 1 的包了，我的初始序列号为y
3. 主机1收到了这个SYN包，开始将data打包，并携带（SYN（ACK = y + 1，seq = x + 1））信息，表示告诉主机2，我现在发送序列号为x + 1的包，你可以发送y + 1的包了。

一定要

### 7.对除get、post其他请求类型的了解

OPTIONS：返回服务器针对特定资源所支持的HTTP请求方法。也可以利用向Web服务器发送'\*'的请求来测试服务器的功能性。   
HEAD就像GET，只不过服务端接受到HEAD请求后只返回响应头，而不会发送响应内容。当我们只需要查看某个页面的状态的时候，使用HEAD是非常高效的，因为在传输的过程中省去了页面内容。

GET：向特定的资源发出请求。   
POST：向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST请求可能会导致新的资源的创建和/或已有资源的修改。   
PUT：向指定资源位置上传其最新内容。   
DELETE：请求服务器删除Request-URI所标识的资源。   
TRACE：回显服务器收到的请求，主要用于测试或诊断。

CONNECT：HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。

   虽然HTTP的请求方式有8种，但是我们在实际应用中常用的也就是get和post，其他请求方式也都可以通过这两种方式间接的来实现。

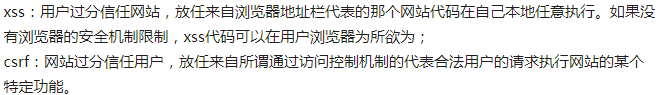
在web程序的设计原则上，GET传递参数的操作，不应该改变程序的内部结构，主要用于查询信息的过滤。对于数据库的更删改操作，一定要使用POST方式传值。

### 8.前端缓存

### 9.跨域

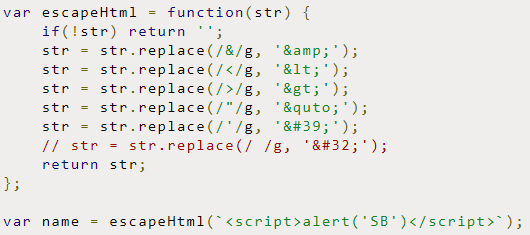
### 10.xss、csrf攻击及防范

XSS（（Cross-site scripting））成因概括 ：XSS定义的主语是“脚本”，是一种跨站执行的脚本，也就是javascript脚本，指的是在网站上注入我们的javascript脚本，执行非法操作。 XSS其实就是Html的注入问题，攻击者的输入没有经过严格的控制进入了数据库，最终显示给来访的用户，导致可以在来访用户的浏览器里以浏览用户的身份执行Html代码，数据流程如下：攻击者的Html输入—>web程序—>进入数据库—>web程序—>用户浏览器。



防御：

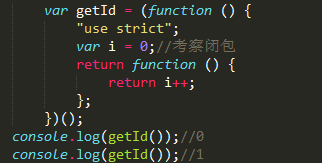
（1）对内容进行转义和过滤。（escape和unescape存在的意义）



（2）内容安全策略，Content Security Policy，简称CSP。在输出html时，加上Content Security Policy的Http Header。  
（3）CSRF：Cross Site Request Foggy，即跨站请求攻击，可以通过判断来源HTTP Referer、使用post、使用验证码、为每个表单添加令牌token并验证等方式防御。  
（1. 用户访问某个表单页面。2. 服务端生成一个Token，放在用户的Session中，或者浏览器的Cookie中。【这里已经不考虑XSS攻击】3. 在页面表单附带上Token参数。4. 用户提交请求后， 服务端验证表单中的Token是否与用户Session（或Cookies）中的Token一致，一致为合法请求，不是则非法请求。）

### 11.快速排序

### 12.一个函数，每次调用输出值自增1，2，3



### 13.http状态码 301、302区别

301 Moved Permanently 被请求的资源已永久移动到新位置，并且将来任何对此资源的引用都应该使用本响应返回的若干个URI之一。如果可能，拥有链接编辑功能的客户端应当自动把请求的地址修改为从服务器反馈回来的地址。除非额外指定，否则这个响应也是可缓存的。301比较常用的场景是使用域名跳转。

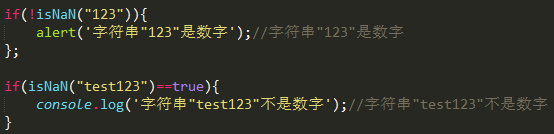
302 Found 请求的资源现在临时从不同的URI响应请求。由于这样的重定向是临时的，客户端应当继续向原有地址发送以后的请求。只有在Cache-Control或Expires中进行了指定的情况下，这个响应才是可缓存的。

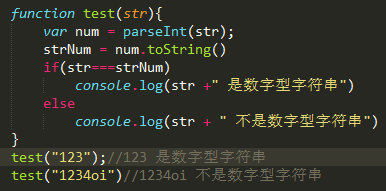
字面上的区别就是301是永久重定向，而302是临时重定向。 当然，他们之间也是有共同点的，就是用户都可以看到url替换为了一个新的，然后发出请求。

**302重定向只是暂时的重定向，搜索引擎会抓取新的内容而保留旧的地址，因为服务器返回302，所以，搜索搜索引擎认为新的网址是暂时的。**

**而301重定向是永久的重定向，搜索引擎在抓取新的内容的同时也将旧的网址替换为了重定向之后的网址。**

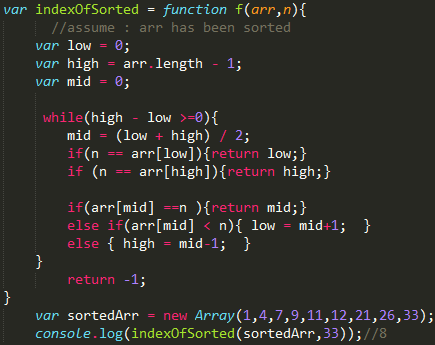
### 14.判断字符串是不是数字型字符串





### 15.一个有序数组，输出指定值（可能重复）的下标，计算时间、空间复杂度

//用折半法查找，时间复杂度为O(logn)，空间复杂度O（1）



### 17.你所知道的DOM元素操作方法（获取元素或者操作元素），尽可能多；

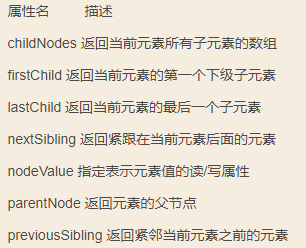


removeAttribute(name)－－等于attribute.removeNamedItem(name)。

nodeName String 节点的名字;根据节点的类型而定义

nodeValue String 节点的值;根据节点的类型而定义

nodeType Number 节点的类型常量值之一



### 18.你所知道的DOM事件；



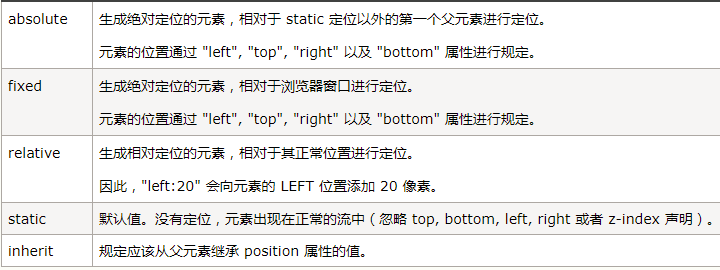
### 19.跨浏览器的事件模型

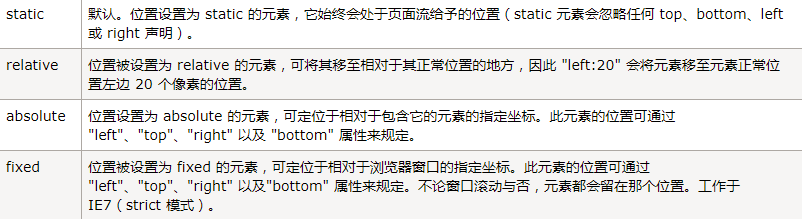
### 20.事件代理（事件委托）

事件委托就是利用事件冒泡，只指定一个事件处理程序，就可以管理某一类型的所有事件。

事件委托是利用事件的冒泡原理来实现的，何为事件冒泡呢？就是事件从最深的节点开始，然后逐步向上传播事件，举个例子：页面上有这么一个节点树，div>ul>li>a;比如给最里面的a加一个click点击事件，那么这个事件就会一层一层的往外执行，执行顺序a>li>ul>div，有这样一个机制，那么我们给最外面的div加点击事件，那么里面的ul，li，a做点击事件的时候，都会冒泡到最外层的div上，所以都会触发，这就是事件委托，委托它们父级代为执行事件。

### 21.position属性介绍





### 22.display属性介绍



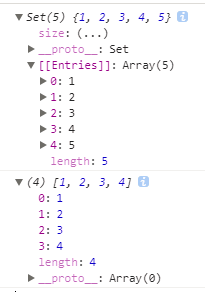
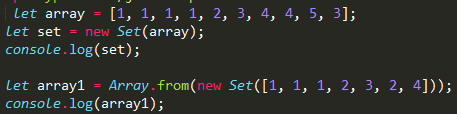
一个元素被设置为绝对定位时，会脱离文档流，然后相对其包含块进行偏移。

### 23.BFC，实现方式（清除浮动）

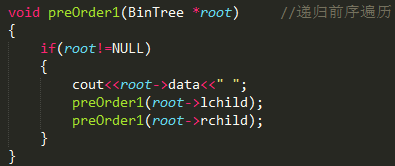
BFC(Block formatting context)直译为"块级格式化上下文"。它是一个独立的渲染区域，只有Block-level box参与， 它规定了内部的Block-level Box如何布局，并且与这个区域外部毫不相干。

### 24.数组去重（提到利用ES6的Set对象去重）

ES6中新增了Set数据结构，类似于数组，但是 它的成员都是唯一的 ，其构造函数可以接受一个数组作为参数，如：



### 25.二叉树前序遍历思路



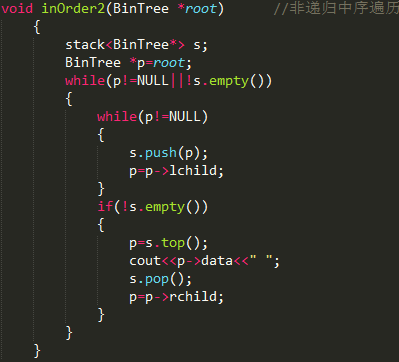
　　对于任一结点P：

     1)访问结点P，并将结点P入栈;

     2)判断结点P的左孩子是否为空，若为空，则取栈顶结点并进行出栈操作，并将栈顶结点的右孩子置为当前的结点P，循环至1);若不为空，则将P的左孩子置为当前的结点P;

     3)直到P为NULL并且栈为空，则遍历结束。





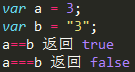
https://www.cnblogs.com/SHERO-Vae/p/5800363.html

### 26.==和===

”==”与”===”是不同的,一个是判断值是否相等,一个是判断值及类型是否完全相等。

==， 两边值类型不同的时候，要先进行类型转换，再比较。

==，不做类型转换，类型不同的一定不等。

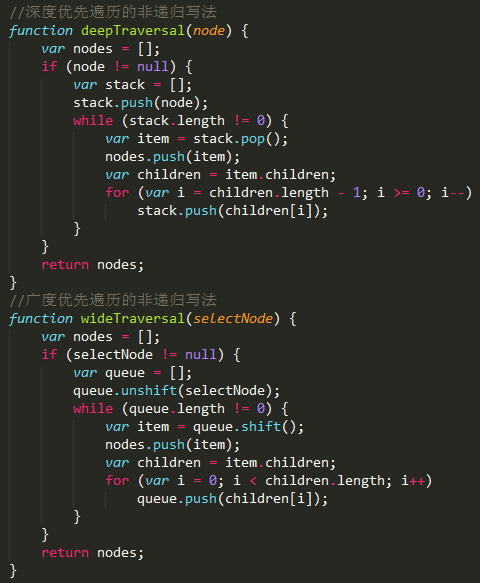


### 27.清除浮动的原理方法

**浮动概念**

浮动元素会脱离文档的普通流,根据float的值向左或向右移动,直到它的外边界**碰到父元素的内边界或另一个浮动元素的外边界为止**。由于浮动框不在文档的普通流中,所以文档的普通流中的块级元素表现得就像浮动元素不存在一样。

### 29.树的广度和深度优先遍历



### 30.进程和线程的区别

* 进程是资源分配的最小单位，线程是程序执行的最小单位。
* 进程有自己的独立地址空间，每启动一个进程，系统就会为它分配地址空间，建立数据表来维护代码段、堆栈段和数据段，这种操作非常昂贵。而线程是共享进程中的数据的，使用相同的地址空间，因此CPU切换一个线程的花费远比进程要小很多，同时创建一个线程的开销也比进程要小很多。
* 线程之间的通信更方便，同一进程下的线程共享全局变量、静态变量等数据，而进程之间的通信需要以通信的方式（IPC)进行。不过如何处理好同步与互斥是编写多线程程序的难点。
* 但是多进程程序更健壮，多线程程序只要有一个线程死掉，整个进程也死掉了，而一个进程死掉并不会对另外一个进程造成影响，因为进程有自己独立的地址空间。

### 31.应用层协议

(1)域名系统(Domain Name System，[DNS](https://baike.baidu.com/item/DNS/427444))：用于实现网络设备名字到IP地址映射的网络服务。53

(2)文件传输协议(File Transfer Protocol，FTP)：用于实现交互式文件传输功能。

(3)简单邮件传送协议(Simple Mail Transfer Protocol, SMTP)：用于实现电子邮箱传送功能

(4)超文本传输协议(HyperText Transfer Protocol，[HTTP](https://baike.baidu.com/item/HTTP))：用于实现WWW服务。

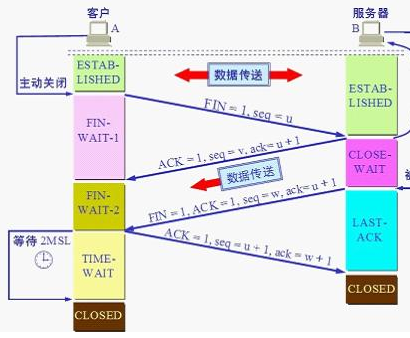
(5)简单网络管理协议(simple Network Management Protocol，SNMP)：用于管理与监视网络设备。161

(6)远程登录协议(Telnet)：用于实现远程登录功能。

应用层的协议用于网络进程间相互通信，分为响应报文和请求报文。

32.TCP为什么3次握手，每个阶段做什么事情，和UDP区别





[UDP](https://www.baidu.com/s?wd=UDP&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1dWnhwhn1f3ry7-mHFBmhub0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHm3rjbkn1f3)是一个[无连接协议](https://www.baidu.com/s?wd=%E6%97%A0%E8%BF%9E%E6%8E%A5%E5%8D%8F%E8%AE%AE&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1dWnhwhn1f3ry7-mHFBmhub0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHm3rjbkn1f3)，传输数据之前源端和终端不建立连接，当它想传送时就简单地去抓取来自应用程序的数据，并尽可能快地把它扔到网络上。也就是说，没有发送方和接收方之间连接建立过程，直接发。使用UDP协议包括：[TFTP](https://baike.baidu.com/item/TFTP)、[SNMP](https://baike.baidu.com/item/SNMP)、NFS、DNS、BOOTP

### 33.FTP介绍下

FTP 是 TCP/IP 协议组中的协议之一，是英文File Transfer Protocol的缩写。

### 34.页面解析顺序

1. 用户输入网址（假设是个 HTML 页面，并且是第一次访问），浏览器向服务器发出请求，服务器返回 HTML 文件；

2. 浏览器开始载入 HTML 代码，发现 <head> 标签内有一个 <link> 标签引用外部 CSS 文件；

3. 浏览器又发出 CSS 文件的请求，服务器返回这个 CSS 文件；

4. 浏览器继续载入 HTML 中 <body> 部分的代码，并且 CSS 文件已经拿到手了，可以开始渲染页面了；

5. 浏览器在代码中发现一个 <img> 标签引用了一张图片，向服务器发出请求。此时浏览器不会等到图片下载完，而是继续渲染后面的代码；

6. 服务器返回图片文件，由于图片占用了一定面积，影响了后面段落的排布，因此浏览器需要回过头来重新渲染这部分代码；

7. 浏览器发现了一个包含一行 JavaScript 代码的 <script> 标签，赶快运行它；

8. JavaScript 脚本执行了这条语句，它命令浏览器隐藏掉代码中的某个 <div>（style.display=”none”）。杯具啊，突然就少了这么一个元素，浏览器不得不重新渲染这部分代码；

9. 终于等到了 </html> 的到来，浏览器泪流满面……

10. 等等，还没完，用户点了一下界面中的“换肤”按钮，JavaScript 让浏览器换了一下 <link> 标签的 CSS 路径；

11. 浏览器召集了在座的各位 <div><span><ul><li> 们，“大伙儿收拾收拾行李，咱得重新来过……”，浏览器向服务器请求了新的CSS文件，重新渲染页面。 浏览器每天就这么来来回回跑着，要知道不同的人写出来的 HTML 和 CSS 代码质量参差不齐，说不定哪天跑着跑着就挂掉了。好在这个世界还有这么一群人——页面重构工程师，平时挺不起眼，也就帮视觉设计师们切切图啊改改字，其实背地里还是干了不少实事的。

所以把css和js文件作为外部文件是为了一次性加载渲染，防止多次加载导致页面重构，多次渲染。

**如何加快HTML页面加载速度**

 1、页面减肥、删除不必要的空格注释，多引用外部的js和css

2.js和css最好多做合并，减少页面上引用的文件数量

3.减少域名查询，减少对不同域名下资源的引用。

4.应用缓存，对重复使用的数据进行缓存。

### 35.css的选择器及其权重

一般来说，行间>内部>外部样式；ID>class>元素

!important 的作用是提升优先级，换句话说。加了这句的样式的优先级是最高的（比内联样式的优先级还高)。

第一等：代表内联样式，如: style=””，权值为1000。

第二等：代表ID选择器，如：#content，权值为0100。

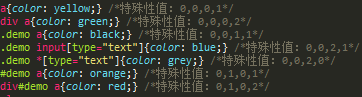
第三等：代表类，伪类和属性选择器，如.content，权值为0010。

第四等：代表类型选择器和伪元素选择器，如div p，权值为0001。

通配符、子选择器、相邻选择器等的。如\*、>、+,权值为0000。

继承的样式没有权值。

**1,0,0,0 > 0,99,99,99**

**比如：**

### 36.css的行内标签在DOM渲染时是怎么解析的

解析html以构建dom树 -> 构建render树 -> 布局render树 -> 绘制render树

浏览器是解析DOM生成DOM Tree，结合CSS生成的CSS Tree，最终组成render tree，再渲染页面。由此可见，在此过程中CSS完全无法影响DOM Tree，因而无需阻塞DOM解析。

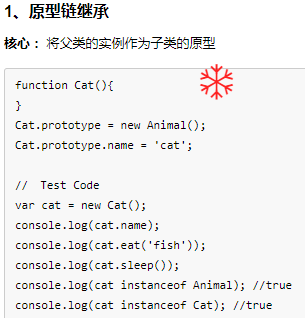
### 37.判断两个字符串是否互相包含的算法

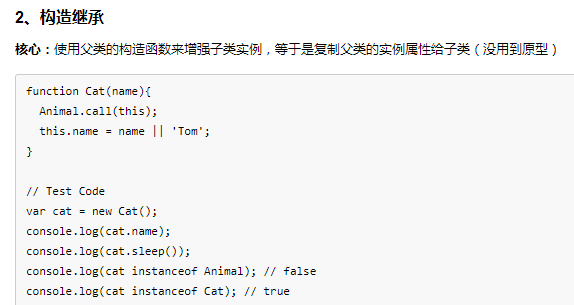


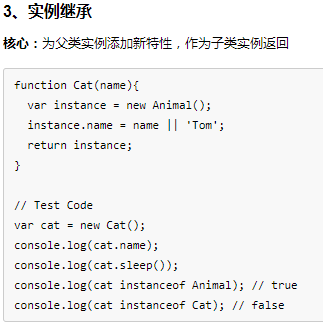
### 38.全排列算法

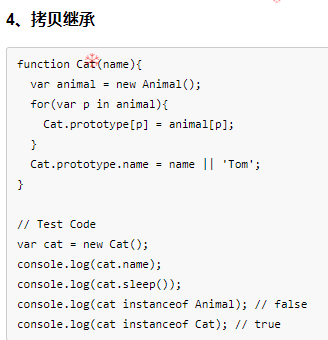
牛客网的permutation

### 39.js除了原型继承还有哪些其他的继承方式？



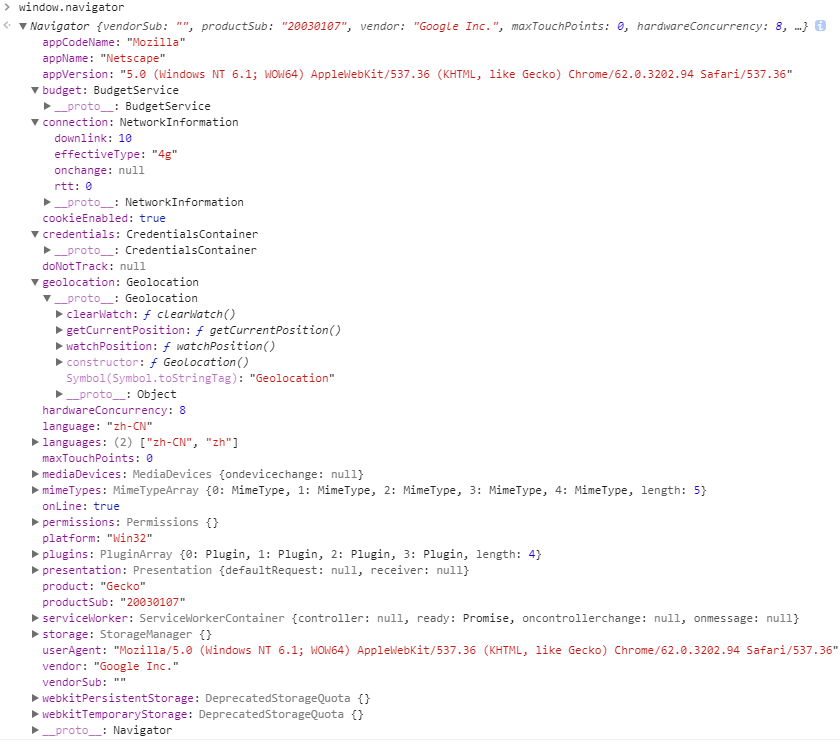






### 40.对主流框架了解多少，比如Vue、React等等？

### 41.怎么查看浏览器类型，是Chrome啊还是别的浏览器？





### 42.有没有移动端开发经验？（老实回答没有）

### 43.移动端开发和PC端开发有什么区别？

1.pc端没有触摸的各种事件，移动端有；移动端没有鼠标移入移出事件，但是pc端有

2.Pc端最常用的布局是固定宽度为980px（也有960px，1000px，1200px）；而在移动端，因为有很多网页是可以横屏看也可以竖屏看，并且很多屏幕的饿分辨率都是不一样的，所以只要牵涉到移动端都要考虑用响应式布局

3.Pc布局考虑更多的是浏览器的兼容性，但是在移动端，考虑更多的是手机兼容性，因为手机的型号类型非常的多，导致屏幕的分辨率是不一样的

4.移动端触摸弹出的手机键盘处理也是在pc端不会遇到的问题。

5.移动端和pc端适用的js框架也是不一样

6.Pc端浏览器内核有很多，webkit，Gecko，。。。。，但是移动端的浏览器内核比较单一，基本都是webkit以及webkit相关的blink

7.Pc端的交互是鼠标，键盘的交互而移动端变成了触摸，手势的交互

### 44.在移动端怎样做到自适应布局？有哪些方法？（左侧固定大小，右侧自适应的情况）

### 45.怎样设置两列布局？（卡在这个问题上了，心痛）

46.怎样设置元素隐藏的效果？

1、使用display:none来隐藏所有内容

2、使用visibility:hidden来隐藏内容

3、使用overflow:hidden隐藏溢出内容

### 47.CSS定位的方式有哪些？

CSS 有三种基本的定位机制：普通流、浮动和绝对定位

### 48.描述一下CSS盒模型（谈到box-sizing属性）

### 49..假如有海量的IP请求，怎样对这些IP地址进行处理可以找到请求次数最多的IP。

用ES6的map计算出现的次数。

### 50是否有了解跨域？jsonp只能用get方法吗？

所以JSONP的实现原理就是创建一个script标签, 再把需要请求的api地址放到src里. 这个请求只能用GET方法, 不可能是POST

51如何做到点击链接后从服务器下载文件?(项目相关)

52。mysql中Innodb和MyISAM的区别和应用场景？

### 53.在head和body中写js脚本的区别？

在HTML body部分中的JavaScripts会在页面加载的时候被执行。

在HTML head部分中的JavaScripts会在被调用的时候才执行。

浏览器解析html是从上到下的。

如果把javascript放在head里的话，则先被解析,但这时候body还没有解析，所以会返回空值。一般都会绑定一个监听，当全部的html文档解析完之后，再执行代码：

windows.onload=function(){

//这里放入执行代码

}

放在head中的JS代码会在页面加载完成之前就读取，而放在body中的JS代码，会在整个页面加载完成之后读取。

这就说明了，如果我们想定义一个全局对象，而这个对象是页面中的某个按钮时，我们必须将其放入body中，道理很明显：如果放入head，那当你定义的时候，那个按钮都没有被加载，可能获得的是一个undefind。

### 54.遍历数组有哪些方法，区别是啥?

第一种最常用的：for循环

 for(j = 0; j < arr.length; j++) { }

 优化版for循环

for(j = 0,len=arr.length; j < len; j++) {

}

 对于数组较大时，优化比较明显；

 第二种：foreach

var arr=[1,2,3,4,5,6];

arr.forEach(function(v,i){//v==value　为arr项，i==index　为arr索引

console.log(i+' 'v );

})

第三种：for ……in

var arr = new Array("first", "second", "third")

for(var item in arr) {

document.write(arr[item]+",");

}

   for……in 的效率比较低

第四种：map 遍历

arr.map(function(n){

});

第五种：for……of 遍历（需要ES6支持）

var arr=[1,2,3,4,5,6];

for(var value of Arr){

console.log(value);

}

for、map是比较常用的两种方法，性能也还行。

55.谈谈对web安全问题的理解？

56.304状态码的理解？

57.怎么实现登录时"记住我"的功能？

### 58.获取页面元素位置与宽高？

element.clientWidth = content + padding

element.clientHeight = content + padding

element.getBoundingClientRect() 返回值情况

left:包围盒左边 border 以外的边缘距页面左边的距离

right:包围盒右边 border 以外的边缘距页面左边的距离

top:包围盒上边 border 以外的边缘距页面顶部的距离

bottom:包围盒下边 border 以外的便于距页面顶部的距离

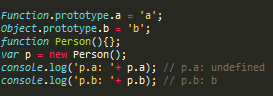
width: content + padding + border

height: content + padding + border

注意，设置外边距时外边距合并的情况

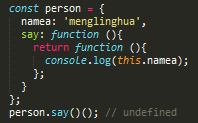
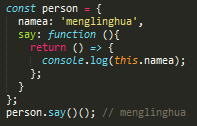
### 59.requestAnimationFrame 原理？是同步还是异步？

60.下面代码输出结果？为什么？

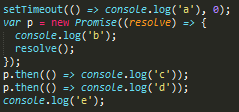


Object、 Function是两种type

### 61.下面代码输出结果？为什么？

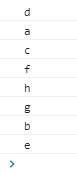
 

62.下面代码输出结果？为什么？

63.



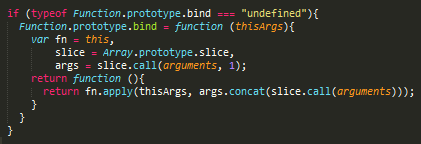
控制台：

/谁知道为啥结果不一样？？？？？？？？？？？？？

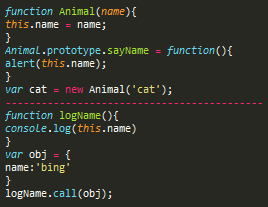
// 直接在控制台中运行结果： d a c f h g b e

// 在页面的script标签中运行结果：d a c f h b g e

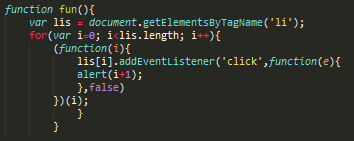
64.js bind 实现机制？手写一个 bind 方法？



65.手写闭包，继承

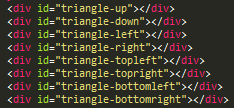


66..手写事件绑定

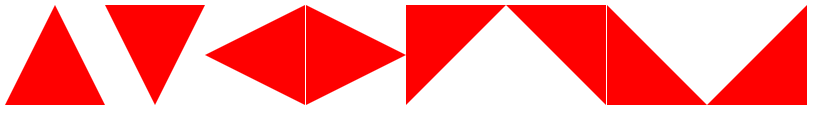


67.css3画一个三角形，用content：0，boder：很大，三面border透明

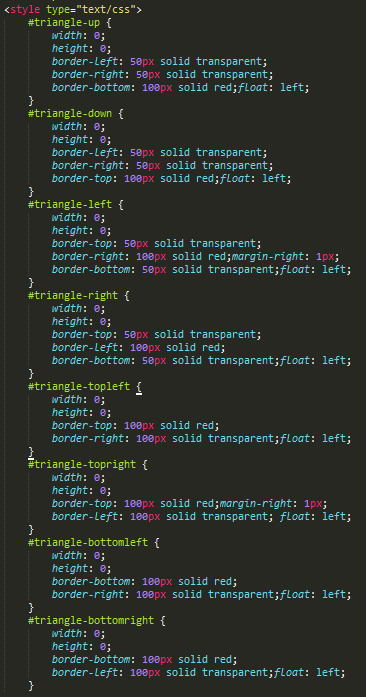
html



结果：



样式如下：



68..图片延迟加载怎么实现，监听scroll事件，替换src

69.websocket原理，应用场景：扫描二维码后，跳转登陆

70.前端路由的实现原理，hashchange

71.手写after方式清除浮动，一个冒号和两个冒号的区别

72.CSS选择器，nth-chlid(n)和nth-type(n)区别

73函数声明有哪几种形式，function aaa(){} 和var aaa= function(){}有什么区别，我没讲清楚.

74.手写一个盒子模型中，获取盒子内子节点的class样式，盒子内节点的类型不可知

我用了事件监听，但没考虑浏览器兼容性，也没能写出来怎么判定节点的类型（枚举不可）

面试官问我自己给这个代码打几分，我说60分

面试官说这种代码兼容性有问题，根本不能在线上运行，所以60分也没QwQ

75.javascript实现的底层原理

76.项目中写过最复杂的组件

77.当场写一个复杂的酒店预订table，没回答好

78.js继承中父类如果抛出异常如何解决？比如父类要求定长的参数，不满足时会抛出异常，此时如何处理？

79.常见的method有哪些？都是什么作用？PUT和POST的区别？

80.常见的http状态码都有哪些？

81.http请求头的connection:keep-alive是什么作用？与TCP的keep-alive的区别？

82.如何用两个栈实现一个队列？

83.手写实现：如何判断“abc”中字符是否都在“abcd”出现？

84.几种常见的排序算法中哪些是不稳定的？什么是不稳定？

85 如何实现一个计数器；（闭包就可以）   
86加入上面实现的计数器为f函数。

a = f();

b = f();

a();

b();

1

2

3

4

问输出是多少？（这一题答得不好，其实a和b是一样的函数，应该都输出1）   
  
87. 如何提升页面加载速度？   
88 因为上面提到回流和重绘，又问回流和重绘的区别？   
89. Web安全是否了解过？XSS原理和如何防范？   
90.. 跨域？如何实现？（提到JSONP，然后问缺点是啥？）   
  
91. 盒模型是啥？   
92. 有一个div父元素，里面有很多子元素，比如div，p等等，如何实现点击div父元素里面任意一个紧邻的子元素，打印出其中的innerHTML？（提到事件委托）   
93. 下来是两道简单的算法题：有一个排好序的数组，并且数字都没有重复，给定一个数字m，判断数组中是否存在m的值，并且存在返回位置，如果不存在，返回-1.（用二分查找，属于递归，递归的终止条件）.另一个是，求m的n次方，m和n都是整数。（这里需要考虑整数包括正整数和负整数，另外说了一个O(n)的方法，又问有没有更优的。后面想到可以判断n是否为2的倍数，一直除2，每次最终的数等于上一次得到的数乘以m的2次方，这样就是O(lgn)的复杂度应该。）    
  
94. HTTPS是什么？用了什么加密算法？   
95. 如果打开一个页面，出现白屏，可能的原因是什么？   
96. 如果页面加载的很慢，可能是什么原因？   
97.如何去除浮动？

方法一：使用一个空的div

方法二：overflow:hidden

方法三： display:inline-block;

方法四： position:absolute;

方法五：float:left;

方法六： zoom:1

方法七：after伪类+content方式

98..jsonp原理和缺点：利用在页面中创建<script>节点的方法向不同域提交HTTP请求，代码如下：

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

scriptElement.type = "text/javascript";

scriptElement.src = "跨域的url";

document.getElementByTaName("head")[0].appendChild(scriptElement);

jsonp只支持get请求不支持post等其他请求，不能解决不同域的两个页面之间进行javascript调用的问题。

99.jauery中map和each的区别？

map将遍历的结果映射到一个新的数组，each还是原来的数组

100.谈谈对作用域链和继承的理解。

http://www.cnblogs.com/lhb25/archive/2011/09/06/javascript-scope-chain.html

101.谈谈对闭包的理解。闭包会造成哪些影响？为什么会造成该影响？

102.图片都有哪几种格式？各自的优缺点是什么？png格式有哪些？各自的特点是什么？

PNG8：8位的png最多支持256（2的8次方）种颜色，8位的png其实支持不透明、索引透明、alpha透明。

PNG24：支持2的24次方种颜色，表现为不透明。

PNG32：支持2的32次方种颜色，32位是我们最常使用的格式，它是在png在24位的png基础上增加了8位的透明信息，支持不同程度的半透效果。

103.translate和animate的区别？

104.对css3中bfc的理解和应用？如何实现左边是固定宽度，右边自适应的布局？

105.webapp和native app的优缺点？

106.在百度搜索框输入关键字，回车都发生了什么

107.Div是块级元素吗？块级元素和行内元素的区别。说说display有哪几种属性

108数组判断

109：Js判断Null类型返回什么:

Object

110说说对xml的了解

111：Html5和之前HTML的区别

112：13寸电脑的分辨率

113:对象和数据类型的区别

## ES6面试题：

Promise原理、实现异步的方法

webpack类似的构建工具

为什么前端模块化、工程化

js数据类型，我多说了一个symbol，然后问我symbol还有啥API

VUE：

vue双向数据绑定

vue组件通信、数组值的修改

vue生命周期

vue mixin

项目中出彩的地方

最近了解的新技术

37.平时怎么学习的，途径1，2，3，4，……

38.你觉得最有成就感的事情是什么1，2，3，4，……

你有团队协作经历没1，2，3，4，……

你了解的前端技术前沿的团队有哪些1，2，3，4，……

你平时有什么爱好1，2，3，4，……

5年后希望成为什么样的人

实习经历

为什么学前端

你不打算以后做产品之类的吗

你最看重公司什么

之前有了解过美团吗

问我有什么问题。我问了美团点评和美团的关系，之后hr就在分析

怎样学习前端的？

3、都看了哪些技术博客？（因为我回答上题的时候说经常会看一些博客）

4、最近接触到哪些前端的技术，正在学习的？

5、你认为对前端来说，什么样的代码才算是好的代码？

你认为前端的主要工作职责是什么？为什么需要前端？

职业规划，业余时间兴趣等等

至此，3面挂