1、什么是盒子模型？

在网页中，一个元素占有空间的大小由几个部分构成，其中包括元素的内容(content)，元素的内边距(padding)，元素的边框(border)，元素的外边距(margin)四个部分。这四个部分占有的空间中，有的部分可以显示相应的内容，而有的部分只用来分隔相邻的区域或区域。4个部分一起构成了css中元素的盒模型。

2、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

3、CSS实现垂直水平居中

一道经典的问题，实现方法有很多种，以下是其中一种实现：

HTML结构：

3.1

<div class="wrapper">

<div class="content"></div>

</div>

.wrapper{position:relative;}

.content{

background-color:#6699FF;

width:200px;

height:200px;

position: absolute; //父元素需要相对定位

top: 50%;

left: 50%;

margin-top:-100px ; //二分之一的height，width

margin-left: -100px;

}

3.2

.div2 {

width: 40px;

height: 40px;

background-color: red;

position: absolute;

margin: auto;

top: 0;

left: 0;

right: 0;

bottom: 0;

}

3.3

<style type="text/css">

.div1 {

width: 100px;

height: 100px;

border: 1px solid #000000;

display: flex;

/\*!\*flex-direction: column;\*!可写可不写\*/

justify-content: center;

align-items: center;

}

.div2 {

height: 40px;

width: 40px;

background-color: red;

}

</style>

<div class="div1 ">

<div class="div2 ">

</div>

</div>

3.4

<style type="text/css">

.div1 {

width: 100px;

height: 100px;

border: 1px solid #000000;

position: relative;

}

.div2 {

height: 40px;

width: 40px;

background-color: red;

position: absolute;

top: 50%;

left: 50%;

-ms-transform: translate(-50%, -50%);

-moz-transform: translate(-50%, -50%);

-o-transform: translate(-50%, -50%);

transform: translate(-50%, -50%);

}

</style>

<div class="div1 ">

<div class="div2 ">

</div>

</div>

4、简述一下src与href的区别

href 是指向网络资源所在位置，建立和当前元素(锚点)或当前文档(链接)之间的链接，用于超链接。

src是指向外部资源的位置，指向的内容将会嵌入到文档中当前标签所在位置;在请求src资源时会将其指向的资源下载并应用到文档内，例如js脚本，img图片和frame等元素。当浏览器解析到该元素时，会暂停其他资源的下载和处理，直到将该资源加载、编译、执行完毕，图片和框架等元素也如此，类似于将所指向资源嵌入当前标签内。这也是为什么将js脚本放在底部而不是头部。

5、什么是CSS Hack？

一般来说是针对不同的浏览器写不同的CSS,就是 CSS Hack。

IE浏览器Hack一般又分为三种，条件Hack、属性级Hack、选择符Hack(详细参考CSS文档：css文档)。例如：

// 1、条件Hack

<!--[if IE]>

<style>

.test{color:red;}

</style>

<![endif]-->

// 2、属性Hack

.test{

color:#0909; /\* For IE8+ \*/

\*color:#f00; /\* For IE7 and earlier \*/

\_color:#ff0; /\* For IE6 and earlier \*/

}

// 3、选择符Hack

\* html .test{color:#090;} /\* For IE6 and earlier \*/

\* + html .test{color:#ff0;} /\* For IE7 \*/

6、简述同步和异步的区别

同步是阻塞模式，异步是非阻塞模式。

同步就是指一个进程在执行某个请求的时候，若该请求需要一段时间才能返回信息，那么这个进程将会一直等待下去，直到收到返回信息才继续执行下去;

异步是指进程不需要一直等下去，而是继续执行下面的操作，不管其他进程的状态。当有消息返回时系统会通知进程进行处理，这样可以提高执行的效率。

7、px和em的区别

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

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

8、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

9、浏览器的内核分别是什么？

IE: trident内核

Firefox：gecko内核

Safari：webkit内核

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

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

JavaScript部分

1、怎样添加、移除、移动、复制、创建和查找节点？

1)创建新节点

createDocumentFragment() //创建一个DOM片段

createElement() //创建一个具体的元素

createTextNode() //创建一个文本节点

2)添加、移除、替换、插入

appendChild() //添加

removeChild() //移除

replaceChild() //替换

insertBefore() //插入

3)查找

getElementsByTagName() //通过标签名称

getElementsByName() //通过元素的Name属性的值

getElementById() //通过元素Id，唯一性

2、实现一个函数clone，可以对JavaScript中的5种主要的数据类型(包括Number、String、Object、Array、Boolean)进行值复制。

/\*\*

\* 对象克隆

\* 支持基本数据类型及对象

\* 递归方法

\*/

function clone(obj) {

var o;

switch (typeof obj) {

case "undefined":

break;

case "string":

o = obj + "";

break;

case "number":

o = obj - 0;

break;

case "boolean":

o = obj;

break;

case "object": // object 分为两种情况 对象(Object)或数组(Array)

if (obj === null) {

o = null;

} else {

if (Object.prototype.toString.call(obj).slice(8, -1) === "Array") {

o = [];

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

o.push(clone(obj[i]));

}

} else {

o = {};

for (var k in obj) {

o[k] = clone(obj[k]);

}

}

}

break;

default:

o = obj;

break;

}

return o;

}

3、如何消除一个数组里面重复的元素？

// 方法一：

var arr1 =[1,2,2,2,3,3,3,4,5,6],

arr2 = [];

for(var i = 0,len = arr1.length; i< len; i++){

if(arr2.indexOf(arr1[i]) < 0){

arr2.push(arr1[i]);

}

}

5、在Javascript中什么是伪数组？如何将伪数组转化为标准数组？

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

6、Javascript中callee和caller的作用？

caller是返回一个对函数的引用，该函数调用了当前函数;

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

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

sessionStorage用于本地存储一个会话(session)中的数据，这些数据只有在同一个会话中的页面才能访问并且当会话结束后数据也随之销毁。因此sessionStorage不是一种持久化的本地存储，仅仅是会话级别的存储。而localStorage用于持久化的本地存储，除非主动删除数据，否则数据是永远不会过期的。

web storage和cookie的区别

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

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

8、手写数组快速排序

关于快排算法的详细说明，可以参考阮一峰老师的文章快速排序

“快速排序”的思想很简单，整个排序过程只需要三步：

(1)在数据集之中，选择一个元素作为”基准”(pivot)。

(2)所有小于”基准”的元素，都移到”基准”的左边;所有大于”基准”的元素，都移到”基准”的右边。

(3)对”基准”左边和右边的两个子集，不断重复第一步和第二步，直到所有子集只剩下一个元素为止。

9、统计字符串”aaaabbbccccddfgh”中字母个数或统计最多字母数。

var str = "aaaabbbccccddfgh";

var obj = {};

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

var v = str.charAt(i);

if(obj[v] & obj[v].value == v){

obj[v].count = ++ obj[v].count;

}else{

obj[v] = {};

obj[v].count = 1;

obj[v].value = v;

}

}

for(key in obj){

document.write(obj[key].value +'='+obj[key].count+' '); // a=4 b=3 c=4 d=2 f=1 g=1 h=1

}

10、写一个function，清除字符串前后的空格。(兼容所有浏览器)

function trim(str) {

if (str & typeof str === "string") {

return str.replace(/(^s\*)|(s\*)$/g,""); //去除前后空白符

}

}

其他部分

1、一次完整的HTTP事务是怎样的一个过程？

基本流程：

a. 域名解析

b. 发起TCP的3次握手

c. 建立TCP连接后发起http请求

d. 服务器端响应http请求，浏览器得到html代码

e. 浏览器解析html代码，并请求html代码中的资源

f. 浏览器对页面进行渲染呈现给用户

2、对前端工程师这个职位你是怎么样理解的？

css

calc, support, media各自的含义及用法？

@support主要是用于检测浏览器是否支持CSS的某个属性，其实就是条件判断，如果支持某个属性，你可以写一套样式，如果不支持某个属性，你也可以提供另外一套样式作为替补。

calc() 函数用于动态计算长度值。 calc()函数支持 "+", "-", "\*", "/" 运算；

@media 查询，你可以针对不同的媒体类型定义不同的样式。

css水平、垂直居中的写法，请至少写出4种？

这题考查的是css的基础知识是否全面，所以平时一定要注意多积累

水平居中

行内元素: text-align: center

块级元素: margin: 0 auto

position:absolute +left:50%+ transform:translateX(-50%)

display:flex + justify-content: center

垂直居中

设置line-height 等于height

position：absolute +top:50%+ transform:translateY(-50%)

display:flex + align-items: center

display:table+display:table-cell + vertical-align: middle;

1rem、1em、1vh、1px各自代表的含义？

rem

rem是全部的长度都相对于根元素<html>元素。通常做法是给html元素设置一个字体大小，然后其他元素的长度单位就为rem。

em

子元素字体大小的em是相对于父元素字体大小

元素的width/height/padding/margin用em的话是相对于该元素的font-size

vw/vh

全称是 Viewport Width 和 Viewport Height，视窗的宽度和高度，相当于 屏幕宽度和高度的 1%，不过，处理宽度的时候%单位更合适，处理高度的 话 vh 单位更好。

px

px像素（Pixel）。相对长度单位。像素px是相对于显示器屏幕分辨率而言的。

一般电脑的分辨率有{1920\*1024}等不同的分辨率

1920\*1024 前者是屏幕宽度总共有1920个像素,后者则是高度为1024个像素

画一条0.5px的直线？

考查的是css3的transform

height: 1px;

transform: scale(0.5);

说一下盒模型？

盒模型是css中重要的基础知识，也是必考的基础知识

盒模型的组成，由里向外content,padding,border,margin.

在IE盒子模型中，width表示content+padding+border这三个部分的宽度

在标准的盒子模型中，width指content部分的宽度

box-sizing的使用

box-sizing: content-box 是W3C盒子模型

box-sizing: border-box 是IE盒子模型

box-sizing的默认属性是content-box

画一个三角形？

这属于简单的css考查，平时在用组件库的同时，也别忘了原生的css

.a{

width: 0;

height: 0;

border-width: 100px;

border-style: solid;

border-color: transparent #0099CC transparent transparent;

transform: rotate(90deg); /\*顺时针旋转90°\*/

}

<div class="a"></div>

清除浮动的几种方式，及原理？

清除浮动简单，但这题要引出的是BFC，BFC也是必考的基础知识点

::after / <br> / clear: both

创建父级 BFC(overflow:hidden)

父级设置高度

BFC （块级格式化上下文），是一个独立的渲染区域，让处于 BFC 内部的元素与外部的元素相互隔离，使内外元素的定位不会相互影响。

触发条件:

根元素

position: absolute/fixed

display: inline-block / table

float 元素

ovevflow !== visible

规则:

属于同一个 BFC 的两个相邻 Box 垂直排列

属于同一个 BFC 的两个相邻 Box 的 margin 会发生重叠

BFC 的区域不会与 float 的元素区域重叠

# **页面两栏布局**

法一：左侧float:left;右侧margin-left;







计算 BFC 的高度时，浮动子元素也参与计算

文字层不会被浮动层覆盖，环绕于周围

html

说一下<label>标签的用法

label标签主要是方便鼠标点击使用，扩大可点击的范围，增强用户操作体验

遍历A节点的父节点下的所有子节点

这题考查原生的js操作dom,属于非常简单的基础题，但长时间使用mvvm框架，可能会忘记

<script>

var b=document.getElementById("a").parentNode.children;

console.log(b)

</script>

js

用js递归的方式写1到100求和？

递归我们经常用到，vue在实现双向绑定进行数据检验的时候用的也是递归，但要我们面试的时候手写一个递归，如果对递归的概念理解不透彻，可能还是会有一些问题。

function add(num1,num2){

var num = num1+num2;

if(num2+1>100){

return num;

}else{

return add(num,num2+1)

}

}

var sum =add(1,2);

页面渲染html的过程？

不需要死记硬背，理解整个过程即可

浏览器渲染页面的一般过程：

1.浏览器解析html源码，然后创建一个 DOM树。并行请求 css/image/js在DOM树中，每一个HTML标签都有一个对应的节点，并且每一个文本也都会有一个对应的文本节点。DOM树的根节点就是 documentElement，对应的是html标签。

2.浏览器解析CSS代码，计算出最终的样式数据。构建CSSOM树。对CSS代码中非法的语法它会直接忽略掉。解析CSS的时候会按照如下顺序来定义优先级：浏览器默认设置 < 用户设置 < 外链样式 < 内联样式 < html中的style。

3.DOM Tree + CSSOM --> 渲染树（rendering tree）。渲染树和DOM树有点像，但是是有区别的。

DOM树完全和html标签一一对应，但是渲染树会忽略掉不需要渲染的元素，比如head、display:none的元素等。而且一大段文本中的每一个行在渲染树中都是独立的一个节点。渲染树中的每一个节点都存储有对应的css属性。

4.一旦渲染树创建好了，浏览器就可以根据渲染树直接把页面绘制到屏幕上。

以上四个步骤并不是一次性顺序完成的。如果DOM或者CSSOM被修改，以上过程会被重复执行。实际上，CSS和JavaScript往往会多次修改DOM或者CSSOM。

说一下CORS？

CORS是一种新标准，支持同源通信，也支持跨域通信。fetch是实现CORS通信的

如何中断ajax请求？

一种是设置超时时间让ajax自动断开，另一种是手动停止ajax请求，其核心是调用XML对象的abort方法，ajax.abort()

说一下事件代理？

事件委托是指将事件绑定到目标元素的父元素上，利用冒泡机制触发该事件

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

var target = event.target || event.srcElement;

if(!!target && target.nodeName.toUpperCase() === "LI"){

console.log(target.innerHTML);

}

}, false);

target、currentTarget的区别？

currentTarget当前所绑定事件的元素

target当前被点击的元素

说一下宏任务和微任务？

宏任务：当前调用栈中执行的任务称为宏任务。（主代码快，定时器等等）。

微任务： 当前（此次事件循环中）宏任务执行完，在下一个宏任务开始之前需要执行的任务为微任务。（可以理解为回调事件，promise.then，proness.nextTick等等）。

宏任务中的事件放在callback queue中，由事件触发线程维护；微任务的事件放在微任务队列中，由js引擎线程维护。

说一下继承的几种方式及优缺点？

说比较经典的几种继承方式并比较优缺点就可以了

借用构造函数继承，使用call或apply方法，将父对象的构造函数绑定在子对象上

原型继承，将子对象的prototype指向父对象的一个实例

组合继承

原型链继承的缺点

字面量重写原型会中断关系，使用引用类型的原型，并且子类型还无法给超类型传递参数。

借用构造函数（类式继承）

借用构造函数虽然解决了刚才两种问题，但没有原型，则复用无从谈起。

组合式继承

组合式继承是比较常用的一种继承方法，其背后的思路是使用原型链实现对原型属性和方法的继承，而通过借用构造函数来实现对实例属性的继承。这样，既通过在原型上定义方法实现了函数复用，又保证每个实例都有它自己的属性。

说一下闭包？

闭包的实质是因为函数嵌套而形成的作用域链

闭包的定义即：函数 A 内部有一个函数 B，函数 B 可以访问到函数 A 中的变量，那么函数 B 就是闭包

export和export default的区别？

使用上的不同

export default xxx

import xxx from './'

export xxx

import {xxx} from './'

说一下自己常用的es6的功能？

此题是一道开放题，可以自由回答。但要注意像let这种简单的用法就别说了，说一些经常用到并有一定高度的新功能

像module、class、promise等，尽量讲的详细一点。

什么是会话cookie,什么是持久cookie?

cookie是服务器返回的，指定了expire time（有效期）的是持久cookie,没有指定的是会话cookie

数组去重？

此题看着简单，但要想面试官给你高分还是有难度的。至少也要写出几种方法

js

var arr=['12','32','89','12','12','78','12','32'];

// 最简单数组去重法

function unique1(array){

var n = []; //一个新的临时数组

for(var i = 0; i < array.length; i++){ //遍历当前数组

if (n.indexOf(array[i]) == -1)

n.push(array[i]);

}

return n;

}

arr=unique1(arr);

// 速度最快， 占空间最多（空间换时间）

function unique2(array){

var n = {}, r = [], type;

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

type = typeof array[i];

if (!n[array[i]]) {

n[array[i]] = [type];

r.push(array[i]);

} else if (n[array[i]].indexOf(type) < 0) {

n[array[i]].push(type);

r.push(array[i]);

}

}

return r;

}

//数组下标判断法

function unique3(array){

var n = [array[0]]; //结果数组

for(var i = 1; i < array.length; i++) { //从第二项开始遍历

if (array.indexOf(array[i]) == i)

n.push(array[i]);

}

return n;

}

es6

es6方法数组去重

arr=[...new Set(arr)];

es6方法数组去重，第二种方法

function dedupe(array) {

return Array.from(new Set(array)); //Array.from()能把set结构转换为数组

}

get、post的区别

此题比较简单，但一定要回答的全面

1.get传参方式是通过地址栏URL传递，是可以直接看到get传递的参数，post传参方式参数URL不可见，get把请求的数据在URL后通过？连接，通过&进行参数分割。psot将参数存放在HTTP的包体内

2.get传递数据是通过URL进行传递，对传递的数据长度是受到URL大小的限制，URL最大长度是2048个字符。post没有长度限制

3.get后退不会有影响，post后退会重新进行提交

4.get请求可以被缓存，post不可以被缓存

5.get请求只URL编码，post支持多种编码方式

6.get请求的记录会留在历史记录中，post请求不会留在历史记录

7.get只支持ASCII字符，post没有字符类型限制

你所知道的http的响应码及含义？

此题有过开发经验的都知道几个，但还是那句话，一定要回答的详细且全面。

1xx(临时响应)

100: 请求者应当继续提出请求。

101(切换协议) 请求者已要求服务器切换协议，服务器已确认并准备进行切换。

2xx(成功)

200：正确的请求返回正确的结果

201：表示资源被正确的创建。比如说，我们 POST 用户名、密码正确创建了一个用户就可以返回 201。

202：请求是正确的，但是结果正在处理中，这时候客户端可以通过轮询等机制继续请求。

3xx(已重定向)

300：请求成功，但结果有多种选择。

301：请求成功，但是资源被永久转移。

303：使用 GET 来访问新的地址来获取资源。

304：请求的资源并没有被修改过

4xx(请求错误)

400：请求出现错误，比如请求头不对等。

401：没有提供认证信息。请求的时候没有带上 Token 等。

402：为以后需要所保留的状态码。

403：请求的资源不允许访问。就是说没有权限。

404：请求的内容不存在。

5xx(服务器错误)

500：服务器错误。

501：请求还没有被实现。