2017/11

想要进入腾讯，首先会进行一些基础知识的考核，这些基础及时包括数据结构、算法、数据库、安全等方面，最后是关于性格和项目经验的问题。

**基础知识**

1. prototype和**proto**的关系是什么
2. meta viewport原理
3. 域名收敛是什么
4. float和display：inline-block；的区别
5. 前端优化策略列举
6. 首屏、白屏时间如何计算
7. 闭包
8. 作用域链
9. ajax如何实现、readyState五中状态的含义
10. jsonp如何实现
11. 怎么处理跨域
12. restful的method解释
13. get和post的区别
14. 事件模型解释
15. 编写一个元素拖拽的插件
16. 编写一个contextmenu的插件
17. 编写web端cookie的设置和获取方法
18. 兼容ie６的水平垂直居中
19. 兼容ie的事件封装
20. h5和原生android的优缺点
21. 编写h5需要注意什么
22. xss和crsf的原理以及怎么预防
23. css优先级
24. 如何实现点击radio的文字描述控制radio的状态（通过label实现）
25. delegate如何实现
26. jsonp有什么安全问题？为什么有这些安全问题？
27. 单链表和双向链表的区别？
28. 有用过reactjs？简单说一下其生命周期方法

**框架**

如何了解这些前端的框架？如angular，react，vue。

**angularjs**

1. angular的directive怎么写
2. angular的脏检查（双向绑定）是如何实现的
3. 依赖注入如何实现
4. scope如何实现
5. $parse模块如何实现（主要自己写了一个类似的库）

**react**

1. react在setState后发生了什么，请结合源码说说你的理解。
2. flux解释。
3. 说说你对React框架的理解。

**requirejs**

1. amd和cmd区别，怎么了解到这些区别的，是否是去看了规范。
2. requirejs那些经常用的方法，然后对其进行解释。

**Node**

1. Buffer模块是干什么的。
2. Stream是什么，使用的两种模式。
3. http模块如何将异步处理方式实现成同步处理方式。

**其他问题**

1. utf8和gbk的区别。
2. 知道页面上某个点的坐标，如何获取该坐标上的所有元素。
3. angular、react和jQuery适合哪些应用场景（建议查看各个框架产生背景）。
4. 7点15分小于180度的夹角是多少。
5. 如何实现大数相加。
6. 给５升和６升的水杯如何倒出３升的水。
7. 一班喜欢足球的人60%，喜欢排球的70%，喜欢篮球的80%，求喜欢足球和排球的占多少。
8. 前端异常监测如何实现。
9. 直播点赞按钮的冒泡功能如何实现。
10. js的uglify如何实现。

**优化方面**

1. 前端工程化方面做了哪些东西，怎么做。

**算法**

什么二叉树，从一堆大数中选出满足条件的数要求最快。

2015/09

**【一面】~=110分钟  2013/04/24 11:20  星期三**

进门静坐30分钟做题。

填空题+大题+问答题

>>填空题  
何时接触电脑 何时接触前端  
运算符 字符串处理        延时 display        position        XMLHttpRequest        正则  
Jquery绑定事件        cookie

>>大题  
BOM浏览器信息  
浏览器兼容性举例  
闭包作用与举例  
新闻ul列表，插入3个，获取index  
全文单词首字母大写  
ajax过程

>>逻辑题  
任何时间分针和时针夹角  
文章单词（字母）频率出现最高  
30分钟等到车概率是70%，那10分钟呢？  
一个班学舞蹈的有75%，学唱歌的有85%，问同时学习舞蹈和唱歌的最多多少，最少多少？  
两个很大数的加法运算

>>我的提问  
为什么不问我框架  
你觉得我哪些方面还有欠缺，觉得我如何  
如果有望进入二面，什么时候可以得到通知

>>完了之后一起下楼时  
我说其他几家公司的问题和腾讯的有比较大的区别，  
区别是XXXXXX  
然后他问我家在哪里  
然后说深圳还是比较适合我  
我说但愿还能见到你  
他笑了笑

**【二面】~=40分钟 2013/04/25 14:00  星期四**  
面试时间是两点钟，两点过几分进入了面试官所在的包间。  
进去之前楼下签到的hr要我把一摞草稿纸带到621房间的面试官，进门给他之后，他说那把大椅子是坏的，   
要我往后坐一点（防止向前倾倒）。

开始的时候他没怎么说话，埋头看着我的简历，于是我主动开始扯了几句关于自己的介绍。也不知他听到没   
有，等我说完，马上又说：好吧，你先自我介绍下。（一脸黑线...）

期间问了我为什么不打算考研，为什么选择前端，我做的最自豪的事情是什么。

叽哩吧啦一阵子后，他要我说说我的人生规划和职业规划（颇为蛋疼，这些问题不是应该HR面才会问的么   
），我顺着他的来，也比较xx的说了一番。

反向代理  
输入网址到出现页面的过程（IP解析，DNS解析等）  
页面流量  
堆栈

接着，问了我-1在计算机中怎么表示，如何判断数组A是否为数组B的子集。  
然后是一个概率问题，假设长江以北有80W人，长江以南有20W人，A向B打电话算一个电话，如果打了   
10W个电话，那么长江以北打向长江以南的电话有多少个？

没问太多问题，都是些基础的，最后他问我有什么问题。

>>我问了问  
为什么不问我前端的问题  
如果通过二面，什么时候可以给我回复  
对我有什么学习上的建议

面试官留言：“一把刀有刀锋和刀背，刀锋要磨锐利，但是刀背的锻炼也要加强。”

**【HR面】~=30分钟 2013/04/27 20:00 星期六**  
进去先看上几位面试官对我的评价，安静时刻我跟她扯淡了几句，缓和了下尴尬的气氛。  
然后她瞄了瞄我的简历，接着就问我考不考研，很明显我的答案是不。然后她又问为什么~  
接着让我说说自己最满意的一件事和最不满意的一件事，自己最大的缺点和最大的优点  
扯了半天之后（此处略去20分钟扯淡），又开始看我的简历，问了我项目中遇到哪些困难，我说了一个还未   
解决的bug，她反问为什么不解决，然后叽哩吧啦了一阵子。

最后问我有没有什么问题。按惯例，问了问最早什么时候给通知，然后问了问期间如果有不明白问题可以通过   
什么方式联系她之类的。

整个过程还算顺利。

**HR告诉我五一之后可以查询录取状态，求人品吧~**

**顺便把在搜狐和网易的面试笔试经历也说说吧~**

**>>搜狐**

搜狐的太久了，记不太清楚了

那次只为攒经验，是俺平生第一次参加公司面试。不过收获还是有的~

时间太久了，也记不清了，没有提交笔试题，但是还是要我去面试了。  
问他有几轮面试，告诉我因时间匆忙，只给一轮技术面，如果满意直接HR。

大概问题记不太清。  
虽然感觉答的还不错，但是最后还是被刷了。

【技术面】~=60分钟  
-问了下我在团队学了些什么。   
-然后就是很多“你是怎么理解XX”。。  
-XX中有 闭包，原型链， 框架结构等

>>**网易**

今天4.27，好像是3.26参加的面试，时间太久，也记得不是很清楚了。

我没有把任何面试官当做正经的面试官，男的就是我的学长，女的就是我的学姐，仅此而已，所以每次交流都是平和的心态。

学长们（每次面我的都是个男的）时不时露出笑意，现场气氛没有那么紧张。

【一面】 ~=30分钟  
闭包  
IE6趋势  
框架

**简要阐述XSS和CSRF攻击及防范**

**11.强类型语言有继承，在JS里面如何做到继承呢？(*proto*和prototype)**

11.1 例如构造函数有一个test()方法，prototype上也有一个test()，这两个方法有区别吗？

**12.页面性能优化有哪些方式？**

12.1 文件、脚本合并是如何优化的呢？

12.2 重绘，重排是怎么回事？有什么区别？

12.3 请举例说明重绘，什么情况下会重绘？

12.4 你认为样式是会引起重绘的吗？什么样式会引起重绘/重排？什么属性会引起重绘/重排？

**13.浏览器如何知道一个文件资源是否需要缓存？**

**14.HTTP状态码304,502,503**

14.1 *1,2,3,4,5* 开头的状态码都表示什么？

**15.Node、Vim、angular简述**

15.1 Node上如何处理异常情况？（Ps.我回答的是开发调试。。。找js文件）   
15.2 已经上线的项目，出问题，怎么样去处理异常？

**16.构建工具除了Webpack，还接触过其他的吗？**

16.1 Webpack的优点和应用场景？

## 编程大题

1、页面内有一个正方形元素A以及一个待放置区域B，实现对其拖拽和放下到B区域内，并且改变B区域背景颜色(不可用html5原生事件)。

2、实现超出整数存储范围的两个大正整数相加 function add(a, b) 。   
注意：参数 a 和 b 以及函数的返回值都是字符串。

3、页面内有一个input输入框，实现在数组arr查询命中词并和autocomplete效果。

这个环节基本上就知道自己GG了。。。

第一题考基本功，DOM、事件、定位、jsAPI

第二题考算法，基本逻辑

第三题考原理，基础

不得不说面试官的技术确实比较全面，腾讯的远程面试也是好狠，直接远程监控写代码的。。。鼠标和界面都不敢静止，静止可能就会被认为作弊（不知道他们是不是这样来判断的，反正我没有停下来过）。

【二面】 ~=90分钟  
框架比较 jQuery prototype YUI  
框架特点  
css兼容性 IE6 bug  
seaJS AMD CMD kissy  
JS原型链 闭包  
JS基础细节

不让谈百度框架

【HR面】~=25分钟  
自我介绍  
身边人的评价  
最不满意的一件事情  
规划  
为什么不想考研  
有什么困难

薪资待遇  
何时通知

收到腾讯的面试短信之后，心里很激动，随即就着手开始准备面试（对于笔试我就不赘述，要准备笔试的同学可以到[牛客](https://www.nowcoder.com/)或者[赛码](http://www.acmcoder.com/)进行练习）。从网上搜了很多面试的题目，复习的范围很广，但是也很杂。感觉这样的效率的确是有一些低。

    接下来我会说一下我的面试题目，以及对面试题目进行一个总结。由于面试时间是早上9点，所以我6点就起了床收拾一番之后就去面试地点，去的路上和面试完整个人的脑袋都是蒙的。有些面试的题目已经记不得，所以我就尽力回忆，有所遗漏请见谅。

    By the way，我遇到的这个面试官出牌有点不按套路，与以往首先让你介绍自己的流程不太一样。大家在面试的时候也要注意一下，学会随机应变。

**① 介绍自己在前端学习的经历**

    我把从大一到大三学习的与有关前端的课程，以及课程内容进行了描述；同时还说了一下自己在课余时间学习前端的一些方法，还有正在学习的ReactJs。

**② 有关ReactJs**

这个问题算是自己给自己挖了坑（后来了解到他所在的部门2015年用的框架是Angular，到了2016年以后就换成了ReactJs）。

    问：平时怎么学习React？

    答：因为是刚开始学习，所以就一步一步按照官方文档来学习。

    问（并用自己的电脑打开了官方文档给我看）：是这个英文版的吗？

    答：是的，没课的时候就会自己看一下，学的比较慢。现在才学习到React的State机制。

    问：好，那说一下自己学习ReactJs的一些收获。

    这个具体怎么答的我忘了，总之对这个答的不太满意。

**③ 在纸上写下来一个前端开发工程师必备的一些技能（专业技能，并非通用技能）。**

    我写的有：HTML/JS/当下流行的前端框架/计算机网络/数据结构/操作系统/对后端语言也要有一些了解。

    然后接下来的问题都是围绕着我写的这几点问的（又是自己挖坑自己跳）。

**④ 有关HTML/CSS的问题。**

    问：说一下display和position的取值。

    答：display的取值有：table/inline/block/inline-block（当时就说了这些，还有经常用到的none，table-cell这些一紧张忘记说了）；position的取值有：fixed/relative/absolute（static忘记说了）。

    问：说一下absolute。

    答：absolute是对元素的一个绝对定位，是相对于浏览器可视区域的一个定位，比如：position:absolute;top:o;这个CSS代码就会让元素显示在浏览器的最上方。position属性设置为absolute的元素会脱离文档流（又给自己挖坑了）。

    问：什么是文档流？

    答：我对文档流的理解就是：在文档流中的元素就像在一块画布上画的画，而脱离文档流就像在画布的上面放了个东西，脱离文档流只会就会遮挡处于文档流中的元素。再举个例子就像是：云彩和大地的一种关系。

**⑤ 有关Js的问题。**

    问：说一下你对事件的理解。

    答：我认为事件就是对用户操作的一种响应。

    （关于事件大家可以看一下这篇博客：[有关JS事件理解和总结](https://www.cnblogs.com/Leesoo/p/6892204.html)）

    问：再说一下事件代理吧。

    答：我对时间代理的理解就是事件A不由a做，而让b去做。

    问：为什么这样做呢？

    这里我不太了解，就如实招了，顺口说了一下有可能是为了降低耦合度吧。有关时间代理大家可以看一下下面两篇博客：

<https://www.cnblogs.com/liugang-vip/p/5616484.html>

<https://blog.csdn.net/supercoooooder/article/details/52190100>

**⑥ 有关HTTP的问题。**

    有关HTTP我叽里呱啦说了一些，大概内容有：HTTP的头部，HTTP的状态码，HTTP返回码，ajax中xmlHttpRequest()的应用，还有在浏览器输入域名之后的操作（比如：重定向->检查DNS缓存->DNS解析->发送请求等）。

    随后面试官在电脑上打开了一个抓包工具，让我讲一下每一个字段代表什么含义，这里大家可以参考这篇博客：[HTTP头部字段含义](https://blog.csdn.net/melody_day/article/details/53559054)（现在每个浏览器应该都会有开发者工具的插件，大家可以下载下来好好研究一下，熟练使用浏览器的插件会对开发效率有很大提升），下图是chrome的开发者工具里的抓包截图，大家可以参考一下：

      说完每个字段的含义又让我说了一下Cookie包含那些，以及Cookie，LocalStorage和Storage的区别（这一部分我没答上来）。

**⑦ 总结**

    这个面试持续了50分钟左右，总的来说问的问题并没有特别的偏，都是规规矩矩的面试题目，并且面试官也很和善。从上面的题目和我做过的面试题来看，最重要的就是打牢基础！！！下面我总结了一下面试前需要认真准备的内容（仅就个人经历而言）：

    1. 对于计算机基础课程尤其是计算机网络一定要认真学习，比如：DNS，TCP/IP，UDP以及HTTP，七层架构，五层架构之类的一定要烂熟于心；操作系统中进程的通信方式，进程阻塞的必要条件；数据结构里常见的排序和数据结构（链表，哈希表，栈，队等）的实现；

    2. 还有前端一些常见的布局，比如：垂直/水平居中布局，左中右布局等；如何清除浮动；

    3. JS中对ajax的实现；设置/获取Cookie；事件代理（也称事件委托）；虽然说JS是单线程的一门语言，但是也有异步操作，如：setTimeout()等；JS闭包，作用域，作用链域等。

    4. 尽可能去学习当下比较流行的前端框架，在和面试官交流的过程中他也建议我去好好学习一下React，现在各个大公司对React的人才需求很大。

**⑧ 结束语**

    今天的面试内容和自己的一些心得体会也就这些，作为一名大三的学生能通过笔试进入面试着实很开心，不管能不能拿到offer都希望自己的一些经历能帮助到大家。如果有什么错误欢迎指出，我会加以修改！

2018年02月01日 21:28:00

阅读数：1107

 今天下午的时候面了腾讯的前端工程师实习生岗位，感觉自己学的不够深入，很多小细节都没搞清楚，所以还是要多钻研吧。

 贴上面试题：

1.      块级标签和行内标签

2.      DOM事件流

3.      事件冒泡、事件捕获

4.      事件代理

5.      盒模型

6.      作用域、作用域链的形成及原理

7.      https的握手过程

8.      ajax请求过程

9.      XSS/CSRF(场景以及防范)

10.  http状态码

11.  304缓存过程（除了Etag还有什么）

12.  闭包怎么形成的

13.  Cookie包含哪些字段、expires、path是什么

14.  左右两边定宽，中间自适应

15.  怎么使文字垂直排布

16.  Call、apply、bind三者的区别

17.  智力题：5点15分，时针和分针的角度（67.5）

  智力题是真的问的很懵啊，当时在实验室面试，我一直在重复这道题，然后我师兄就在旁边告诉我是67.5度，然后我就立马说出来了。没想到他还问我是怎么算的，我以为只是一道简单的数学题，然后我就想哭了，早知道就直接说不知道了，心累，后面他就给我解释了一遍怎么算的。一道小学生题目居然没答出来，所以一定不要紧张，自己慢慢算，其实很简单的。还有就是，不知道就说不知道，不懂装懂才是最可怕的。

 后面面试官还给了些建议：

1.      多练习html+css（主要是html5），实现一个淘宝网首页的静态页面，主要是页面布局和样式布局

2.      轮播图……..

3.      打好js基础

4.      找一些前端的图谱，对照着学习哪些技能

5.      项目经验：自己做一些demo或者接一点项目私活

 虽然说大部分都知道，但是组织语言这方面还是有待提高，有很多东西都是只可意会不可言传。还有一些问的都不知道，就会很有挫败感。哎，前端渣渣，求不喷。

# 腾讯2018校招WEB前端开发笔试有感

2017年09月12日 21:27:02

阅读数：2896

　　鹅厂的校招考察的我感觉比较“杂”。   
　　前端开发感觉我的印象，考察了JS、TCP/IP协议、MVVM、数据库。都是在不定项选择中。没有单选题，无疑增加了难度。还考察了一些Linux操作系统、sizeof函数，数据结构也有一部分题。根结点，JS中考到了跨域。还有一两道数学几何体。竟然还考察了用css和table布局的不同。   
　　最后有三道编程题，难度都不易。   
　　今年系统升级了作弊筛查功能，让博主在宿舍考试有点尴尬。 

## JavaScript中的严格模式

var a = 5;

(function() {

var a = b = 3;

})();

console.log(a);

console.log(b);

* 1
* 2
* 3
* 4
* 5
* 6

如果是上面的代码，会输出什么？   
我想很多人都会知道，输出5和3。因为b前面没有var声明，会被当作全局变量声明。

"use strict";

var a = 5;

(function() {

var a = b = 3;

})();

console.log(a);

console.log(b);

* 1
* 2
* 3
* 4
* 5
* 6
* 7

那么，如果在严格模式下会怎么办呢？会在var a = b = 3;抛出未捕获的异常。

b is not defined

即全局变量必须显示声明考察

## JavaScript单线程

window.setTimeout(function() {

console.log(1);

}, 1);

window.setTimeout(function() {

console.log(2);

}, 2);

window.setTimeout(function() {

console.log(3);

}, 3);

window.requestAnimationFrame(function() {

console.log(4);

});

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12

上面代码执行情况可能有哪几种?   
一共有四种情况，分别是：

* 4123
* 1423
* 1243
* 1234

setTimeout表示的是隔了多少时间后，就将函数添加到事件处理后面，但是并不代表马上就要执行

## 考察IPV6

## 考察协议

* DNS
* FTP
* TLS
* POP3
* HTTP2

## 考察JavaScript单线程

var a = 1;

setTimeout(function() {

console.log(a);

}, 0);

var date1 = new Date();

while(new Date().getTime() - date1.getTime() <= 10) {

console.log(a);

}

a = 2;

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9

这道题，也是考察JavaScript的单线程，setTimeout的参数即使是0，表示0毫秒过后，将这个事件添加到队列里去，但是它依然要等待其他JavaScript代码处理结束后，才会执行队列里的代码。因此它的执行结果是：

* n个1
* 2

## JavaScript题目

var x = 1;

function a() {

alert(x);

var x = 2;

x++;

alert(x);

}

function b() {

alert(x);

x++;

alert(x);

}

if(0 < 100 < 0) {

a();

} else {

b();

}

if(0 < 100 < (0 + 4)) {

a();

} else {

b();

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25

经过测试，发现这道题，考察的是if语句数据类型的转换。它的实际过程是这么回事:

0小于100返回true，true再次被比较时回被转为1。   
true 小于 0 返回false，所以第一个if语句将会执行函数b里的代码。   
会依次弹出1, 2。   
第二个if 0 小于 100会返回true，true会被转为1。   
true 小于 4返回true，所以第二个if将会执行函数a里的代码。   
会依次弹出undefined, 3。

// 即   
1 2 undefined 3

## 考察nodejs

## 考察UDP的首部头

UDP首部与伪首部

UDP的首部由8个字节组成。

* 源端口：2个字节。
* 目的端口：2个字节。
* 长度：2个字节。
* 检验和：2个字节。

## 考察TCP的首部

TCP首部字段

一个TCP报文由TCP首部和数据组成的。TCP报文首部的前20个字节是固定的，后面有4n个字节是根据需要而增加的, 因此TCP首部的最小长度是20字节。

20个字节由以下部分组成：

* 源端口：2个字节
* 目的端口：2个字节
* 序号：4个字节, 因为TCP是面向字节流的，在一个TCP连接中传送的字节流中的每一个字节都按顺序编号。
* 确认号：4个字节，是期望收到下一个报文段的第一个数据字节的序号。
* 数据偏移：占4位，它指出TCP报文段的数据起始处距离TCP报文段的起始处有多远。
* 保留：占6位。
* 6个状态位：占6位，URG，ACK，PSH，RST，SYN，FIN。
* 窗口：占2个字节，指的是接受窗口的大小。
* 检验和：占2个字节。
* 紧急指针：占2个字节，只有URG=1时才有意义。
* 选项和填充：一共占4个字节，选项大小可变。
* 排序算法的时间复杂度，是否是稳定的   
  什么是稳定的？稳定其实指的是如果两个数大小一样，当经过排序后，它们的顺序依然没有改变，那么排序就是稳定的。

## 常见排序算法是否稳定

* 冒泡排序，n的平方，稳定
* 插入排序，n的平方，稳定
* 归并排序，n的log2n，稳定
* 基数排序，d(r+n), 稳定, r代表关键字的基数，d代表长度，n代表关键字的个数。
* 快速排序，n的log2n, 不稳定
* 希尔排序，n的1.3次方，不稳定
* 选择排序，n的平方，不稳定
* 堆排序，n的log2n, 不稳定

## sizeof(double)是表达式还是函数？

sizeof不是函数，它是编译时处理的，函数只能在运行时求值。

## 树

描述：有一棵树，它的度是4。度为4的节点个数是1，度为3的节点个数是2，度为2的节点个数是3，度为1的节点个数是4，问总节点的个数和节点为0的个数是多少。

所谓度就是节点子树的个数，即分支数目，我画出来是21个节点，度为0的个数是11。

## 网站性能优化的方法有哪些？

参见雅虎军规。

## 从输入URL到页面加载的过程

参见google。

## 求任意整数有多少种解法

2 = 1 + 1   
3 = 1 + 2   
4 = 1+ 3；4 = 2 + 2；

## 给定有序数组，求出数字出现的次数

可以自己设置一个hashTable

# 腾讯前端面试经验（一）

2018年03月12日 08:36:30

阅读数：1217

作者：汪汪  
链接：https://zhuanlan.zhihu.com/p/22606894  
来源：知乎  
著作权归作者所有，转载请联系作者获得授权。

这篇文章是对我大四秋招以来面试的总结，里面包含前端面试知识的方方面面，目前本人已经拿到腾讯offer，希望能对后面找工作的学习学妹们有所帮助。

腾讯面试对基础比较看重，然后需要你有两三个比较好的项目，一面重视面试者对前端基础的把握，还要手写代码，不过不难，二面部门的leader面，这一面比较难，面试官会对你的项目细节进行深挖，所以说项目要牛逼一点，最后还会有一道逻辑题（我没有答上来），三面是HR面，如果你想进大公司的话，下面这些技术是肯定要掌握的：html5，css3，JavaScript，略懂一点jQuery源码，Node.js，express，mongoose，数据库mongodb。大公司问的核心在于JavaScript。如果下面的知识点你都可以打上来，恭喜你拿下bat不是问题--2016-11-11写

转载请注明出处，码这么多字不容易。

一、html+css部分、

（1）css盒模型，可能会要求手写一个布局，这个布局基本上用到的css是margin的负值，boxing-sizing：border-box，布局尽量往这方面想。浏览器布局的基本元素是盒，在w3c的标准模式下，width=width，但是在怪异模式下，width=border\*2+padding\*2+width;其中后代元素的width：100%；参照的是右边的那个width，

（2）html5的新特性

1、标签语义化，比如header，footer，nav，aside，article，section等，新增了很多表单元素，入email，url等，除去了center等样式标签，还有除去了有性能问题的frame，frameset等标签

2、音视频元素，video，audio的增加使得我们不需要在依赖外部的插件就可以往网页中加入音视频元素。

3、新增很多api，比如获取用户地理位置的window.navigator.geoloaction，

4、websocket

websocket是一种协议，可以让我们建立客户端到服务器端的全双工通信，这就意味着服务器端可以主动推送数据到客户端，

5、webstorage，webstorage是本地存储，存储在客户端，包括localeStorage和sessionStorage，localeStorage是持久化存储在客户端，只要用户不主动删除，就不会消失，sessionStorage也是存储在客户端，但是他的存在时间是一个回话，一旦浏览器的关于该回话的页面关闭了，sessionStorage就消失了，

6、缓存

html5允许我们自己控制哪些文件需要缓存，哪些不需要，具体的做法如下：

1. 1、首先给html添加manifest属性，并赋值为cache.manifest
2. 2、cache.manifest的内容为:
3. CACHE MANIFEST
4. #v1.2
5. CACHE : *//表示需要缓存的文件*
6. a.js
7. b.js
8. NETWORK: *//表示只在用户在线的时候才需要的文件，不会缓存*
9. c.js
10. FALLBACK
11. / /index.html //表示如果找不到第一个资源就用第二个资源代替

7、web worker，web worker是运行在浏览器后台的js程序，他不影响主程序的运行，是另开的一个js线程，可以用这个线程执行复杂的数据操作，然后把操作结果通过postMessage传递给主线程，这样在进行复杂且耗时的操作时就不会阻塞主线程了。

（3）对html5的语义话的理解

html5的语义化指的是用正确的标签包含正确的内容，比如nav标签，里面就应该包含导航条的内容，而不是用做其他的用途，标签语义化的好处就是结构良好，便于阅读，方便威化，也有利于爬虫的查找，提高搜索率。

（4）cookie，sessionStorage，localeStorage的区别

cookie是存储在浏览器端，并且随浏览器的请求一起发送到服务器端的，它有一定的过期时间，到了过期时间自动会消失。sessionStorage和localeStorage也是存储在客户端的，同属于web Storage，比cookie的存储大小要大有8m，cookie只有4kb，localeStorage是持久化的存储在客户端，如果用户不手动清除的话，不会自动消失，会一直存在，sessionStorage也是存储在客户端，但是它的存活时间是在一个回话期间，只要浏览器的回话关闭了就会自动消失。

（5）多个页面之间如何进行通信

使用cookie，使用web worker，使用localeStorage和sessionStorage

（6）浏览器的渲染过程

1、首先获取html，然后构建dom树

2、其次根据css构建render树，render树中不包含定位和几何信息

3、最后构建布局数，布局是含有元素的定位和几何信息

（7）重构、回流

浏览器的重构指的是改变每个元素外观时所触发的浏览器行为，比如颜色，背景等样式发生了改变而进行的重新构造新外观的过程。重构不会引发页面的重新布局，不一定伴随着回流，

回流指的是浏览器为了重新渲染页面的需要而进行的重新计算元素的几何大小和位置的，他的开销是非常大的，回流可以理解为渲染树需要重新进行计算，一般最好触发元素的重构，避免元素的回流；比如通过通过添加类来添加css样式，而不是直接在DOM上设置，当需要操作某一块元素时候，最好使其脱离文档流，这样就不会引起回流了，比如设置position：absolute或者fixed，或者display：none，等操作结束后在显示。

二、JavaScript部分

（1）JavaScript的数据类型

基本数据类型：Number，String，Boolean，Undefined，Null

复杂数据类型：Object，Array，Function，RegExp，Date，Error

全局数据类型：Math

（2）JavaScript的闭包

闭包简单的说就是一个函数能访问外部函数的变量，这就是闭包，比如说：

1. function a(x){
2. var tem=3;
3. function b(y){
4. console.log(x+y+(++tem));
5. }
6. }

a函数中的b函数就是闭包了，b函数可以使用a函数的局部变量，参数，最典型的闭包应该是下面这样，将定义在函数中的函数作为返回值

1. function a(x){
2. var tem=3;
3. function b(y){
4. console.log(x+y+(++tem));
5. }
6. return b;
7. }

闭包的缺点是，因为内部闭包函数可以访问外部函数的变量，所以外部函数的变量不能被释放，如果闭包嵌套过多，会导致内存占用大，要合理使用闭包。

（3）new 操作符到底做了什么

首先，new操作符为我们创建一个新的空对象，然后this变量指向该对象，

其次，空对象的原型执行函数的原型，

最后，改变构造函数内部的this的指向

代码如下：

1. var obj={};
2. obj.\_\_proto\_\_=fn.prototype;
3. fn.call(obj);

（4）改变函数内部this指针的指向函数

call和apply，假设要改变fn函数内部的this的指向，指向obj，那么可以fn.call(obj);或者fn.apply(obj);那么问题来了，call和apply的区别是什么，其是call和apply的区别在于参数，他们两个的第一个参数都是一样的，表示调用该函数的对象，apply的第二个参数是数组，是[arg1,arg2,arg3]这种形式，而call是arg1,arg2,arg3这样的形式。还有一个bind函数，

var bar=fn.bind(obj);那么fn中的this就指向obj对象了，bind函数返回新的函数，这个函数内的this指针指向obj对象。

（5）JavaScript的作用域和作用域链

JavaScript的作用域指的是变量的作用范围，内部作用域由函数的形参，实参，局部变量，函数构成，内部作用域和外部的作用域一层层的链接起来形成作用域链，当在函数内部要访问一个变量的时候，首先查找自己的内部作用域有没有这个变量，如果没有就到外部的作用域中去查找，还是没有的话，就到在外面一层作用域中找，直到到window所在的作用域，每个函数在声明的时候就默认有一个外部作用域链存在了，比如：

1. var t=4;
2. function foo(){
3. var tem=12;
4. funciton bar(){
5. var temo=34;
6. console.log(t+" "+tem+" "+temo);
7. }
8. }

在bar函数中找t变量的过程就是，先到自己的内部作用域中找，发现没有找到，然后到bar所在的最近的外部环境中找，也就是foo的内部作用域，还是没有找到，再到window的作用域中找，结果找到了

（6）JavaScript的继承

1. function A(name){ this.name=name; }
2. A.prototype.sayName=function(){ console.log(this.name); }
3. function B(age){ this.age=age; }

**原型继承**

1. B.prototype=new A("mbj"); *//被B的实例共享*
2. var foo=new B(18);
3. foo.age; *//18,age是本身携带的属性*
4. foo.name; *//mbj，等价于foo.\_\_proto\_\_.name*
5. foo.sayName(); *//mbj,等价于foo.\_\_proto\_\_.proto\_\_.sayName()*
6. foo.toString(); *//"[object Object]",等价于foo.\_\_proto\_\_.\_\_proto\_\_.\_\_proto\_\_.toString();*

这样B通过原型继承了A，在new B的时候，foo中有个隐藏的属性\_\_proto\_\_指向构造函数的prototype对象，在这里是A对象实例，A对象里面也有一个隐藏的属性\_\_proto\_\_,指向A构造函数的prototype对象，这个对象里面又有一个\_\_proto\_\_指向Object的prototype

这种方式的缺第一个缺点是所有子类共享父类实例，如果某一个子类修改了父类，其他的子类在继承的时候，会造成意想不到的后果。第二个缺点是在构造子类实例的时候，不能给父类传递参数。

**构造函数继承**

1. function B(age,name){ this.age=age;A.call(this,name); }
2. var foo=new B(18,"wmy");
3. foo.name; *//wmy*
4. foo.age; *//18*
5. foo.sayName(); *//undefined*

采用这种方式继承是把A中的属性加到this上面，这样name相当于就是B的属性，sayName不在A的构造函数中，所以访问不到sayName。这种方法的缺点是父类的prototype中的函数不能复用。

**原型继承+构造函数继承**

1. function B(age,name){ this.age=age;A.call(this,name); }
2. B.prototype=new A("mbj");
3. var foo=new B(18,"wmy");
4. foo.name; *//wmy*
5. foo.age; *//18*
6. foo.sayName(); *//wmy*

这样就可以成功访问sayName函数了，结合了上述两种方式的优点，但是这种方式也有缺点，那就是占用的空间更大了。

（7）JavaScript变量提升

请看下面代码

1. var bar=1;
2. function test(){
3. console.log(bar); *//undeifned*
4. var bar=2;
5. console.log(bar); *//2*
6. }
7. test();

为什么在test函数中会出现上述结果呢，这就是JavaScript的变量提升了，虽然变量bar的定义在后面，不过浏览器在解析的时候，会把变量的定义放到最前面，上面的test函数相当于

1. function test(){
2. var bar;
3. console.log(bar); *//undefined*
4. bar=2;
5. console.log(bar); *//2*
6. }

再看

1. var foo=function(){ console.log(1); }
2. function foo(){ console.log(2); }
3. foo(); *//结果为1*
4. 同样的，函数的定义也会到提升到最前面，上面的代码相当于
5. function foo(){ console.log(2); }
6. var foo;
7. foo=funciton(){ console.log(1); }
8. foo(); *//1*

（8）JavaScript事件模型

原始事件模型，捕获型事件模型，冒泡事件模型，

原始事件模型就是ele.onclick=function(){}这种类型的事件模型

冒泡事件模型是指事件从事件的发生地（目标元素），一直向上传递，直到document，

捕获型则恰好相反，事件是从document向下传递，直到事件的发生地（目标元素）

IE是只支持冒泡事件模型的，下面是兼容各个浏览器的事件监听代码

1. EventUtil={
2. addListener:function(target,type,handler){
3. if(target.addEventListener){
4. target.addEventListener(type,handler);
5. }else if(target.attachEvent){
6. target.attach("on"+type,function(){
7. handler.call(target); *//让handler中的this指向目标元素*
8. });
9. }else{
10. target["on"+type]=handler;
11. }
12. },
13. removeListener:function(target,type,handler){
14. if(target.removeEventListener){
15. target.removeEventListener(type,handler);
16. }else if(target.detachEvent){
17. target.detachEvent("on"+type,handler);
18. }else{
19. target["on"+type]=null;
20. }
21. },
22. getEvent:function(e){ *//获取事件对象*
23. var evt=window.event||e;
24. return evt;
25. },
26. getTarget:function(e){ *//获得目标对象*
27. var evt=EventUtil.getEvent(e);
28. var target;
29. if(evt.target){ target=evt.target;}
30. else {target=evt.srcElement;}
31. return target;
32. },
33. stopPropagation:function(e){ *//停止冒泡*
34. var evt=EventUtil.getEvent(e);
35. if(evt.stopPropagation) {evt.stopPropagation();}
36. else {evt.cancelBubble=true;}
37. },
38. preventDefault:function(e){ *//阻值默认行为的发生*
39. var evt=EventUtil.getEvent(e);
40. if(evt.preventDefault){ evt.preventDefault(); }
41. else {e.returnValue=false;}
42. }
43. }

（9）内存泄漏

内存泄漏指的是浏览器不能正常的回收内存的现象

（10）浏览器的垃圾回收机制

垃圾收集器必须跟踪哪个变量有用哪个变量没用，对于不再有用的变量打上标记，以备将来收回其占用的内存，内存泄露和浏览器实现的垃圾回收机制息息相关， 而浏览器实现标识无用变量的策略主要有下两个方法：

**第一，引用计数法**

跟踪记录每个值被引用的次数。当声明一个变量并将引用类型的值赋给该变量时，则这个值的引用次数就是1。如果同一个值又被赋给另一个变量，则该值的引用次 数加1.相反，如果包含对这个值引用的变量又取得另外一个值，则这个值的引用次数减1.当这个值的引用次数变成0时，则说明没有办法访问这个值了，因此就 可以将其占用的内存空间回收回来。

1. 如： var a = {}; *//对象{}的引用计数为1*
2. b = a; *//对象{}的引用计数为 1+1*
3. a = null; *//对象{}的引用计数为2-1*

所以这时对象{}不会被回收;

IE 6, 7 对DOM对象进行引用计数回收， 这样简单的垃圾回收机制，非常容易出现循环引用问题导致内存不能被回收， 进行导致内存泄露等问题，一般不用引用计数法。

**第二，标记清除法**

到2008年为止，IE,Firefox,Opera,Chrome和Safari的javascript实现使用的都是标记清除式的垃圾收集策略（或类似的策略），只不过垃圾收集的时间间隔互有不同。

标记清除的算法分为两个阶段，标记(mark)和清除(sweep). 第一阶段从引用根节点开始标记所有被引用的对象，第二阶段遍历整个堆，把未标记的对象清除。

（11）同源策略

同源策略是浏览器有一个很重要的概念。所谓同源是指，域名，协议，端口相同。不同源的客户端脚本(javascript、ActionScript)在没明确授权的情况下，不能读写对方的资源。简单的来说，浏览器允许包含在页面A的脚本访问第二个页面B的数据资源，这一切是建立在A和B页面是同源的基础上。

（12）跨域的几种方式

jsonp（利用script标签的跨域能力）跨域、websocket（html5的新特性，是一种新协议）跨域、设置代理服务器（由服务器替我们向不同源的服务器请求数据）、CORS（跨源资源共享，cross origin resource sharing）、iframe跨域、postMessage(包含iframe的页面向iframe传递消息)，document.domain跨域（比如：在一个文件中设置了document.domain="[http://qq.com](http://link.zhihu.com/?target=http%3A//qq.com)",那么另一个设置了document.domain="[http://qq.com](http://link.zhihu.com/?target=http%3A//qq.com)"的，他们两个就是同源）

（13）异步和同步

同步指下一个程序的执行需要等到上一个程序执行完毕，也就是得出结果后下一个才能执行，

异步指的是上一个程序指向后，下一个程序不用等到上一个程序出结果就能执行，等上一个出结果了调用回调函数处理结果就好。

（14）JavaScript的值类型和引用类型

JavaScript有两种类型的数据，值类型和引用类型，一般的数字，字符串，布尔值都是值类型，存放在栈中，而对象，函数，数组等是引用类型，存放在堆中，对引用类型的复制其实是引用复制，相当于复制着地址，对象并没有真正的复制。

1. var a=5;var b=a;a=null; *//那么b是5*
2. var a={},var b=a;b.name="mbj";
3. console.log(a.name); *//mbj，因为a，b指向同一个对象*
4. a=null;console.log(typeof b); *//object，a=null，只是a不再指向该对象，但是这个对象还是在堆中确确实实的存在，b依然指向它。*

（15）浏览器js解析引擎的两个队列

请看下面代码

1. console.log(1); *//(1)*
2. setTimeout(function(){
3. console.log(2); *//(2)*
4. },0)
5. console.log(3); *//(3)*
6. *//输出的结果是1,3,2*

可能会有人疑问，为什么不是输出1,2,3，这是因为上面代码在执行的时候被放到js解析引擎的同步队列中，然后先执行语句(1),在把setTimeout的回调函数放到异步队列，然后再执行语句(3),这样同步队列里面就没有代码需要执行了，然后在执行异步队列中的回调函数。

浏览器的js解析引擎在解析js代码的时候，把代码放入到两个队列，放入到同步队列中的代码会优先被执行，放入异步队列中的代码等同步队列中的代码被执行完了之后才会执行。需要异步等待回调的代码一般都是放到异步队列中的。

如果能明白下面代码，说明你对setTimeout，异步和同步队列掌握的比较好了

1. for(var i=1;i<=4;i++){
2. var time=setTimeout(function(i){
3. clearTimeout(time);
4. console.log(i);
5. },1000,i);
6. }
7. *//输出结果1,2,3*
8. *//PS:setTimeout的第三个以及第三个后面的参数都是分别传给setTimeout的回调函数的*

（16）封装cookie的添加，删除，查询方法

cookie是存储在浏览器端的，可以用于存储sessionID，也可以用于自动登陆，记住密码等，但是在浏览器端并没有官方的操作cookie的方法，下面我们来封装一下：

1. CookieUtil=｛
2. addCookie:function(key,value,options){
3. var str=key+"="+escape(value);
4. if(options.expires){
5. var curr=new Date(); *//options.expires的单位是小时*
6. curr.setTime(curr.getTime()+options.expires\*3600\*1000);
7. options.expires=curr.toGMTString();
8. }
9. for(var k in options){ *//有可能指定了cookie的path，cookie的domain*
10. str+=";"+k+"="+options[k];
11. }
12. document.cookie=str;
13. },
14. queryCookie:function(key){
15. var cookies=document.cookie;
16. *//获得浏览器端存储的cookie,格式是key=value;key=value;key=value*
17. cookies+=";";
18. var start=cookies.indexOf(key);
19. if(start<=-1){ return null; } *//说明不存在该cookie*
20. var end=cookies.indexOf(";",start);
21. var value=cookies.slice(start+key.length+1,end);
22. return unescape(value);
23. },
24. deleteCookie:function(key){
25. var value=CookieUtil.queryCookie(key);
26. if(value===null){return false;}
27. CookieUtil.addCookie(key,value,{expires:0});*//把过期时间设置为0，浏览器会马上自动帮我们删除cookie*
28. }
29. ｝

（17）事件委托机制

事件委托指的是，不再事件的发生地设立监听函数，而是在事件发生地的父元素或者祖先元素设置监听器函数，这样可以大大提高性能，因为可以减少绑定事件的元素，比如：

1. <ul>
2. <li></li>
3. <li></li>
4. <li></li>
5. </ul>

要给li元素绑定click事件，使用事件委托机制的话，就只需要给ul绑定click事件就行了，这样就不需要给每个li'绑定click事件，减小内存占用，提高效率，有兴趣的童鞋可以去看看jQuery的live，bind，on，delegate函数的区别，这几个函数就采用了事件委托机制。

三、其他部分

（1）http状态码

http状态码是表示服务器对请求的响应状态，主要分为以下几个部分

1\*\*：这类响应是临时响应，只包含状态行和某些可选的响应头信息，并以空行结束

2\*\*：表示请求成功，

3\*\*：表示重定向

4\*\*：表示客户端错误

5\*\*：表示服务器端错误

100（continue），客户端应当继续发送请求。这个临时响应是用来通知客户端它的部分请求已经被服务器接收

200（OK），表示请求成功，请求所希望的响应头或数据体将随此响应返回。

202（Accepted），服务器已接受请求，但尚未处理。

204（No-Content），服务器成功处理了请求，但不需要返回任何实体内容

205（Reset-Content），服务器成功处理了请求，且没有返回任何内容。但是与204响应不同，返回此状态码的响应要求请求者重置文档视图。该响应主要是被用于接受用户输入后，立即重置表单，以便用户能够轻松地开始另一次输入。

206（Partial-Content），服务器已经成功处理了部分 GET 请求。

301（Moved-Permanently），永久性重定向

302（Moved-Temporarily），暂时性重定向

304（Not-Modified），浏览器端缓存的资源依然有效

400（Bad-Reques），请求有误，当前请求无法被服务器理解。

401（Unauthorized），当前请求需要用户验证。

403（Forbidden），服务器已经理解请求，但是拒绝执行它。

404（Not-Found），请求的资源没有被找到

500（Interval Server Error），服务器内部错误

502（Bad GateWay），网关出错

503（Service Unavailable），由于临时的服务器维护或者过载，服务器当前无法处理请求。

504（Gateway Timeout），作为网关或者代理工作的服务器尝试执行请求时，未能及时从上游服务器（URI标识出的服务器，例如HTTP、FTP、LDAP）或者辅助服务器（例如DNS）收到响应。

（2）xss，csrf的概念以及防范方法

大公司如bat在面试的时候，web安全问题是必问的问题，所以一定要懂，要彻底理解xss和csrf的概念和防范方式，最好在项目中有用到对这两种攻击的防范，这样会给你的面试加很多分。由xss和csrf涉及的东西比较多，我就不具体给出了，详情请看[XSS攻击及防御](http://link.zhihu.com/?target=http%3A//blog.csdn.net/ghsau/article/details/17027893)，[CSRF攻击](http://link.zhihu.com/?target=http%3A//www.cnblogs.com/hyddd/archive/2009/04/09/1432744.html)

（3）CommonJs，AMD，CMD规范

对于前端模块化来说，这三个规范是必须要了解的，详情请看我的这篇文章[CommonJS，AMD，CMD](https://zhuanlan.zhihu.com/p/22954387)

（4）谈谈对前端模块化的理解

前端模块话就是把复杂的文件分成一个个独立的模块，比如js文件，分成独立的模块之后有利于代码的重用和维护，但是这样又会引来模块与模块之间的依赖问题，所以就有了CommonJS、AMD、CMD规范，最后出现了webpack，webpack就是前端模块话的一种解决方案，基本上大公司都会使用webpack，想要详细的学习webpack的话请看[webpack简明使用教程](https://zhuanlan.zhihu.com/p/23538138)

（5）优雅降级和渐进增强

优雅降级指的是一开始就构建功能完好的网站，然后在慢慢兼容低版本的浏览器，使得各个浏览器之间的差异不要太大。

渐进增强是指在基本功能得到满足的情况下，对支持新特性的浏览器使用新特性，带给用户更换的体验。

优雅降级和渐进增强的出发点不同，前者是慢慢向下兼容，是向后看，后着是慢慢向上，增强功能，是向前看。

（6）前端优化（提高网页的加载速度）

1、使用css sprites，可以有效的减少http请求数

2、使用缓存

3、压缩js，css文件，减小文件体积

4、使用cdn，减小服务器负担

5、懒加载图片

6、预加载css，js文件

7、避免dom结构的深层次嵌套

8、给DOM元素添加样式时，把样式放到类中，直接给元素添加类，减少重构，回流

更多详细的前端优化请看[前端优化：雅虎35条](http://link.zhihu.com/?target=http%3A//www.tuicool.com/articles/J3uyaa)或者[前端性能优化----yahoo前端性能团队总结的35条黄金定律](http://link.zhihu.com/?target=http%3A//www.cnblogs.com/lei2007/archive/2013/08/16/3262897.html)

四、前端学习文章推荐

知乎上面有人推荐了很多前端学习网站，具体信息请看

[关于 Javascript 学习，有哪些好的博客或者网站推荐？](https://www.zhihu.com/question/19651401/answer/46211739)