目录

[一、题库 1](#_Toc480749371)

[1.Front-end-Interview-questions 1](#_Toc480749372)

[综合 1](#_Toc480749373)

[CSS相关问题 7](#_Toc480749374)

[Html部分 7](#_Toc480749375)

[二、零散 7](#_Toc480749376)

[1.7个去伪存真的JavaScript面试题 7](#_Toc480749377)

[2. 80%应聘者都不及格的JS面试题 10](#_Toc480749378)

[3. 我是如何同时拿到阿里和腾讯offer的 11](#_Toc480749379)

[4.2015阿里巴巴秋季校园招聘前端 在线笔试题 16](#_Toc480749380)

[5.2015阿里巴巴秋季校园招聘前端在线笔试题--第二波 16](#_Toc480749381)

[6.魅族前端面试题 - 尝试做 16](#_Toc480749382)

# 一、题库

## 1.Front-end-Interview-questions

### 综合

1.一些开放性题目

2.position的值，relative和absolute分别是相对于谁进行定位的？

absolute:生成绝对定位元素，相对于最近一级的定位不是static的父元素，进行定位。

Fixed(老IE不支持):生成绝对定位元素，相对于浏览器窗口进行定位。

Relative:生成相对定位元素，相对于其在普通流中的位置进行定位。

Static:默认值，没有定位，元素出现在正常的流中

3.如何解决跨域问题？

Hack方式

JSONP

Window.name

Document.domain

非hack方式

CORS

Window.postMessage

websocket

4.XML和JSON的区别？

1).数据体积方面

2).数据交互方面

3).数据描述方面

JSON较差

4).传输速度方面

5.谈谈你对webpack的看法

Webpack是一个模块打包工具，包中可以包含JS CSS HTML等类型的文件

Webpack的两大特色：

1.code splitting实现分割代码，按需加载(可以自动完成)

2.loader 可以处理各种类型的资源

Webpack与Grunt Gulp区别

Gulp是自动化任务构建工具，webpack是模块化解决方案

Webpack与seajs、require.js区别

Sea.js/require.js是一种在线“编译”模块方案，相当于在页面上加载一个模块化解释器

Browserify/webapck是一种预编译模块方案

下面的内容可以选择性回答：

Webpack是加强版的browserify(非浏览器环境commonJS的browserify)

Webpack优势

Require.js的所有功能它都有

编译过程更快，因为require.js会去处理不需要的文件

Webpack是以commonJS的形式来书写脚本的，但对AMD/CMD 的支持也很全面，方便旧项目进行代码迁移

Webpack具有requireJS和browserify的功能，但任由很多自己的新特性：

1. 对commonJS、AMD、ES6的语法做了兼容
2. 对js css 图片的等资源文件都支持打包
3. 串联式模块加载器以及插件机制，让其具有更好的灵活性和扩展性，例如提供对CoffeeScript ES6的支持
4. 有独立的配置文件 webpack.config.js
5. 可以将代码切割成不同的chunk,实现按需加载，降低了初始化时间
6. 支持SourceUrls和SourceMaps,易于调试
7. 具有强大的plugin接口，大多是内部插件，使用起来比较灵活
8. Webpack使用异步IO并具有多级缓存，这使得webpack很快且在增量遍以上更加快

6.说说TCP传输的三次握手四次挥手策略

7.TCP和UDP的区别

TCP是基于连接的协议，在正式收发数据前，必须和对方建立一个可靠的连接

UDP面向非链接的协议。

8.说说你对作用域链的理解

变量对象：变量、函数声明、函数形参

上下文：全局上下文 函数上下文 eval上下文

变量对象在每次进入上下文时创建，对于全局上下文，变量对象是全局对象；对于函数，变量对象是活动对象。

根据编译原理，

作用域的两种工作模式：词法作用域和动态作用域，其中词法作用域为Javascript所采用。

根据《你不知道的Javascript》上卷

作用域：根据名称查找变量的规则

作用域链：以当前作用域（活动对象）为最前端开始向上，直到抵达最末端：全局作用域（全局对象），形成一条作用域链（变量对象链）

查询变量对象中的标识符就是沿着作用域链向上寻找。

遮蔽效应:内部标识符遮蔽外部标识符

查找一级标识符：词法作用域查找只会查找一级标识符，找到变量后，对象属性访问规则会接管对属性的访问。

9．Onmouseover和onmousemove的区别

时间上：先触发onmousemove,再触发onmouseover

动作上：onmouseover只在刚进入区域时触发，onmousemove除了刚进入区域触发外，在区域内移动鼠标，也会触发

Mouseenter和mouseover

类似dragenter 和dragover

对于mouseover mouseenter，  
1.如果没有子元素，两者并没有区别  
2.如果存在子元素，  
 在父元素上绑定事件，进入父元素，都会触发事件，并没有区别；  
 对于绑定mouseover的元素，当鼠标从父元素进入其子元素或者子元素进入父元素以及子元素进入子元素，都会触发父元素上的mouseover事件

10.创建AJAX过程

1.创建XHR对象

2.创建一个新的HTTP请求（连接服务器）

3.设置响应HTTP请求状态变化的函数

4.发送HTTP请求

5.获取异步回调返回的数据

6.局部刷新

11.渐进增强和优雅降级

以功能的完整性为主线、浏览器兼容性作为选择。

渐进增强：先针对低版本浏览器构建页面，保证最基本的功能，再针对高级刘燃气追加功能，得到更好的用户体验

优雅降级：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

12.常见web安全及防护原理

SQL注入

CSRF

XSS

13.Web Worker和Websocket

14.HTTP和HTTPS

http协议运行在TCP之上，传输内容是文明

https运行在SSL/TLS之上，传输内容经过加密。

15.为什么HTTPS安全

基于SSL/TSL协议传输

16.对前端模块化的认识

AMD是RequireJS在推广过程中对模块定的规范化产出

CMD是SeaJS在推广过程中对模块定义的规范化产出

17.Javascript垃圾回收方法

标记清除

引用计数

18.你觉得前端工程的价值体现在哪？

19.谈谈性能优化问题

代码层面

20.移动端性能优化

尽量使用CSS3动画，开启硬件加速。

适当使用touch事件代替click事件

避免使用CSS3渐变阴影效果

21.Expires和Cache-Control

Expires要求客户端和服务端的时钟严格同步。

HTTP1.1引入Cache-Control来克服Expires头的限制

如果max-age和Expires同时出现，则max-age有更高的优先级

22.Etag应用

ETAG由服务端生成，客户端通过IF-Match或者说If-None-Match这个条件判断请求来验证资源是否修改。

为什么使用Etag请求头？

23.栈和队列的区别？

24.栈和堆的区别？

堆有程序员控制，栈由计算机控制

25.快速排序的思想并实现一个快排？

26.你觉得jQuery或zepto源码有哪些写的好的地方？

1．jQuery源码封装在一个匿名函数的自执行环境中，有助于防止变量的全局污染

2.通过传入window对象参数，可以是window对象作为局部变量使用，

好处是当jQuery中访问window对象的时候，就不用将作用域链退回到顶层作用了，从而更快的访问window对象

3.jQuery链式调用可以节约代码，提高代码效率

27.ES6的了解

28.js继承方式及其优缺点

29.关于Http2.0你知道多少？

http2.0引入了“服务端推”的概念

header压缩

多路复用

流量控制

提供更多的加密支持

https://github.com/creeperyang/blog/issues/23

30.谈谈浮动和清除浮动

没有设定高度，需要清除浮动

31.如何评价AngularJS和Backbone.js

32.用过哪些设计模式？

33.构造函数模式

1.构造函数方法没有显示的创建对象（new Object）

2.直接将属性和方法赋值给this对象

3.没有return语句

34.说说你你对闭包的理解

35.请你谈谈Cookie的弊端

36.浏览器本地存储

37.Web Storage和cookie的区别

38.cookie和session的区别

### CSS相关问题

1.Display:none和visibility:hidden的区别？

2.CSS中link和@import的区别是？

Link属于标签，@import是CSS提供的

页面加载时，link会被同时加载，而@import引用的CSS会等到页面被夹在完再加载

3.position:absolute和float属性的异同

共同点：脱离文档流

不通点：float仍会占据位置，position会覆盖文档流中的其他元素

4.介绍一下box-sizing属性？

5.CSS选择符哪些？哪些属性可以继承？优先级算法如何计算？CSS3新增伪劣有哪些？

6.CSS3有哪些新特性？

7.对BFC诡诞的理解？

### Html部分

1. 说说你对语义化的理解？
2. Doctype作用？严格模式与混杂模式如何区分？它们有何意义？
3. 你知道多少种Doctype文档类型？
4. HTML与XHTML 二者有什么区别
5. 常见兼容性问题？
6. 上下margin重合问题
7. 解释下浮动和它的工作原理？清除浮动的技巧
8. 浮动元素引起的问题和解决办法？
9. 清除浮动的几种方法
10. DOM操作---怎样添加、移除、移动、复制、创建和查找节点。
11. HTML5有哪些新特性

## 最近遇到的前端面试题(2017.02.23更新版)

1. 输入url后的加载过程

1.DNS解析获得IP地址

2.建立连接（TCP三次握手）

3.构建网页

http请求html,下载html

解析html,根据需要请求其他资源文件：CSS、JS、图片等

渲染器渲染过程:

1. 根据HTML构建DOM树
2. 根据CSS构建CSSOM树
3. 将DOM树和CSSOM树合成渲染树
4. 生成布局layout
5. 浏览器根据布局调用底层的C++绘图函数，绘制的屏幕上（V8 mfc）

4.断开连接（TCP的四次挥手）

1. 有了解过Common.js吗
2. 有了解过React.js吗
3. Angular与vue的区别
4. Less的特点

变量 嵌套 混含 函数 运算

1. Less的原理

包含一套自定义语法及解析器

1. Gulp的特点

自动化构建工具

1. Ajax的原理
2. 有了解过ES6吗？
3. Git merge

Git merge –no-ff 要合并的分支名

用于合并分支

与Git rebase区别

相同分支用rebase 不同分支用git merge

1. Less不依靠构建转CSS

Koala lessc

1. 冒泡、快排
2. Promise

三态:resolve reject pending

Promise.all promise.race

Aync await

1. 性能优化

# 二、零散

## 1.7个去伪存真的JavaScript面试题

1.1创建Javascript对象的两种方法是什么？

单例模式

var o = {name:’zhangld’}

new 关键字(构造器创建对象)

function Hello(x){this.x = x}

var o = new Hello(1)//new 返回引用

追问：

new关键字做了什么？

<http://blog.liuwanlin.info/newzhi-shao-gan-liao-5jian-shi/>

function Base() {  
 this.str='aa'  
 }  
 // 1.创建空对象  
 var obj = {}  
// 2.将空对象的\_\_proto\_\_指向构造函数的prototype  
 obj.\_\_proto\_\_= Base.prototype;  
// 3.使用空对象作为上下文调用构造函数  
 Base.call(obj)

使用new关键字，什么情况下创建对象？

组合模式

Object.create()//只能拷贝方法

var Person = {  
 name:'zhanglongde',  
 create: function (n) {  
 this.name= n  
 }  
}  
var o = Object.create(Person)  
o.create('Hi')

Object.assign()//属性和方法都能拷贝

var Person = {  
 name:'zhanglongde',  
 create: function (n) {  
 this.name= n  
 }  
}  
var o = Object.***assign***(Person)

手动工厂模式

var Person = (function (p) {  
 p.name = 'zhanglongde'  
  
 p.init = function (n) {  
 p.name = n  
 }  
 return p  
})(Person || {})  
  
window.onload = function () {  
 Person.init('zld')  
}

Class Point{

Constructor(x,y){this.x= x;this.y=y} //通过new命令生成对象实例时，自动调用该方法

}

1.2如何创建数组？

JS数组是稀疏的

使用方括号创建

Var arr = []

Var a = Array()

数组去重方法<http://web.jobbole.com/83425/>

Array.from(new Set(arr))

[…new Set(arr)]

叹为观止的ES6 Array

1.3什么是变量提升

变量声明提升，赋值不提升

Var 和 function提升

提升到作用域头部

1.4全局变量有什么风险，以及如何保护代码不受干扰？

变量名污染

工厂模式

Var o = (funcstion(o){ …. return o;})(o || {})

1.5如何通过Javascript对象中的成员变量迭代？

For (key in obj){

if (obj.hasOwnProperty(key)) {

Obj[key]

}

}

1.6什么是闭包？

有权访问另一函数作用域内变量的函数

1.7请描述你经历过的JAV阿Script单元测试

## 2. 80%应聘者都不及格的JS面试题

for(var i = 0; i<5; i++){  
 setTimeout(function () {  
 console.log(new Date, i)  
 },1000)  
}  
console.log(new Date, i)

## 3. 我是如何同时拿到阿里和腾讯offer的

腾讯面试题

1. 请介绍你的项目
2. XMLHttpRequest

创建XHR

连接服务器

发送

回调

1. Http状态码

200 204

304 Not modified

400参数错误 404

500服务器内部错误

1. Cache-control
2. CSS动画原理

补间动画 缓动曲线

逐帧动画 step

Js 物理动画 摩擦力

1. 圣杯布局

三栏布局 浮动

如何使 中间布局在最前面显示 用到负边距

1. ie的某些兼容性

浮动ie产生的双倍距离

1. 项目
2. 跨域

Hack方法：JSONP window.name 图像ping

非hack方法 CORS postMessage SSE websocket sessionStorage

1. JavaScript数据类型
2. 页面加载

输入地址到呈现

Window.onload $()

1. 字符串转化
2. Jsonp原理

JSON外层过上函数

1. CSS合并方法
2. 盒子模型
3. 定位

淘宝面试题

1. 自我介绍
2. 事件委托
3. 模块化工具的特点
4. Session
5. Ajax跨域
6. This
7. 长期规划
8. 前端怎么实现模块化
9. 最熟悉的框架
10. 前后端协同开发

前后端分离 AJAX websocket

服务端渲染

1. 前端学习路线
2. 基础算法
3. Javascript动画算法
4. 拖拽实现

原生js mouseenter mouseover mouseend

H5 dragstart dragend

Dragenter进入目标元素 dragleave离开目标元素

Dragover拖拽元素在目标元素上移动 必须阻止默认行为

Drop拖拽元素在目标元素上，同时松开鼠标

1. Javascript继承实现

前端性能优化

1. 文件合并
2. CSS精灵
3. 避免页面跳转
4. Javascript延迟加载
5. 减少DOM元素数量：会加大页面加载和脚本执行的效率
6. 浏览器一般对同一个域的下载链接数有所限制，按照域名划分下载内容可以使浏览器增大并行下载链接，但是主语控制域名使用在2-4个之间，不然dns查询也是个问题
7. 一般网站规划会将静态资源放在类似于static.example.com,动态内容放在[www.example.com](http://www.example.com)上。这样做还有一个好处是可以在静态的域名商避免使用cookie.后面我们会在cookie的规则中提到。

每次客户端请求都会带上cookie

1. 404我们都不陌生，代表服务器没有找到资源，我们要特别注意404的情况不要在我们提供的网页资源上，客户端发送一个 请求但是服务器却返回一个无用的结果，时间浪费掉了。
2. 使用CDN
3. 使用缓存:cache-control和expires
4. Gzip压缩传输文件
5. 配置etag
6. 避免空的src：空的图片src仍然会使浏览器发送请求到服务器，这样完全是浪费时间，而且浪费服务器的资源。尤其是你的网站每天被很多人访问的时候，这种空请求造成的上海不容忽略。
7. 减小cookie大小：cookie被用来做认证或个性化设置，其信息被包含在http报文头中，对于cookie我们要注意以下几点，来提高请求的响应速度
8. CSS样式表重置：将样式表(css)放在网页和HEAD中会让网页显示加载速度更快，因为这样做可以使浏览器逐步加载已将下载的网页内容。这对内容比较多的网页尤其重要，用户不用一直等待在一个白屏上，而是可以先看已经下载的内容。

如果将样式表放在底部，浏览器会拒绝渲染已经下载的网页，因为大多数浏览器在实现时都努力避免重绘，样式表中的内容是绘制网页的关键信息，没有下载下来之前只好对不起观众了。

1. CMD vs AMD

CommonJS是服务器端模块的规范，node.js采用了这个规范。

根据CommonJS规范，一个单独的文件就是一个模块，也就是说，在该模块内部定义的变量，无法被其他模块读取，除非定义为global对象的属性。

CommonJS规范加载模块是同步的，也就是说，只有加载完成，才能执行后面的操作。

CMD:sea.js

ADM规范则是非同步加载模块，允许制定回调函数。由于Node.js主要用于服务器编程，模块文件一般都已经存在于本地磁盘，所以加载起来比较快，不用考虑非同步加载的方式，不用考虑非同步加载的方式，所以CommonJS规范比较适用。

但是，如果是浏览器环境，要从服务器端加载模块，这是就必须采用非同步模式，因此浏览器端一般采用AMD规范。

AMD:require.js 异步

前端安全

CSRF

简介：跨站请求为战。

攻击者盗用了你的身份，以你的名义发送恶意请求。他可以做到的事情包括：以你的名义发送邮件、发消息、盗取你的账号、甚至购买商品…造成的问题包括：个人隐私泄露以及财产安全。

原理：登录受信任的网站A，并且在本地生成cookie，在不登出A网站的情况下，访问危险网站B。

常见类型：

1. 使用get更新请求资源，容易在B使用img标签造成攻击
2. 前端虽然改成了post请求，但是B网站通过iframe等手段同样造成了攻击

防御机制

1. 验证码：用户每次提交都需要在表单中填写图片上一个随机的字符串
2. 添加隐藏域token:添加一个隐藏的输入框，包含token，服务端验证是否匹配添加
3. 使用HTTP refer头部进行判断，如果不是业务域名发起的HTTP请求，直接过滤。

XSS

XSS是什么？他的全名是：cross-site scripting,为了和CSS层叠样式区分所以取名XSS。是一种网站应用程序的安全漏洞攻击，是代码注入的一种。

它允许恶意用户将代码注入到网页上，其他用户在观看网页时就会受到影响，这类攻击通常包含了HTML以及用户端脚本语言。

例子：

## 4.2015阿里巴巴秋季校园招聘前端 在线笔试题

## 5.2015阿里巴巴秋季校园招聘前端在线笔试题--第二波

<http://www.cnblogs.com/yuanzm/p/3938373.html>

## 6.魅族前端面试题 - 尝试做

<https://segmentfault.com/a/1190000003118598>

1. 列举3个HTML5标签，3个CSS3新特性，3个ECMAScript5新API

Nav section header footer

Transition animation gradient border-radius

String.prototype.trim()

Array.prototype.isArray

Array.prototype.forEach

1. 2种方式，实现某DIV元素以50px每秒的速度左移100px

### 7.阿里前端笔试总结

1.有一个长度未知的数组a,如果它的长度为0就把数字1添加到数组里面，否则按照先进先出的队列规则让第一个元素出队列。

Arr.length === 0 ? arr.push(0) : arr.shift()

2.下面代码输出是什么

var test = (function(a) {

this.a = a;

return function(b) {

return this.a + b;

}

} (function(a, b) {

return a;

}(1, 2)));

console.log(test(4)); //输出什么？？？？

3.请把**<ul><li>第1行</li><li>第2行</li>...</ul>（ul之间有10个li元素）插入body里面，注意：需要考虑到性能问题。**

// 解法1：innerHTML  
 var ul = document.createElement('ul')  
  
 var lis =''  
 //li以字符串形式生成  
 for(var i = 0; i<= 10; i++){  
 lis += '<li>第' + i + '行</li>';  
 }  
 //最后通过innerHTML插入ul里面  
 ul.innerHTML = lis  
  
 document.body.appendChild(ul)  
  
   
   
// 解法2：documentFragment  
 function CreateFragments(){  
  
 var fragment = document.createDocumentFragment();  
  
 for(var i = 0;i < 10000;i++){  
  
 var tmpNode = document.createElement("div");  
 tmpNode.innerHTML = "test" + i + "<br />";  
 fragment.appendChild(tmpNode);  
 }  
  
 document.body.appendChild(fragment);  
 }

4.不使用loop循环，创建一个长度为100的数组，并且每个元素的值等于它的下标。

// 稀疏数组，没有从0开始的连续的index  
var r = new Array(10).map((ele,index) => index)  
var result =[...new Array(10)].map((ele,index) => index)

5.实现对数组进行乱序

// 解法1：数组sort方法+Math.random()  
 var a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
 var sign = 1;  
 a.sort(function (a, b) {  
 // 因为Math.random产生的数在0-1之间  
 //所以0.5两边的概率是相等的  
 //大于0.5是升序，小于0.5是为奖序  
 sign = (Math.random()>0.5)?1:-1  
 return (a-b)\*sign  
 })  
  
 //解法2:loadash的shuffle方法  
 var r = \_.shuffle(a)

6.有一个长度为100的数组，轻易优雅的方式求出该数组的前10个元素之和

var a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15];  
  
var sum = a.slice(0,10).reduce((s,n)=>s+n)