目录

[call()和apply()的区别 1](#_Toc34240421)

[1、 JavaScript里arguments究竟是什么？ 2](#_Toc34240422)

[3、 请解释JSONP的工作原理，以及它为什么不是真正的AJAX 2](#_Toc34240423)

[4、简单介绍下防抖和节流 4](#_Toc34240424)

[5、var、let、const的区别 4](#_Toc34240425)

[6、从输入URL到浏览器显示页面发生了什么。 5](#_Toc34240426)

[7、说说你对Promise的理解？ 5](#_Toc34240427)

[8、移动端开发的时候，一般怎么实现自适应？ 7](#_Toc34240428)

[9、link和@import的区别 7](#_Toc34240429)

[10、请描述一下 cookies，sessionStorage 和 localStorage 的区别？ 8](#_Toc34240430)

[15、 delete和Vue.delete删除数组的区别 8](#_Toc34240431)

[16、vuex有哪几种属性？ 8](#_Toc34240432)

[17、display:none和visibility:hidden的区别？ 9](#_Toc34240433)

[18、介绍一下CSS的盒子模型？ 9](#_Toc34240434)

[19、CSS 选择符有哪些？哪些属性可以继承？优先级算法如何计算？ CSS3新增伪类有那些？ 10](#_Toc34240435)

[20、对BFC规范的理解？ 11](#_Toc34240436)

[21、clear:both的作用？原理？优缺点？ 11](#_Toc34240437)

[22、什么是ajax? ajax的步骤？ 12](#_Toc34240438)

[23、 什么是闭包，如何使用它，为什么要使用它？ 13](#_Toc34240439)

[24、请解释一下JavaScript的同源策略。 14](#_Toc34240440)

[25、谈谈你对webpack的看法 14](#_Toc34240441)

[26、什么是 FOUC（无样式内容闪烁）？你如何来避免 FOUC？ 15](#_Toc34240442)

[27、说说你对AMD和Commonjs的理解 15](#_Toc34240443)

[28、请解释什么是事件代理 16](#_Toc34240444)

# call()和apply()的区别

#### 1、实际上，apply和call的功能是一样的，只是传入的参数列表形式不同。apply：最多只能有两个参数——新this对象和一个数组argArray。

# JavaScript里arguments究竟是什么？

#### 2、Javascrip中国每个函数都会有一个Arguments对象实例arguments，它引用着函数的实参，可以用数组下标的方式"[]"引用arguments的元素。arguments.length为函数实参个数，arguments.callee引用函数自身。

在函数代码中，使用特殊对象arguments，开发者无需明确指出参数名，通过使用下标就可以访问相应的参数。

function test() {

var s = "";

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

alert(arguments[i]);

s += arguments[i] + ",";

}

return s;

}

test("name", "age");//name,age

复制代码

arguments虽然有一些数组的性质，但其并非真正的数组，只是一个类数组对象。其并没有数组的很多方法，不能像真正的数组那样调用.jion(),.concat(),.pop()等方法。

# 请解释JSONP的工作原理，以及它为什么不是真正的AJAX

#### 3、JSONP (JSON with Padding)是一个简单高效的跨域方式，HTML中的script标签可以加载并执行其他域的javascript，于是我们可以通过script标记来动态加载其他域的资源。例如我要从域A的页面pageA加载域B的数据，那么在域B的页面pageB中我以JavaScript的形式声明pageA需要的数据，然后在 pageA中用script标签把pageB加载进来，那么pageB中的脚本就会得以执行。JSONP在此基础上加入了回调函数，pageB加载完之后会执行pageA中定义的函数，所需要的数据会以参数的形式传递给该函数。JSONP易于实现，但是也会存在一些安全隐患，如果第三方的脚本随意地执行，那么它就可以篡改页面内容，截获敏感数据。但是在受信任的双方传递数据，JSONP是非常合适的选择。

1. AJAX是不跨域的，而JSONP是一个是跨域的，还有就是二者接收参数形式不一样！
2. 原理是：动态插入script标签，通过script标签引入一个js文件，这个js文件载入成功后会执行我们在url参数中指定的函数，并且会把我们需要的json数据作为参数传入。
3. 由于同源策略的限制，XmlHttpRequest只允许请求当前源（域名、协议、端口）的资源，为了实现跨域请求，可以通过script标签实现跨域请求，然后在服务端输出JSON数据并执行回调函数，从而解决了跨域的数据请求。
4. 优点是兼容性好，简单易用，支持浏览器与服务器双向通信。缺点是只支持GET请求。
5. JSONP：json+padding（内填充），顾名思义，就是把JSON填充到一个盒子里
6. CORS：
7. 服务器端对于CORS的支持，主要就是通过设置Access-Control-Allow-Origin来进行的。如果浏览器检测到相应的设置，就可以允许Ajax进行跨域的访问。
8. 通过修改document.domain来跨子域
9. 将子域和主域的document.domain设为同一个主域.前提条件：这两个域名必须属于同一个基础域名!而且所用的协议，端口都要一致，否则无法利用document.domain进行跨域
10. 主域相同的使用document.domain
11. 使用window.name来进行跨域
12. window对象有个name属性，该属性有个特征：即在一个窗口(window)的生命周期内,窗口载入的所有的页面都是共享一个window.name的，每个页面对window.name都有读写的权限，window.name是持久存在一个窗口载入过的所有页面中的
13. 使用[HTML5](http://lib.csdn.net/base/html5" \o "HTML5知识库)中新引进的window.postMessage方法来跨域传送数据
14. 还有flash、在服务器上设置代理页面等跨域方式。个人认为window.name的方法既不复杂，也能兼容到几乎所有浏览器，这真是极好的一种跨域方法。

# 4、简单介绍下防抖和节流

#### 4、何为防抖 多次触发事件后，事件处理函数只执行一次，并且是在触发操作结束时执行，一般用于scroll事件解决原理 对处理函数进行延时操作，若设定的延时到来之前再次触发事件，则清除上一次的延时操作定时器，重新定时。

# 5、var、let、const的区别

#### 5、var 会变量提升；

* let 声明的变量只在它所在的代码块有效；
* const 声明后不能再修改其指向的目标，假如const 指向的是一个对象/数组，那么虽然不能更改指向目标，但是可以更改对象和数组内部的值；

进阶一：说到变量提升，class 声明一个类时，存在变量提升么？为什么？

* 不存在。因为要方便类的继承，先声明子类再声明父类；

进阶二：const 声明一个对象，如何让对象内部的属性的值也无法改变？

* 使用Object.freeze()锁死（es5新增特性）；
* 对数组等引用类型的值，还是能修改的；

进阶三：全局作用域？函数作用域？块级作用域？作用域链？

# 6、从输入URL到浏览器显示页面发生了什么。

#### 6、1.在浏览器中输入url（解析IP地址）

* 2.应用层DNS解析域名
* 3.应用层客户端发送HTTP请求
* 4.传输层TCP传输报文(3次握手)
* 5.网络层IP协议查询MAC地址
* 6.数据到达数据链路层
* 7.服务器接收数据
* 8.服务器响应请求
* 9.服务器返回相应文件
* 二、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。

# 7、说说你对Promise的理解？

依照 Promise/A+ 的定义，Promise 有四种状态：

pending: 初始状态, 非fulfilled 或 rejected.

fulfilled: 成功的操作.

rejected: 失败的操作.

settled: Promise已被fulfilled或rejected，且不是pending

另外， fulfilled 与 rejected 一起合称 settled。

Promise 对象用来进行延迟(deferred) 和异步(asynchronous ) 计算。

Promise 的构造函数

构造一个 Promise，最基本的用法如下：

    var promise = new Promise(function(resolve, reject) {

        if (...) {  // succeed

           resolve(result);

        } else {   // fails

            reject(Error(errMessage));

        }

    });

Promise 实例拥有 then 方法（具有 then 方法的对象，通常被称为 thenable）。它的使用方法如下：

promise.then(onFulfilled, onRejected)

接收两个函数作为参数，一个在 fulfilled 的时候被调用，一个在 rejected 的时候被调用，接收参数就是 future，onFulfilled对应 resolve, onRejected 对应 reject。

是异步编程的一种解决方案；  
所谓Promise，简单说就是一个容器，里面保存着某个未来才会结束的事件（通常是一个异步操作）的结果。从语法上说，Promise 是一个对象，从它可以获取异步操作的消息。  
一般什么时候使用？

处理异步请求时使用，比如ajax请求；  
有哪些状态？

状态有pending、resolved、rejected；  
怎么捕获他的错误？

最后写catch；  
then里面第二个参数可以捕获；  
如果内部抛错，但是没有被捕获，这个没被捕获的错误接下来会发生什么事情？

冒泡；  
能不能被try catch捕获？为什么？

不能，因为是异步编程。  
能不能被window.onerror捕获？

不能  
new Promise((resolve,reject)=>throw new Error('a')).then(fn1).then(fn2,fn3).catch(fn4) 这个函数，会执行哪些函数 \* fn3，没了

# 8、移动端开发的时候，一般怎么实现自适应？

* rem
* vw和vh
* 媒体查询（bootstrap）；

进阶一：他们的特点分别是什么？或者说实现原理是什么？

* rem 根据 html 的 font-size；
* vw是浏览器窗口宽度，vh是高度；
* 媒体查询是根据浏览器窗口宽度或高度，进行响应式选择显示哪个css；

进阶二：弹出输入框会发生定位错误，fixed布局，怎么解决？

1、如果使用的元素不是必须使用fixed布局的话可以不使用；

2、元素必须使用fixed布局时，在input获得焦点时将元素的position设置为static，失去焦点时再将元素的position设置为fixed；

# 9、link和@import的区别

两者都是外部引用 CSS 的方式，但是存在一定的区别：

（1）link是XHTML标签，除了能够加载CSS，还可以定义RSS等其他事务；而@import属于CSS范畴，只可以加载CSS。

（2）link引用CSS时，在页面载入时同时加载；@import需要页面完全载入以后再加载。

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

（4）link支持使用Javascript控制DOM改变样式；而@import不支持。

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

#### 、cookie是网站为了标示用户身份而储存在用户本地终端（Client Side）上的数据（通常经过加密）。

cookie数据始终在同源的http请求中携带（即使不需要），记会在浏览器和服务器间来回传递。

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

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。

有期时间：

    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

# 15、 delete和Vue.delete删除数组的区别

delete只是被删除的元素变成了 empty/undefined 其他的元素的键值还是不变。

Vue.delete直接删除了数组 改变了数组的键值。

# 16、vuex有哪几种属性？

有五种，分别是 State、 Getter、Mutation 、Action、 Module（分包）

vuex的State特性

A、Vuex就是一个仓库，仓库里面放了很多对象。其中state就是数据源存放地，对应于一般Vue对象里面的data

B、state里面存放的数据是响应式的，Vue组件从store中读取数据，若是store中的数据发生改变，依赖这个数据的组件也会发生更新

C、它通过mapState把全局的 state 和 getters 映射到当前组件的 computed 计算属性中

vuex的Getter特性

A、getters 可以对State进行计算操作，它就是Store的计算属性

B、 虽然在组件内也可以做计算属性，但是getters 可以在多组件之间复用

C、 如果一个状态只在一个组件内使用，是可以不用getters

vuex的Mutation特性

Action 类似于 mutation，不同在于：Action 提交的是 mutation，而不是直接变更状态；Action 可以包含任意异步操作。

# 17、display:none和visibility:hidden的区别？

display:none 隐藏对应的元素，在文档布局中不再给它分配空间，它各边的元素会合拢，

就当他从来不存在。

visibility:hidden 隐藏对应的元素，但是在文档布局中仍保留原来的空间。

# 18、介绍一下CSS的盒子模型？

1）有两种， IE 盒子模型、标准 W3C 盒子模型；IE的content部分包含了 border 和 pading;

2）盒模型： 内容(content)、填充(padding)、边界(margin)、 边框(border).

# 19、CSS 选择符有哪些？哪些属性可以继承？优先级算法如何计算？ CSS3新增伪类有那些？

#### 19 \* 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, text-indent;

\* 不可继承的样式：border padding margin width height ;

\* 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:

!important > id > class > tag

important 比 内联优先级高,但内联比 id 要高

CSS3新增伪类举例：

p:first-of-type 选择属于其父元素的首个 <p> 元素的每个 <p> 元素。

p:last-of-type 选择属于其父元素的最后 <p> 元素的每个 <p> 元素。

p:only-of-type 选择属于其父元素唯一的 <p> 元素的每个 <p> 元素。

p:only-child 选择属于其父元素的唯一子元素的每个 <p> 元素。

p:nth-child(2) 选择属于其父元素的第二个子元素的每个 <p> 元素。

:enabled :disabled 控制表单控件的禁用状态。

:checked 单选框或复选框被选中。

# 20、对BFC规范的理解？

#### BFC，块级格式化上下文，一个创建了新的BFC的盒子是独立布局的，盒子里面的子元素的样式不会影响到外面的元素。在同一个BFC中的两个毗邻的块级盒在垂直方向（和布局方向有关系）的margin会发生折叠。

（W3C CSS 2.1 规范中的一个概念，它决定了元素如何对其内容进行布局，以及与其他元素的关系和相互作用。）

# 21、clear:both的作用？原理？优缺点？

清除浮动

原理：添加一个空div，利用css提高的clear:both清除浮动，让父级div能自动获取到高度   
优点：简单、代码少、浏览器支持好、不容易出现怪问题  
缺点：不少初学者不理解原理；如果页面浮动布局多，就要增加很多空div，让人感觉很不好

# 22、什么是ajax? ajax的步骤？

（(1)创建XMLHttpRequest对象,也就是创建一个异步调用对象.

(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.

(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.

）

ajax(异步javascript xml) 能够刷新局部网页数据而不是重新加载整个网页。

如何使用ajax?

第一步，创建xmlhttprequest对象，var xmlhttp =new XMLHttpRequest（);XMLHttpRequest对象用来和服务器交换数据。

var xhttp;

if (window.XMLHttpRequest) {

//现代主流浏览器

xhttp = new XMLHttpRequest();

} else {

// 针对浏览器，比如IE5或IE6

xhttp = new ActiveXObject("Microsoft.XMLHTTP");

}

第二步，使用xmlhttprequest对象的open（）和send（）方法发送资源请求给服务器。

第三步，使用xmlhttprequest对象的responseText或responseXML属性获得服务器的响应。

第四步，onreadystatechange函数，当发送请求到服务器，我们想要服务器响应执行一些功能就需要使用onreadystatechange函数，每次xmlhttprequest对象的readyState发生改变都会触发onreadystatechange函数。

# 什么是闭包，如何使用它，为什么要使用它？

1. 我的理解是，闭包就是能够读取其他函数内部变量的函数。
2. 由于在Javascript语言中，只有函数内部的子函数才能读取局部变量，因此可以把闭包简单理解成"定义在一个函数内部的函数"。
3. 所以，在本质上，闭包就是将函数内部和函数外部连接起来的一座桥梁。
4. 闭包的用途
5. 闭包可以用在许多地方。它的最大用处有两个，一个是前面提到的可以读取函数内部的变量，另一个就是让这些变量的值始终保持在内存中。
6. 使用闭包主要是为了设计私有的方法和变量。闭包的优点是可以避免全局变量的污染，缺点是闭包会常驻内存，会增大内存使用量，使用不当很容易造成内存泄露。在js中，函数即闭包，只有函数才会产生作用域的概念
7. 闭包有三个特性：
8. 1.函数嵌套函数
9. 2.函数内部可以引用外部的参数和变量
10. 3.参数和变量不会被垃圾回收机制回收
11. 闭包就是能够读取其他函数内部变量的函数。由于在Javascript语言中，只有函数内部的子函数才能读取局部变量，因此可以把闭包简单理解成“定义在一个函数内部的函数”。
12. 所以，在本质上，闭包就是将函数内部和函数外部连接起来的一座桥梁。闭包可以用在许多地方。它的最大用处有两个，一个是前面提到的可以读取函数内部的变量，另一个就是让这些变量的值始终保持在内存中。
13. 使用闭包的注意点：
14. · 由于闭包会使得函数中的变量都被保存在内存中，内存消耗很大，所以不能滥用闭包，否则会造成网页的性能问题，在IE中可能导致内存泄露。解决方法是，在退出函数之前，将不使用的局部变量全部删除。
15. · 闭包会在父函数外部，改变父函数内部变量的值。所以，如果你把父函数当作对象（object）使用，把闭包当作它的公用方法（Public Method），把内部变量当作它的私有属性（private value），这时一定要小心，不要随便改变父函数内部变量的值。
16. function func(param) {
17. return function() {
18. alert(param);
19. }
20. }
21. var f = func(1)
22. setTimeout(f, 1000);

# 24、请解释一下JavaScript的同源策略。

在客户端编程语言中，如javascript和 ActionScript，同源策略是一个很重要的安全理念，它在保证数据的安全性方面有着重要的意义。同源策略规定跨域之间的脚本是隔离的，一个域的脚本不能访问和操作另外一个域的绝大部分属性和方法。那么什么叫相同域，什么叫不同的域呢？当两个域具有相同的协议, 相同的端口，相同的host，那么我们就可以认为它们是相同的域。同源策略还应该对一些特殊情况做处理，比如限制file协议下脚本的访问权限。本地的HTML文件在浏览器中是通过file协议打开的，如果脚本能通过file协议访问到硬盘上其它任意文件，就会出现安全隐患，目前IE8还有这样的隐患。

# 25、谈谈你对webpack的看法

WebPack 是一个模块打包工具，你可以使用WebPack管理你的模块依赖，并编绎输出模块们所需的静态文件。它能够很好地管理、打包Web开发中所用到的HTML、[JavaScript](http://lib.csdn.net/base/javascript" \o "JavaScript知识库)、CSS以及各种静态文件（图片、字体等），让开发过程更加高效。对于不同类型的资源，webpack有对应的模块加载器。webpack模块打包器会分析模块间的依赖关系，最后 生成了优化且合并后的静态资源。

webpack的两大特色：

1.code splitting（可以自动完成）

2.loader 可以处理各种类型的静态文件，并且支持串联操作

webpack 是以commonJS的形式来书写脚本滴，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，但仍有很多自己的新特性：

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 很快且在增量编译上更加快

# 26、什么是 FOUC（无样式内容闪烁）？你如何来避免 FOUC？

FOUC - FlashOf Unstyled Content 文档样式闪烁

     <styletype="text/css"media="all">@import"../fouc.css";</style>

    而引用CSS文件的@import就是造成这个问题的罪魁祸首。IE会先加载整个HTML文档的DOM，然后再去导入外部的CSS文件，因此，在页面DOM加载完成到CSS导入完成中间会有一段时间页面上的内容是没有样式的，这段时间的长短跟网速，电脑速度都有关系。

     解决方法简单的出奇，只要在<head>之间加入一个<link>或者<script>元素就可以了。

# 27、说说你对AMD和Commonjs的理解

CommonJS是服务器端模块的规范，[Node.js](http://lib.csdn.net/base/nodejs" \o "Node.js知识库)采用了这个规范。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。AMD规范则是非同步加载模块，允许指定回调函数。

AMD推荐的风格通过返回一个对象做为模块对象，CommonJS的风格通过对module.exports或exports的属性赋值来达到暴露模块对象的目的。

document.write()的用法

document.write()方法可以用在两个方面：页面载入过程中用实时脚本创建页面内容，以及用延时脚本创建本窗口或新窗口的内容。

document.write只能重绘整个页面。innerHTML可以重绘页面的一部分

# 28、请解释什么是事件代理

事件代理（Event Delegation），又称之为事件委托。是 JavaScript 中常用绑定事件的常用技巧。顾名思义，“事件代理”即是把原本需要绑定的事件委托给父元素，让父元素担当事件监听的职务。事件代理的原理是DOM元素的事件冒泡。使用事件代理的好处是可以提高性能。