**说一下正向代理和反向代理的理解或者说是区别。**

**正向代理：**

正向代理就是用户想从服务器拿资源数据，但是只能通过proxy服务器才能拿到，所以用户A只能去访问proxy服务器然后通过proxy服务器去服务器B拿数据，这种情况用户是明确知道你要访问的是谁，在我们生活中最典型的案例就是“翻墙“了，也是通过访问代理服务器最后访问外网的。

**反向代理：**

反向代理其实就是客户端去访问服务器时，他并不知道会访问哪一台，感觉就是客户端访问了Proxy一样，而实则就是当proxy关口拿到用户请求的时候会转发到代理服务器中的随机（算法）某一台。而在用户看来，他只是访问了Proxy服务器而已，典型的例子就是负载均衡了。

**DOM是什么?怎样创建、添加、移动、复制、和查找节点？**

创建：createElement("元素节点名");

createTextNode("文本节点");

添加：appendChild;insertAfter//添加到后面

insertBefore//添加到前面

移动：replaceChild;

复制：cloneNode;

查找：hasChildNodes();

**检测数据类型有哪几种方法？有什么特点？**

（1）**typeof 它返一个字符串,里面包含了对应的数据类型,**

**局限性:检测的不管是数组还是正则都返回的 是"object",所以typeof不能判断一个值是否为数组。**

（2）**instanceof/constructor检测某一个实例是否属于这个类**，**可以检测数组和正则。**

**局限性:用instanceof 检测的时候,只要当前的这个类的实例在原型链上**, **检测出来的结果都是true**，**检测的结果其实是不准确的。**

（3）**Object.prototype.toString.call(value)**

**它常用来判断对象值属于哪种内置属性,它返回一个JSON字符串**——"[object 数据类型]"。

**数组中新增的方法：数组中forEach() 、map()、filter()、some()、every()**

以上6个方法均为ES6语法，IE9及以上才支持。不过可以通过babel转意支持IE低版本。 以上均不改变原数组。  
some、every返回true、false。   
map、filter返回一个新数组。  
forEach 无返回值。

**请你谈谈Cookie的弊端？**

（1）增加流量消耗，每次请求都需要带上cookie信息。

（2）cookie使用明文传输，所以更具有安全性隐患。

如果cookie被人拦截了，那人就可以取得所有的session信息。

（3）Cookie数量和长度的限制。每个domain最多只能有20条cookie，每个cookie长度不能超过4KB，否则会被截掉。

**Web strorage与cookie相比存在的优势是什么？**

优势: 存储空间更大

存储内容不会发送到服务器

存储空间是完全独立的，因此不会造成数据混乱。

更多丰富易用的接口

缺点：存储在本地的数据未加密而且永远不会过期，极易造成隐私泄漏。

**请描述一下cookies,sessionStorage和localStorage的区别？**

（1）数据的生命周期不同：

Cookie ：一般由服务器生成，可设置过期时间。

如果在浏览器端生成Cookie，默认是关闭浏览器后Cookie被销毁

localStorage :除非被销毁，否则永久保存

sessionStorage :关闭页面或关闭浏览器后被销毁

（2）存放数据大小不同 :

cookie : 4KB左右

localStorage 和 sessionStorage : 一般为5MB

（3）与服务器端通信不同：

Cookie : 每次都会携带在HTTP头中,如果使用cookie保存过多数据会带来性能问题

localStorage和sessionStorage:仅在浏览器中保存，不参与和服务器的通信

----------------------------------------------------------------------------

（4）易用性方面也有所不同 ：

Cookie : 需要程序员自己封装，源生的Cookie接口不友好

localStorage 和 sessionStorage ：源生接口可以接受，也可再次封装来对Object和Array有更好的支持；

**cookie和session的区别：**

相同点：

1. 可以存取数据；
2. 整个网站所有页面都有效；
3. Cookie和session都是用户私有的；

不同点：

1. cookie可以长期存储，而session是浏览器关闭就销毁；
2. cookie存储到浏览器中，session存储到服务器中；

**script标签中的defer属性是下载完成就执行，而async是渲染完再执行，多个脚本情况下，defer可以保证加载顺序，这样的说法对吗**

defer异步加载顺序，遇到脚本加载脚本，和dom创建同时进行，

当dom创建完毕 ，才执行脚本

（一般用于dom操作之类的）

async异步加载顺序，遇到脚本加载脚本，和dom创建同时进行，

与defer区别是，脚本加载完毕就执行

**强制缓存和协商缓存的区别**

强缓存：不会向服务器发送请求，直接从缓存中读取资源

协商缓存：向服务器发送请求，服务器会根据这个请求的request header的一些参数来判断是否读取协商缓存，如果读取，会返回304状态码并带上新的response header通知浏览器从缓存中读取资源；

两者的共同点是，都是从客户端缓存中读取资源

[**Web项目开发中用到的缓存技术**](https://www.cnblogs.com/sochishun/p/7326752.html)

在WEB开发中用来应付高流量最有效的办法就是用缓存技术，能有效的提高服务器负载性能，用空间换取时间。  
缓存一般用来

* 存储频繁访问的数据
* 临时存储耗时的计算结果
* 内存缓存可以减少磁盘IO

**HTTP 的状态码有哪些？分别代表是什么意思？**

1xx 代表临时响应

2xx 代表响应成功

3xx 代表重定向

4xx 代表客户端错误

5xx 代表服务器错误

常见状态码：

200 服务器成功返回页面；

204 请求成功但是服务器并没有返回数据；

301 永久性定向 （比如这个域名我们永久性的更改，称为永久定向）

302 临时性定向 （比如这个域名我们临时性的更改，称为临时定向）

304 请求的页面未修改；

400 请求错误；

404 请求的页面不存在；

503 服务器不可用；

504 服务器端网关超时。

**http和https的区别**

http是超文本传输协议，信息是明文传输，https则是具有安全性的ssl加密传输协议。

http和https使用的是完全不同的连接方式，用的端口也不一样，前者是80，后者是443。

**http特点**

无状态协议

多次http请求

基于tcp协议

http请求报文：请求方法、地址、发送的数据

http响应报文：版本、状态码、短语

**三次握手**

第一次握手：客户端向服务器发送请求，并发送客户端的初始序列号

　　第二次握手：服务器接收客户端的请求，然后向客户端返回一个响应，并发送序列号

　　第三次握手：客户端接受响应数据，并向服务器发送一个确认，至此客户端与服务器建立连接建立

**四次挥手**

第一次挥手：客户端发送请求给服务器，说明客户端有发送请求的能力

第二次挥手：服务器发送请求给客户端，告诉客户端我正在接收请求，说明服务器具有接收请求的能力

第三次挥手：服务器发送请求给客户端，告诉客户端我接收完毕，说明服务器具有发送请求的能力

第四次挥手：客户端发送请求给服务器，说明客户端有接收请求的能力。然后断开连接。

**一个完整的http请求过程：**

第一步：客户端浏览器建立与服务器的TCP连接

第二步：客户端浏览器向WEB服务器发送请求头信息

第三步：客户端浏览器向WEB服务器发送请求信息

第四步：WEB服务器准备应答

第五步：WEB服务器发送应答头信息

第六步：WEB服务器向浏览器发送数据

第七步：WEB服务器断开TCP连接

**get和post的区别。**

GET原理：数据附加到url?的后面，并且数据和文档头部一起发送。

特点：大小受到限制，不能传送文件

用法：单纯的获取服务器数据，或传少量数据。

POST原理：数据和文档正文一起传送

特点：大小无限制（可以传送文件）

用法：给服务器传送数据或文件

**Ajax工作原理：**

用户发送请求 给 ajax ，ajax再发送给服务器；

服务器响应之后，将数据发送给ajax。Ajax响应给客户端浏览器

**Ajax应用：**登录验证、关键字提示、ajax瀑布流、用户名是否被占用

**Ajax是什么？如何创建一个Ajax对象？Ajax执行流程。**

Ajax用来实现前后端数据交互。

创建一个ajax

IE9+：new XMLHttpRequest()

IE6、7、8：new ActiveXObject("Microsoft.XMLHTTP");

Ajax执行流程

1.创建ajax对象（获取要发送的数据）

2.设置请求的方法和接口地址

3.设置请求的编码

4.通过onreadystatechange事件去监听ajax对象请求过程

5.发送请求

**display:none、visibility:hidden和opacity:0的区别**

1. 是否占据空间这个角度

Display:none;隐藏之后不占据空间

Visibility:hidden;和opacity:0;隐藏之后仍然占据空间

1. 子元素是否继承

Display:none;不会被子元素继承

Visibility:hidden;会被子元素继承，子元素可以通过visibility:visible进行显示

Opacity:0;也会被子元素继承，但是子元素不能通过opacity:1;进行显示

1. 是否触发绑定事件

Display:none;和 Visibility:hidden;无法触发它的绑定事件

Opaccity:0;可以触发它的绑定事件

1. 过渡动画

Transition对于display和visibility是无效的。

对于opacity是有效。

**svg和canvas有了解吗？svg的优势？canvas和svg的区别？**

**SVG 是使用 XML 去描述 2D 图形的一种语言。**

这意味着 SVG DOM 中的每个元素都是可用的。

也可以为某个元素附加 JavaScript 事件处理器。

每个图形都会被视为对象。如果 SVG 对象的属性发生变化，那么浏览器能够自动重现图形。

**Canvas 通过 JavaScript 来绘制 图形，而且是逐像素进行渲染。**

在 canvas 中，一旦图形被绘制完成，它就不会继续得到浏览器的关注。如果其位置发生变化，那么整个场景也需要重新绘制。

Canvas

依赖分辨率

不支持事件处理器

具有弱的文本渲染能力

能够以 .png 或 .jpg 格式保存结果图像

最适合图像密集型的游戏，其中的许多对象会被频繁重绘

SVG

不依赖分辨率

支持事件处理器

最适合带有大型渲染区域的应用程序（比如谷歌地图）

复杂度高会减慢渲染速度

不适合游戏应用

**css预处理用过没？ 它的主要用途和优点呢**

用过less

优点：

**a.**结构清晰。

b.可以屏蔽浏览器私有语法的差异

**C.**可以轻松实现多重继承

1. 完全兼容 CSS 代码，可以应用到老项目中。

**缺点：**

 须要编译。无论是放在客户端还是服务器端，都是一种额外的花销。

**常用的数组方法以及它们 的返回值**

concat()连接两个或更多的数组，并返回结果。

join()把数组的所有元素放入一个字符串。元素通过指定的分隔符进行分隔。

**pop()**删除并返回数组的最后一个元素

**push()**向数组的末尾添加一个或更多元素，**并返回新的长度**。

**reverse()**颠倒数组中元素的顺序。

**shift()**删除并返回数组的第一个元素

**unshift()**向数组的开头添加一个或更多元素，**并返回新的长度**。

slice()从某个已有的数组返回选定的元素

**sort()**对数组的元素进行排序

**splice()**删除元素，并向数组添加新元素。

toString()把数组转换为字符串，并返回结果。

valueOf()返回数组对象的原始值

**常用字符串的方法以及他们的返回值**

charAt(index) 返回index下标对应的字符，位置不存在返回空字符串

charCodeAt(index) 返回index下标对应字符的unicode编码

String.fromCharCode(XXX编码) 返回XXX编码对应的字符串值

concat() 返回连接后的字符串

indexOf(值) 返回值对应的下标，没有检索到返回-1

lastIndexOf(值) 返回值对应的下标，没有检索返回-1

match(/正则表达式/) 返回一个数组，数组元素是由正则匹配到的值组成

replce(/正则表达式/) 返回值是某部分被替换，其他部分保持不变（字符串）

search(/正则表达式/) 返回值是从左到右匹配到的值，对应下标

slice(startIndex,endIndex) （在endIndex下标之前）返回值是截取的字符串

substring(startIndex,endIndex) （在endIndex下标之前）返回值是截取的字符串

substr(startIndex,length) （在length之前） 返回值是截取的字符串

toLowerCase() 将小写字母转大写字母

toUpperCase() 将大写字母转小写字母

split(特定字符) 按照特定字符将字符串转成数组

**为什么使用闭包?（闭包的使用场景）**

一方面是因为函数是一个作用域，如果想要在函数外边使用函数内局部变量就要突破作用域，使用闭包函数。

另一个方面是在面向对象编程中为了实现封装的概念，就是说不允许用户直接操纵对象中的属性，必须通过对象提供的方法来操纵对象中的属性，使用闭包函数。

**都调试过哪些浏览器的兼容性？解决方案**

1. 在div中存图片时，图片下方有3像素bug

解决：img{ vertical-align:top }

1. Window对象属性innerHeight存在兼容

IE9+ innerHeight

IE6,7,8 document.documentElement.clientHeight

1. 通过父节点改变子节点

IE9+ firstElementChild

IE6,7,8 firstChild

1. 更改为下一个兄弟节点

IE9+ nextElementSibling

IE6,7,8 nextSibling

1. 事件监听

IE11+ window.addEventListener

IE6,7,8 window.attachEvent

IE9,10 都支持

1. 事件冒泡与捕获

事件冒泡：e.stopProgation()

IE浏览器：e.cancelBubble=true

捕获事件：preventDefault()

IE6,7,8：return.value=false

1. Json字符串转js对象

IE9+ JSON.parse()

IE6,7,8 eval( “(” +json字符串 “)” )

1. Ajax对象创建

IE9+ xmlhttp=new XMLHttpRequest()

IE6,7,8 xmlhttp=new ActiveXObject("Microsoft.XMLHTTP")

9.const问题  
说明: Firefox下,可以使用const关键字或var关键字来定义常量;  
IE下,只能使用var关键字来定义常量.

**你做页面都在哪些浏览器测试过？这些浏览器的内核分别是什么？**

谷歌公司的谷歌浏览器，它的内核是webkit

微软公司的IE浏览器，它的内核是trident

码字蓝基金会的火狐浏览器，它的内核是gecko

**遍历器iterator**

1.iterator接口就是为了所有数据类型(Array Object Map Set)提供了一种统一的访问机制,

使用for..of会自动去寻找这个接口

2.ES6规定，默认的iterator接口部署在数据结构的symbol.iterator属性，

也就是说一个数据结构只有具有（iterator）属性就可以用for...of和扩展运算符...

3.使用场景：for...of、Array.from()、 Map()、 Set()、 WeakMap()、 WeakSet()、 Promise.all()、 Promise.race()

这些方法都会自动调用这个接口

**CSS引入的方式有哪些？link和@import的区别是什么？**

引入方式:

（1）行内样式

（2）内嵌样式

（3）外联样式

（4）导入样式

Link和@import的区别:

<link>

1、属于XHTML

2、优先加载CSS文件到页面

3、 link没有兼容性 并支持使用javascript改变样式

4、是和html同时进行导入，html导入完成，css导入完成

5、link不仅可以加载css，还可以定义rss，rel连接属性

@import

1、属于CSS2.1

2、先加载HTML结构在加载CSS文件

3、 不支持使用javascript改变样式

4、@import是在html加载完成后加载，用户体验不好

5、@import属于CSS范畴，只能加载CSS

**深拷贝方法。**

1. 递归函数实现深拷贝。递归原理：在递归函数内，用Object.prototype.toString.call（）方法判断数据是对象还是数组，如果说都不是，直接返回，如果是数组，push进数组同时这个数据也需要再调用递归函数做判断，直到数据类型既不是对象也不是数组。如果是对象，将数据添加到对象中，并且也需要再调用递归函数做判断，直到数据类型既不是对象也不是数组
2. Object.assign()
3. JSON.parse(JSON.stringify())

**null和undefined的区别?**

Null代表没有对象，用来初始化一个变量，也可以用来与一个已经初始化的变量比较。

当函数的参数期望是对象时，被用作参数传入。

当函数的返回值是对象时，被用作返回值传出。

作为对象原型链的终点。

Undefined代表缺少值，就是此处应该有值。

变量被声明，但没有赋值，值是undefined

对象没有赋值的属性，属性值是undefined

函数没有返回值，值是undefined

**清除浮动的几种方式,以及使用场景**

1. 使用空标签清除浮动 （理论上能够清除任何标签，增加无意义的标签）
2. clear：both 清除两侧浮动

3、使用overflow：auto

4、使用after伪元素清除浮动（用于非IE浏览器）（使用zoom：1 用于兼容IE）

**xhtml和html有什么区别？**

1.html是一种基本的web网页设计语言；

2.xhtml是一个基于xml的置标语言；

3.xhtml的元素必须被正确的嵌套；

4.xhtml的语言必须被关闭；

5.标签名必须用小写字母；

6.xhtml文档必须拥有根元素；

**new操作符具体干了什么呢？**

创建一个空对象，将对象的\_\_proto\_\_指向构造器的prototype,改变this指向为当前创造对象，判断返回值类型，若为**值类型**则返回obj，若为**引用类型**则返回这个引用类型的对象。

**列举原生查找DOM的方法。（在原生中如何获取DOM元素）**

document .getElementById() // 通过Id，获取DOM元素。

document .getElementsByTagName() //通过标签名,获取DOM元素组成的数组

document .getElementsByName() //通过Name属性的值, 获取DOM元素组成的数组

document.getElementsByClassName(class名)//通过类名获得DOM对象组成的数组----------IE9+

document.querySelector(基本选择器)//通过选择器获得DOM对象

document.querySelectorAll(基本选择器)//通过选择器获得DOM对象组成的数组

**常见的6种JavaScript设计模式**

构造函数模式

工厂模式

模块模式

混合模式

单例模式

[观察者模式（发布订阅模式）](https://www.cnblogs.com/xianyulaodi/p/5827821.html#_label4)

https://www.jianshu.com/p/d8c1c426d028

**如何居中一个浮动元素？ （不定宽高的水平垂直居中）**

**网站重构的理解**

重构：在不改变页面的前提下，简化代码结构、增强代码可读性。

a. 针对CSS做一些兼容处理。

b. 针对于SEO，进行移动平台的优化。

c. 减少代码间的耦合，让代码保持弹性

d. 压缩或合并JS、CSS、image等前端资源

**谈谈以前端角度出发做好SEO需要考虑什么？**

a. 了解搜索引擎如何抓取网页和如何索引网页

b. meta标签优化

c. 关键词分析

d. 付费给搜索引擎

e. 链接交换和链接广泛度（Link Popularity）

f. 合理的标签使用

**一个页面上有大量的图片（大型电商网站），加载很慢，你有哪些方法优化这些图片的加载，给用户更好的体验。**

a. 图片懒加载，滚动到相应位置才加载图片。

b. 图片预加载，如果为幻灯片、相册等，将当前展示图片的前一张和后一张优先下载。

c. 使用CSSsprite， Iconfont、Base64等技术，对图片进行优化。

d. 如果图片过大，可以使用特殊编码的图片，加载时会先加载一张压缩的特别厉害的缩略图，以提高用户体验。

**轮播图的原理**

<https://www.jianshu.com/p/237b58bcafd4>

实现自动播放主要设置一个定时器，每2s自动播放下一张图片，播放到最后一张转到第一张图片时需要使用无缝衔接。

这个无缝衔接采用定位，当播放到最后一张图片时，采用relative将第一张图片定位到最后一张后面，播放完后再拉取到最初的位置。

**从输入url到页面加载完毕经过哪些步骤？**

输入一个网址到页面渲染

1. 用户输入网址
2. 会建立tcp连接
3. 浏览器向服务器发送http请求
4. 服务器接收并响应，向客户端发送数据
5. 浏览器收到数据
6. 开始渲染页面

**变量提升的意思？**

**变量提升**就是把定义在后面的变量或者函数,提升到块作用域的顶端位置,但是,变量提升,只是提升变量的声明,并不会将变量赋的值一同提升上来。

**数组遍历的方法**

1.for循环

2.map即可遍历对象又可以遍历数组，并且是变异方法，会更改原数组

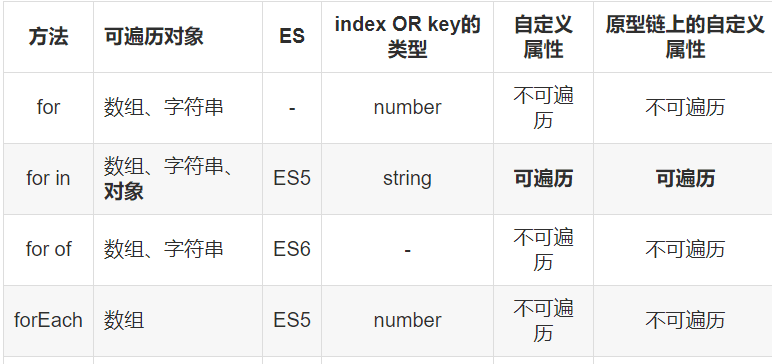
并且有返回值

3.forEach遍历 非变异方法，不会更改原数组，没有返回值

4.for...in 遍历

5.for...of 遍历

**for, for in ,for of, forEach的区别？**



**说几条写JavaScript的基本规范？**

1.不要在同一行声明多个变量。

5.Switch语句必须带有default分支

6.函数不应该有时候有返回值，有时候没有返回值。

7.For循环必须使用大括号

8.If语句必须使用大括号

9.for-in循环中的变量 应该使用var关键字明确限定作用域，从而避免作用域污染。

**如何居中div?居中一个浮动div,如何让绝对定位的div居中?**

**1.**

width: 400px;

height: 18px;

padding:20px;

background: orange;

color: #fff;

position: absolute;

left:0;

right:0;

top: 0;

bottom: 0;

margin: auto;

**2.**

width: 400px;

height: 18px;

padding: 20px;

background: orange;

color: #fff;

position: absolute;

top:50%;

left:50%;

**3.**

width: 500px;

height: 300px;

background: orange;

color: #fff;

/\*只需要在父元素上加这三句\*/

justify-content: center; /\*子元素水平居中\*/

align-items: center; /\*子元素垂直居中\*/

display: -webkit-flex;

4.

padding: 20px;

background: orange;

color: #fff;

position: absolute;

top: 50%;

left: 50%;

border-radius: 5px;  
 transform: translate(-50%, -50%);

**js事件流**

1.冒泡型事件流：事件的传播是从最特定的事件目标到最不特定的事件目标。

即从DOM树的叶子到根。

2.捕获型事件流：事件的传播是从最不特定的事件目标到最特定的事件目标。

即从DOM树的根到叶子。

为什么父级作用域访问不到子作用域？

因为执行上下文栈和根据作用域链，内部作用域是可以访问外部作用域的，并且一层一层的向上访问，直到找到全局作用域为止，但是父级作用域却不能访问子级作用域中的变量和方法

**Html5有哪些新特性、移除了哪些元素？如何处理HTML5新标签的浏览器兼容问题？如何区分HTML和HTML5?**

**解析：**

新特性：

（1） 拖拽释放(Drag and drop) API

（2）语义化更好的内容标签（header,nav,footer,aside,article,section）

（3） 音频、视频API(audio,video)

（4） 画布(Canvas) API

（5） 本地离线存储 localStorage 长期存储数据，浏览器关闭后数据不丢失；

sessionStorage 的数据在浏览器关闭后自动删除

（6） 表单控件，calendar、date、time、email、url、search

移除的元素：

（1） 纯表现的元素：big，center，font,strike，u；

（2） 对可用性产生负面影响的元素：frame，frameset；

支持HTML5新标签：

IE8/IE7/IE6支持通过 document.createElement 方法产生的标签，可以利用这一特性让这些浏览器支持 HTML5 新标签，浏览器支持新标签后，还需要添加标签默认的样式（当然最好的方式是直接使用成熟的框架、使用最多的是html5shim框架）：

如何区分：

DOCTYPE声明新增的结构元素、功能元素

**使用Css预处理器的优缺点？（SASS，Compass ，Stylus，Less）描述下你曾经使用过的CSS预处理器的优缺点。**

解析：（拆装的笔记）

**伪类和伪元素的区别**

我自己的理解：

伪类，类嘛，就类似class类这样，表示一些元素的状态的。

伪元素，就是一个元素，类似p标签，但是实际并不存在于DOM树中。

最简单的理解和区分就是： 伪类前面是一个[冒号](https://wenwen.sogou.com/s/?w=%E5%86%92%E5%8F%B7&ch=ww.xqy.chain)，伪元素前面是两个冒号。E:[first-child](https://wenwen.sogou.com/s/?w=first-child&ch=ww.xqy.chain) 伪类，E::[first-line](https://wenwen.sogou.com/s/?w=first-line&ch=ww.xqy.chain)为元素。特殊情况::before 和 ::after 伪元素， before 和after是在元素前面和后面添加一些元素，因此是伪元素。

**css3新增伪类**

* :empty 选择没有子元素的元素
* :not(selector) 选择除 selector 元素意外的元素
* :enabled 选择可用的表单元素
* :disabled 选择禁用的表单元素
* :after 在元素内部最前添加内容
* :before 在元素内部最后添加内容
* :nth-child(n) 匹配父元素下指定子元素，在所有子元素中排序第n
* :first-child
* :last-child
* :only-child
* :first-line 选择元素中的第一行
* :first-letter 选择元素中的第一个字符

**什么是BFC**

* BFC（Block Formatting Context）格式化上下文，它是一个隔离的独立容器。运用BFC主要是解决父元素塌陷问题，开发过程中用的最多的是overflow:hidden触发BFC。

**形成BFC的条件**

* 1、浮动元素，float 除 none 以外的值；   
        2、定位元素，position（absolute，fixed）；   
        3、display 为以下其中之一的值 inline-block，table-cell，table-caption；   
        4、overflow 除了 visible 以外的值（hidden，auto，scroll）；

**BFC的特性**

* 1.内部的Box会在垂直方向上一个接一个的放置。  
        2.垂直方向上的距离由margin决定  
        3.bfc的区域不会与float的元素区域重叠。  
        4.计算bfc的高度时，浮动元素也参与计算  
        5.bfc就是页面上的一个独立容器，容器里面的子元素不会影响外面元素。

**css画一个三角形**

/\*向上\*/

    width:0;  height:0;  border-width:0 28px 28px;  border-style: solid;

    border-color: transparent transparent #f00 transparent ;/\*透明（上） 透明（右）  红（下） 透明（左）\*/

**prototype 和\_\_proto\_\_区别**

联系：prototype和\_\_proto\_\_都指向原型对象，任意一个函数都有一个prototype属性，指向该函数的原型对象，同样任意一个构造函数实例化的对象，都有一个\_\_proto\_\_属性，指向构造函数的原型对象。

区别：

1. JavaScript中的函数是对象，而且除了使用字面量定义外，都需要通过函数来创建对象；
2. prototype是构造函数访问原型对象，\_\_proto\_\_是对象实例访问原型对象。

# 浏览器兼容问题 （随便挑几个简述）

1. **浏览器兼容问题一：不同浏览器的标签默认的外补丁和内补丁不同**

问题症状：随便写几个标签，不加样式控制的情况下，各自的margin 和padding差异较大。

碰到频率:100%

解决方案：CSS里    \*{margin:0;padding:0;}

备注：这个是最常见的也是最易解决的一个浏览器兼容性问题，几乎所有的CSS文件开头都会用通配符\*来设置各个标签的内外补丁是0。

1. **浏览器兼容问题二：块属性标签float后，又有横行的margin情况下，在IE6显示margin比设置的大**

 问题症状：随便写几个标签，不加样式控制的情况下，各自的margin 和padding差异较大。

碰到频率:100%

解决方案：CSS里    \*{margin:0;padding:0;}

备注：这个是最常见的也是最易解决的一个浏览器兼容性问题，几乎所有的CSS文件开头都会用通配符\*来设置各个标签的内外补丁是0。

1. **浏览器兼容问题三：设置较小高度标签（一般小于10px），在IE6，IE7，遨游中高度超出自己设置高度**

 问题症状：IE6、7和遨游里这个标签的高度不受控制，超出自己设置的高度

碰到频率：60%

解决方案：给超出高度的标签设置overflow:hidden;或者设置行高line-height 小于你设置的高度。

备注：这种情况一般出现在我们设置小圆角背景的标签里。出现这个问题的原因是IE8之前的浏览器都会给标签一个最小默认的行高的高度。即使你的标签是空的，这个标签的高度还是会达到默认的行高。

1. **浏览器兼容问题四：行内属性标签，设置display:block后采用float布局，又有横行的margin的情况，IE6间距bug**

 问题症状：IE6里的间距比超过设置的间距

碰到几率：20%

解决方案：在display:block;后面加入display:inline;display:table;

备注：行内属性标签，为了设置宽高，我们需要设置display:block;(除了input标签比较特殊)。在用float布局并有横向的margin后，在IE6下，他就具有了块属性float后的横向margin的bug。不过因为它本身就是行内属性标签，所以我们再加上display:inline的话，它的高宽就不可设了。这时候我们还需要在display:inline后面加入display:talbe。

1. **浏览器兼容问题五：图片默认有间距**

 问题症状：几个img标签放在一起的时候，有些浏览器会有默认的间距，加了问题一中提到的通配符也不起作用。

碰到几率：20%

解决方案：使用float属性为img布局

备注：因为img标签是行内属性标签，所以只要不超出容器宽度，img标签都会排在一行里，但是部分浏览器的img标签之间会有个间距。去掉这个间距使用float是正道。（我的一个学生使用负margin，虽然能解决，但负margin本身就是容易引起浏览器兼容问题的用法，所以我禁止他们使用）

1. **浏览器兼容问题六：标签最低高度设置min-height不兼容**

 问题症状：因为min-height本身就是一个不兼容的CSS属性，所以设置min-height时不能很好的被各个浏览器兼容

碰到几率：5%

解决方案：如果我们要设置一个标签的最小高度200px，需要进行的设置为：{min-height:200px; height:auto !important; height:200px; overflow:visible;}

备注：在B/S系统前端开时，有很多情况下我们又这种需求。当内容小于一个值（如300px）时。容器的高度为300px；当内容高度大于这个值时，容器高度被撑高，而不是出现滚动条。这时候我们就会面临这个兼容性问题。

1. **浏览器兼容问题七：透明度的兼容CSS设置**

做兼容页面的方法是：每写一小段代码（布局中的一行或者一块）我们都要在不同的浏览器中看是否兼容，当然熟练到一定的程度就没这么麻烦了。建议经常会碰到兼容性问题的新手使用。很多兼容性问题都是因为浏览器对标签的默认属性解析不同造成的，只要我们稍加设置都能轻松地解决这些兼容问题。如果我们熟悉标签的默认属性的话，就能很好的理解为什么会出现兼容问题以及怎么去解决这些兼容问题。

/\* CSS hack\*/

我很少使用hacker的，可能是个人习惯吧，我不喜欢写的代码IE不兼容，然后用hack来解决。不过hacker还是非常好用的。使用hacker我可以把浏览器分为3类：IE6 ；IE7和遨游；其他（IE8 chrome ff safari opera等）

IE6认识的hacker 是下划线\_ 和星号 \*

IE7 遨游认识的hacker是星号 \*

比如这样一个CSS设置：

height:300px;\*height:200px;\_height:100px;

IE6浏览器在读到height:300px的时候会认为高时300px；继续往下读，他也认识\*heihgt， 所以当IE6读到\*height:200px的时候会覆盖掉前一条的相冲突设置，认为高度是200px。继续往下读，IE6还认识\_height,所以他又会覆盖掉200px高的设置，把高度设置为100px；

IE7和遨游也是一样的从高度300px的设置往下读。当它们读到\*height200px的时候就停下了，因为它们不认识\_height。所以它们会把高度解析为200px，剩下的浏览器只认识第一个height:300px;所以他们会把高度解析为300px。因为优先级相同且想冲突的属性设置后一个会覆盖掉前一个，所以书写的次序是很重要的。

简述移动端性能优化方法？

答：

（1）减少HTTP请求

合并CSS、JavaScript、合并小图片，使用雪碧图

（2）Css在头部引入，js在尾部引入，减少页面回流重绘，还能使用懒加载，还有下拉刷新都可以优化性能。

（3）使用缓存可以减少向服务器的请求数，节省加载时间，所以所有静态资源都要在服务器端设置缓存。

（4）使用首屏加载 JavaScript，滚屏加载将不影响首屏的资源和当前屏幕资源，不用的资源放到用户需要时才加载，可以大大提升重要资源的显示速度和降低总体流量。

Lazyload实现的基本原理？

核心原理是：

1 设置一个定时器，计算每张图片是否会随着滚动条的滚动，而出现在视口（也就是浏览器中的**展现网站的空白部分**）中；

2 为<img>标签设置一个暂存图片URL的自定义属性（例如loadpic），当图片出现在视口时，再将loadpic的值赋给图片的src属性；

懒加载的原因：

对于图片过多的场景，为了提高页面的加载速度，降低服务器的负载，增强用户体验，我们对还没出现在视野的图片先不加载，当元素出现在我们视野中的时候再加载。

懒加载的原理：

我们先将img标签中的src链接设置为一样的图片（空白图片），将真正的图片链接放在自定义属性中，如（data-src），当js监听到图片元素进入到可视窗口的时候，将自定义属性中的地址存储到src中，达到懒加载的效果。

一、移动端点击**300ms延迟问题**

双击缩放、双击滚动引起，

解决方案：

1.禁用缩放、更改默认的视口宽度

<meta name="viewport" content="width=device-width,initial-scale=1,minimum-scale=1,maximum-scale=1,user-scalable=no" />

1. fastclick.js

移动端事件触发顺序：touchstart --> touchmove -> touchend -->click。

监听addEventListener("touchend", handle);

原理是：在检测到touchend事件的时候，会通过DOM自定义事件立即出发模拟一个click事件，并把浏览器在300ms之后真正的click事件阻止掉。同样可以解决移动端点击穿透现象。

1. **点击穿透现象现象：**当A/B两个层上下z轴重叠，上层的A点击后消失或移开（这一点很重要），并且B元素本身有默认click事件（如a标签）或绑定了click事件。在这种情况下，点击A/B重叠的部分，就会出现点透的现象。

解决：

（1）尽量都使用touch事件来替换click事件。例如用touchend事件(推荐)。

（2）用fastclick，https://github.com/ftlabs/fastclick

（3）用preventDefault阻止a标签的click

（4）延迟一定的时间(300ms+)来处理事件 （不推荐）

（5）以上一般都能解决，实在不行就换成click事件。

1. **通过transform进行skew变形，rotate旋转会造成出现锯齿现象**
2. **圆角bug**

某些Android手机圆角失效background-clip: padding-box;

**五、旋转屏幕时，字体大小调整的问题**

html, body, form, fieldset, p, div, h1, h2, h3, h4, h5, h6{

-webkit-text-size-adjust:100%;

}

**六、iphone及ipad下输入框默认内阴影**

-webkit-appearance:none;

七、长时间按住页面出现闪退

-webkit-touch-callout:none;

八、**上下拉动滚动条时卡顿、慢**

body {-webkit-overflow-scrolling:

touch;overflow-scrolling: touch;}

Android3+和iOS5+支持CSS3的新属性为overflow-scrolling

九、**format-detection**

format-detection 启动或禁用自动识别页面中的电话号码。

默认情况下，设备会自动识别任何可能是电话号码的字符串。设置telephone=no可以禁用这项功能。

十、**图片加载**

图片加载很慢，手机开发一般用canvas方法加载：

十一、**移动端 HTML5 audio autoplay 失效问题**

苹果、安卓系统通常都会禁止自动播放和使用 JS 的触发播放，必须由用户来触发才可以播放。

解决思路：先通过用户 touchstart 触碰，触发播放并暂停（音频开始加载，后面用 JS 再操作就没问题了）。

解决代码：

document.addEventListener('touchstart',function() {

document.getElementsByTagName('audio')[0].play();

document.getElementsByTagName('audio')[0].pause();

});