# 前端面试题及答案

1. **Doctype作用？标准模式与兼容模式各有什么区别?**

Doctype 主要用于告诉浏览器的解析器该以什么文档标准渲染页面，

不存在或格式不正确文档会以兼容模式呈现

标准模式：标准模式的排版和js的运作模式都是以该浏览器支持的最高标准运行

兼容模式：一些ie怪异现象等

1. **行内元素、块级元素、空元素**

行内： span em b a p

块级： div form ul h1-h5

空 ： br

1. **link和@import的区别**
2. 服务对象不同：link是为当前页面服务，import是对css服务
3. 加载顺序不同：当一个页面被加载的时候（就是被浏览者浏览的时候），link引用的CSS会同时被加载，而@import引用的CSS 会等到页面全部被下载完再被加载。所以有时候浏览@import加载CSS的页面时开始会没有样式（就是闪烁），网速慢的时候还挺明显
4. [兼容性的差别。由于@import是CSS2.1提出的所以老的浏览器不支持，@import只有在IE5以上的才能识别，而link标签无此问题。](mailto:兼容性的差别。由于@import是CSS2.1提出的所以老的浏览器不支持，@import只有在IE5以上的才能识别，而link标签无此问题。)
5. 使用dom控制样式时的差别。当使用javascript控制dom去改变样式的时候，只能使用link标签，因为@import不是dom可以控制的。
6. 介绍一下你对浏览器内核的理解

**渲染引擎：**负责取得网页的内容(、XML、图像等等)、整理讯息(例如加入CSS等)，以及计算网页的显示方式，然后会输出至显示器或打印机。浏览器的内核的不同对于网页的语法解释会有不同，所以渲染的效果也不相同。

**js 引擎：**解析和执行javascript来实现网页的动态效果。

最开始渲染引擎和JS引擎并没有区分的很明确，后来JS引擎越来越独立，内核就倾向于只指渲染引擎。

主流的浏览器使用的渲染引擎：

Trident : IE、360、搜狗浏览器

WebKit : chrome，safari（现在的Chrome使用的Blink内核是webkit的分支）

Gecko : FirFox

Presto :Opera7及以上（Opera现在的内核是Blink）

**5、怎样理解html标签的语义化**

含义：根据内容的结构化（内容语义化），选择合适的标签（代码语义化）便于开发者阅读和写出更优雅的代码的同时让浏览器的爬虫和机器很好地解析。

原因：

1. 为了在没有css的情况下，页面也能呈现出良好的内容结构、代码结构
2. 有利于SEO。和浏览器建立良好的联系，有利于爬虫抓取更多有效的信息，爬虫依赖于标签来确定上下文和各个关键字的权重；
3. 方便其他设备解析（如屏幕阅读器、盲人阅读器、移动设备）以意义的方式来渲染网页；
4. 便于团队的开发和维护
5. **实现不适用border画出高1px高的线（是没有宽度的），在不同浏览器的兼容模式和标准模式下均能最正常使用**

<div style="height:1px;background-color: red;overflow: hidden;">不加overflow的话IE5会不兼容</div>

好像可以使用box-shadow模拟（图解css3）

1. 使用box-shadow模拟遮罩层 但是不可避免的要考虑兼容性
2. **b与strong,i与em的区别**

b i是物理元素，显示效果

Strong em强调文档逻辑的，便于seo优化

物理元素是告诉浏览器我应该以何种格式显示文字，逻辑元素告诉浏览器这些文字有怎么样的重要性。

**9、盒模型**

可通过box-sizing:border-box|content-box|padding-box改变

viewWidth是最终看见的宽度，width为css中设置的宽度的大小

**content-box**与标准盒模型一样 不包括内边距和边框的值，对象的**实际宽度**等于设置的width值和border、padding之和 改变border padding会改变元素的外观大小 viewWidth=width+border+padding

Border-box：padding和border被包含在定义的width和height之内。对象的实际宽度就等于设置的width值，即使定义有border和padding也不会改变对象的实际宽 viewWidth=width(border,padding)

1. **网页验证码是干嘛的，是为了解决什么安全问题？**

可以防止：恶意破解密码、刷票、论坛灌水，有效防止某个黑客对某一个特定注册用户用特定程序暴力破解方式进行不断的登陆尝试，实际上用验证码是现在很多网站通行的方式，我们利用比较简易的方式实现了这个功能。这个问题可以由计算机生成并评判，但是必须只有人类才能解答。由于计算机无法解答CAPTCHA的问题，所以回答出问题的用户就可以被认为是人类。

1. **css的优先级算法怎样计算？**

CSS specificity

1. 选择器权重值的计算

A：如果规则是写在标签的style属性中（内联样式），则A=1，否则，A=0. 对于内联样式，由于没有选择器，所以B、C、D的值都为0，即A=1, B=0, C=0, D=0（简写为1,0,0,0，下同）。

B：计算该选择器中ID的数量。（例如，#header 这样的选择器，计算为0, 1, 0, 0）。

C：计算该选择器中伪类及其它属性的数量（包括class、属性选择器等，不包括伪元素）。 （例如， .logo[id='site-logo'] 这样的选择器，计算为0, 0, 2, 0）。

D：计算该选择器中伪元素及标签的数量。（例如，p:first-letter 这样的选择器，计算为0, 0, 0, 2）。

1. 权重值的比较

按照四组计算的正确方法，上面例子中的样式一权重值应该是0, 0, 0, 11，样式二的权重值是0, 0, 1, 0。

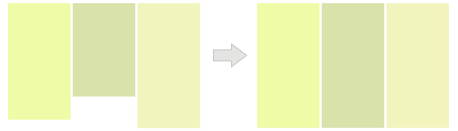
根据规范，计算权重值时，A,B,C,D四组值，从左到右，分组比较，如果A相同，比较B，如果B相同，比较C，如果C相同，比较D，如果D相同，后定义的优先。

1. ！Important的优先级最高

如果多条规则中都对同一个属性指定了 !important 呢？这时候 !important 的作用相互抵销，依然按照ABCD四组计算比较。

因此 !important 的作用只有在具有唯一性时才能提现，但是我们永远无法预料自己什么时候又需要覆盖一个已经指定了 !important 的属性，所以最好的办法就是：不要使用 !important。

1. 总结
2. 一条样式规则的整体权重值包含四个独立的部分：[A, B, C, D];
3. A表示内联样式，只有1或者0两个值；
4. B表示规则中ID的数量；
5. C表示规则中除了ID、标签和伪元素以外的其它选择器数量；
6. D表示规则中标签和伪元素的数量；
7. 比较时从高位到低位（从A到D）分别比较，高位相同才需要比较低位；
8. 有 !important 标记的属性权重值无视没用 !important 指定的一切情况；
9. 多次指定 !important 时，相互抵销。
10. **position的值relative和absolute定位原点是？**
11. **Css实现多列等高？**



1. 一、假等高列

这种方法是我们实现等高列最早使用的一种方法，就是使用背景图片，在列的父元素上使用这个背景图进行Y轴的铺放，从而实现一种等高列的假像：

1. Padding-bottom设置为一个极大的值 margin-bottom 设置负的极大的这个值 外层overflow:hidden
2. **常见的浏览器兼容问题有哪些？**

(1)png24位的图片在iE6浏览器上出现背景，解决方案是做成PNG8。

(2)浏览器默认的margin和padding不同。解决方案是加一个全局的\*{margin:0;padding:0;}来统一。

(3)IE6双边距bug: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。

(4)IE下,可以使用获取常规属性的方法来获取自定义属性,也可以使用getAttribute()获取自定义属性;

Firefox下,只能使用getAttribute()获取自定义属性；

解决方法:统一通过getAttribute()获取自定义属性。

(5)

IE下,even对象有x,y属性,但是没有pageX,pageY属性;

Firefox下,event对象有pageX,pageY属性,但是没有x,y属性；

解决方法：（条件注释）缺点是在IE浏览器下可能会增加额外的HTTP请求数。

1. Chrome 中文界面下默认会将小于 12px 的文本强制按照 12px 显示,可通过加入 CSS 属性 -webkit-text-size-adjust: none; 解决。
2. **li与li之间有看不见的空白间隔是什么原因引起的？有什么解决办法？**

**引起这种空白间隔的原因：**

浏览器的默认行为是把inline元素间的空白字符（空格换行tab）渲染成一个空格，也就是我们上面的代码<li>换行后会产生换行字符，而它会变成一个空格，当然空格就占用一个字符的宽度。

**解决方案：**

**方法一：**既然是因为<li>换行导致的，那就可以将<li>代码全部写在一排。

**方法二：**我们为了代码美观以及方便修改，很多时候我们不可能将<li>全部写在一排，那怎么办？既然是空格占一个字符的宽度，那我们索性就将<ul>内的字符尺寸font-size:0px。

**方法三：**本来以为方法二能够完全解决问题，但经[测试](http://lib.csdn.net/base/softwaretest" \o "软件测试知识库" \t "http://blog.csdn.net/sjinsa/article/details/_blank)，将li父级标签字符设置为0在**Safari浏览器**依然出现间隔空白；既然设置字符大小为0不行，那咱就将间隔消除了，letter-spacing:0。同样随来而来的问题是li内的字符间隔也被设置了，我们需要将li内的字符间隔设为默认normal。

1. **为什么要初始化css样式**

因为浏览器的兼容问题，不同浏览器对有些标签的默认值是不同的，如果没对CSS初始化往往会出现浏览器之间的页面显示差异。

当然，初始化样式会对SEO有一定的影响，但鱼和熊掌不可兼得，但力求影响最小的情况下初始化

1. **absolute 的 containing block（容器块）计算方式跟正常流有什么不同？**

无论属于哪种，都要先找到其祖先元素中最近的 position 值不为 static 的元素，然后再判断：

(1)若此元素为 inline 元素，则 containing block 为能够包含这个元素生成的第一个和最后一个 inline box 的 padding box (除 margin, border 外的区域) 的最小矩形；

(2)否则,则由这个祖先元素的 padding box 构成。

如果都找不到，则为 initial containing block。

补充：

(1)static(默认的)/relative：简单说就是它的父元素的内容框（即去掉padding的部分）

(2)absolute: 向上找最近的定位为absolute/relative的元素

(3)fixed: 它的containing block一律为根元素(html/body)，根元素也是initial containing block.

**18、谈谈你对 BFC 规范（块级格式化上下文：block formatting context）的理解？**

（W3C CSS 2.1 规范中的一个概念,它是一个独立容器，决定了元素如何对其内容进行定位,以及与其他元素的关系和相互作用，不受外界的影响，只有Block-level box参与。）一个页面是由很多个 Box 组成的,元素的类型和 display 属性,决定了这个 Box 的类型。不同类型的 Box,会参与不同的 Formatting Context（决定如何渲染文档的容器）,因此Box内的元素会以不同的方式渲染,也就是说BFC内部的元素和外部的元素不会互相影响。

Position:relative不能触发BFC

那么哪些情况下能触发BFC呢？

1. **什么是浮动，什么时候需要清除浮动，清除浮动都有哪些方法？**

父元素内子元素浮动，且高度设为auto或没有高度,那么父元素的高度会塌陷。

清除浮动的方法：

**方法一**：添加新的元素 应用 clear：both；

.clear{clear:both; height: 0; line-height: 0; font-size: 0}

**方法二：**父级div定义 overflow: auto，zoom：1//兼容

**方法三：** 据说是最高大上的方法 :after 方法：（注意：作用于浮动元素的父元素）

.outer {zoom:1;} /\*==for IE6/7 Maxthon2==\*/.outer:after {clear:both;content:'.';display:block;width: 0;height: 0;visibility:hidden;} /\*==for FF/chrome/opera/IE8==\*/

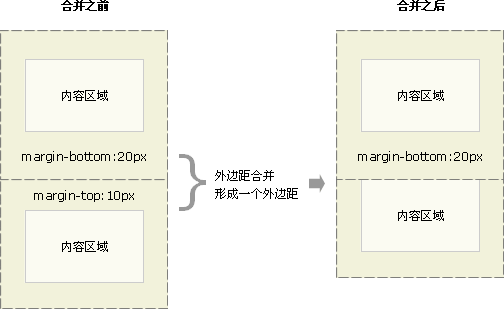
1. **什么是外边距合并？**

## 外边距合并指的是，当两个垂直外边距相遇时，它们将形成一个外边距。

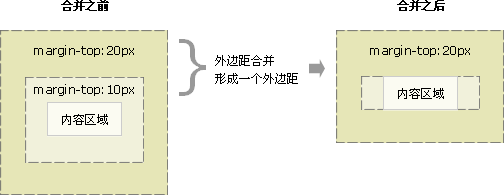
## 合并后的外边距的高度等于两个发生合并的外边距的高度中的较大者。

## 外边距合并。简单地说，外边距合并指的是，当两个垂直外边距相遇时，它们将形成一个外边距。合并后的外边距的高度等于两个发生合并的外边距的高度中的较大者。

当一个元素出现在另一个元素上面时，第一个元素的下外边距与第二个元素的上外边距会发生合并。请看下图：

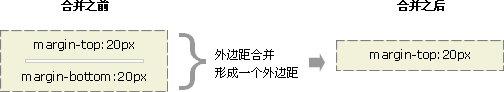


当一个元素包含在另一个元素中时（假设没有内边距或边框把外边距分隔开），它们的上和/或下外边距也会发生合并。请看下图：

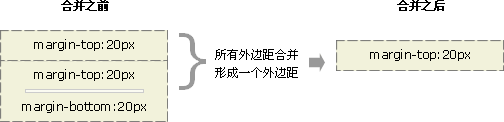


尽管看上去有些奇怪，但是外边距甚至可以与自身发生合并。

假设有一个空元素，它有外边距，但是没有边框或填充。在这种情况下，上外边距与下外边距就碰到了一起，它们会发生合并：

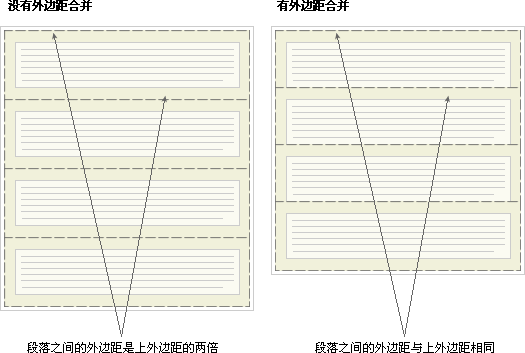


如果这个外边距遇到另一个元素的外边距，它还会发生合并：



这就是一系列的段落元素占用空间非常小的原因，因为它们的所有外边距都合并到一起，形成了一个小的外边距。

外边距合并初看上去可能有点奇怪，但是实际上，它是有意义的。以由几个段落组成的典型文本页面为例。第一个段落上面的空间等于段落的上外边距。如果没有外边距合并，后续所有段落之间的外边距都将是相邻上外边距和下外边距的和。这意味着段落之间的空间是页面顶部的两倍。如果发生外边距合并，段落之间的上外边距和下外边距就合并在一起，这样各处的距离就一致了。



**注释：**只有普通文档流中块框的垂直外边距才会发生外边距合并。行内框、浮动框或绝对定位之间的外边距不会合并。

在css2.1中，水平的margin不会被折叠。

垂直margin可能在一些盒模型中被折叠：

1、在常规文档流中，2个或以上的块级盒模型相邻的垂直margin会被折叠。

　最终的margin值计算方法如下：

 a、全部都为正值，取最大者；

 b、不全是正值，则都取绝对值，然后用正值减去最大值；

 c、没有正值，则都取绝对值，然后用0减去最大值。

注意：相邻的盒模型可能由DOM元素动态产生并没有相邻或继承关系。

 2、相邻的盒模型中，如果其中的一个是浮动的（floated），垂直margin不会被折叠，甚至一个浮动的盒模型和它的子元素之间也是这样。

 3、设置了overflow属性的元素和它的子元素之间的margin不会被折叠（overflow取值为visible除外）。

 4、设置了绝对定位（position:absolute）的盒模型，垂直margin不会被折叠，甚至和他们的子元素之间也是一样。

 5、设置了display:inline-block的元素，垂直margin不会被折叠，甚至和他们的子元素之间也是一样。

 6、如果一个盒模型的上下margin相邻，这时它的margin可能折叠覆盖（collapse through）它。在这种情况下，元素的位置（position）取决于它的相邻元素的margin是否被折叠。

  a、如果元素的margin和它的父元素的margin-top折叠在一起，盒模型border-top的边界定义和它的父元素相同。

  b、另外，任意元素的父元素不参与margin的折叠，或者说只有父元素的margin-bottom是参与计算的。如果元素的border-top非零，那么元素的border-top边界位置和原来一样。

  一个应用了清除操作的元素的margin-top绝不会和它的块级父元素的margin-bottom折叠。

   注意，那些已经被折叠覆盖的元素的位置对其他已经被折叠的元素的位置没有任何影响；只有在对这些元素的子元素定位时，border-top边界位置才是必需的。

7、根元素的垂直margin不会被折叠。

　　浮动的块级元素的margin-bottom总是与它后面的浮动块级兄弟元素（floated next in-flow block-level sibling）的margin-top相邻，除非那个同级元素使用了清除操作。

　　浮动的块级元素的margin-top和它的第一个浮动块级子元素（floated first in-flow block-level child）的margin-top相邻（如果该元素没有border-top，没有padding-top，并且子元素没有使用清除操作）。

　　浮动的块级元素的margin-bottom如果符合下列条件，那么它和它的最后一个浮动块级子元素的margin-bottom相邻（如果该元素没有指定padding-bottom或border）：

  a、指定了height:auto

  b、min-height小于元素的实际使用高度（height）

  c、max-height大于元素的实际使用高度（height）

　　如 果一个元素的min-height属性设置为0，那么它所拥有的margin是相邻的，并且它既没有border-top和border- bottom，也没有padding-top和padding-bottom，它的height属性可以是0或auto，它不能包含一个内联的盒模型 （line box），它所有的浮动子元素（如果有的话）的margin也都是相邻的。

　　当一个元素拥有的margin折叠了，并且它使用了清除操作，那么它的margin-top会和紧随其后的兄弟元素的相邻margin折叠，但结果是它的margin将无法和其块级父元素的margin-bottom折叠。

　　折叠操作是以padding、margin、border的值为基础的（即在浏览器解析所有这些值之后），折叠后的margin计算将覆盖已使用的不同margin的值。

如何解决

　　W3C的CSS2.1 定义了几种情况，简单为元素添加边框或者间距（border or padding)就可以不使边距重叠。

　 我们添加一个内边距:#box{margin:10px;padding:1px;}

1. **zoom：1的清除浮动原理**

这条命令的作用是激活父元素的"hasLayout"属性，让父元素拥有自己的布局，IE使用Layout概念来控制元素的尺寸和位置。如果一个元素有Layout，它就有自身的尺寸和位置；如果没有，它的尺寸和位置由最近的拥有布局的祖先元素控制。

Zoom属性是IE浏览器的专有属性，它可以设置或检索对象的缩放比例。解决ie下比较奇葩的bug。

来龙去脉大概如下：

当设置了zoom的值之后，所设置的元素就会就会扩大或者缩小，高度宽度就会重新计算了，这里一旦改变zoom值时其实也会发生重新渲染，运用这个原理，也就解决了ie下子元素浮动时候父元素不随着自动扩大的问题。

Zoom属是IE浏览器的专有属性，火狐和老版本的webkit核心的浏览器都不支持这个属性。然而，zoom现在已经被逐步标准化，出现在 CSS 3.0 规范草案中。

目前非ie由于不支持这个属性，它们又是通过什么属性来实现元素的缩放呢？

可以通过css3里面的动画属性scale进行缩放。

1. **css性能优化**

CSS 优化主要是四个方面：

加载性能这个方面相关的 best practice 太多了，网上随便找一找就是一堆资料，比如不要用 import 啊，压缩啊等等，主要是从减少文件体积、减少阻塞加载、提高并发方面入手的，任何 hint 都逃不出这几个大方向。

1. 选择器性能

可以参考 GitHub 的这个分享 https://speakerdeck.com/jonrohan/githubs-css-performance，但 selector 的对整体性能的影响可以忽略不计了，selector 的考察更多是规范化和可维护性、健壮性方面，很少有人在实际工作当中会把选择器性能作为重点关注对象的，但也像 GitHub 这个分享里面说的一样——知道总比不知道好。

1. 渲染性能

渲染性能是 CSS 优化最重要的关注对象。页面渲染 junky 过多？看看是不是大量使用了 text-shadow？是不是开了字体抗锯齿？CSS 动画怎么实现的？合理利用 GPU 加速了吗？什么你用了 Flexible Box Model？有没有测试换个 layout 策略对 render performance 的影响？这个方面搜索一下 CSS render performance 或者 CSS animation performance 也会有一堆一堆的资料可供参考。

1. 可维护性、健壮性

命名合理吗？结构层次设计是否足够健壮？对样式进行抽象复用了吗？优雅的 CSS 不仅仅会影响后期的维护成本，也会对加载性能等方面产生影响。这方面可以多找一些 OOCSS（不是说就要用 OOCSS，而是说多了解一下）等等不同 CSS Strategy 的信息，取长补短。

1、避免使用多类选择符

2、移除空的css规则

3、正确使用display属性

4、不滥用浮动

5、不滥用web字体

6、不声明过多的font-size

7、不在选择符中使用id标示符

8、不重复定义h1~h6元素

9、值为0的时候不需要任何单位

10、标准化各种浏览器前缀

11、使用CSS渐变等高级特性，需要指定多有浏览器前缀

12、遵守盒模型规则

**23、浏览器是怎样解析CSS选择器的？**

从上到下，从右到左

1. **margin 、padding适合什么情景下使用？**

何时当用margin:

需要再border外侧添加空白时；

空白处不需要背景时；

上下相连的两个盒子之间的空白，需要相互抵消时。

何时当用padding:

需要在border内侧添加空白时；

空白处需要背景时；

上下相连的两个盒子之间的空白，希望能与两者之和时

1. **::before 和 :after中双冒号和单冒号有什么区别？解释一下这2个伪元素的作用。**

单冒号(:)用于CSS3伪类，双冒号(::)用于CSS3伪元素。

伪类：伪类选择元素基于的是当前元素处于的状态，或者说元素当前所具有的特性，例如： :link :hover :active :focus :first-child :before :after

伪元素：与伪类针对特殊状态的元素不同的是，伪元素是对元素中的特定内容进行操作。::first-letter ::first-line ::before ::after

1. **怎样让chrome浏览器支持小于12px的字体**
2. 针对谷歌浏览器内核，加webkit前缀，用transform:scale()这个属性进行缩放！
3. 设置：-webkit-text-size-adjust:none;即可解决问题。
4. **让页面里的字体变清晰，变细用CSS怎么做？**

-webkit-font-smoothing: antialiased;

1. **font-style属性可以让它赋值为“oblique” oblique是什么意思？**

文本倾斜显示

italic 和 oblique 的区别

font-style 非常简单：用于在 normal 文本、italic 文本和 oblique 文本之间选择。唯一有点复杂的是明确 italic 文本和 oblique 文本之间的差别。

斜体（italic）是一种简单的字体风格，对每个字母的结构有一些小改动，来反映变化的外观。与此不同，倾斜（oblique）文本则是正常竖直文本的一个倾斜版本。

通常情况下，italic 和 oblique 文本在 web 浏览器中看上去完全一样。

1. **display:inline-block 什么时候会显示间隙？**
2. **你能描述一下渐进增强和优雅降级之间的不同吗?**
3. 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。
4. 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。
5. 区别：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，而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。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。

**31、为什么利用多个域名来存储网站资源会更有效？**

1. CDN缓存更方便
2. 突破浏览器并发限制
3. 节约cookie带宽
4. 节约主域名的连接数，优化页面响应速度
5. 防止不必要的安全问题
6. **说一下src与href的区别？**

src用于替换当前元素，href用于在当前文档和引用资源之间确立联系。

1. src是source的缩写，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。

<script src =”js.js”></script>

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。

1. href是Hypertext Reference的缩写，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，如果我们在文档中添加

<link href=”common.css” rel=”stylesheet”/>

那么浏览器会识别该文档为css文件，就会并行下载资源并且不会停止对当前文档的处理。这也是为什么建议使用link方式来加载css，而不是使用@import方式。

**33、图片格式有哪些？**

png-8，png-24，jpeg，gif，svg。

但是上面的那些都不是面试官想要的最后答案。面试官希望听到是Webp。（是否有关注新技术，新鲜事物）

科普一下Webp：WebP格式，谷歌（google）开发的一种旨在加快图片加载速度的图片格式。图片压缩体积大约只有JPEG的2/3，并能节省大量的服务器带宽资源和数据空间。Facebook Ebay等知名网站已经开始测试并使用WebP格式。

在质量相同的情况下，WebP格式图像的体积要比JPEG格式图像小40%

**34、**在css/js代码上线之后开发人员经常会优化性能，从用户刷新网页开始，一次js请求一般情况下有哪些地方会有缓存处理？****

dns缓存，cdn缓存，浏览器缓存，服务器缓存。

1. 浏览器缓存：

1、浏览器缓存的缺点

客户端缓存减少了的服务器请求，避免了文件重复加载，显著地提升了用户地方。但是当网站发生了更新的时候（如替换了css、js以及图片文件），浏览器本地仍保存着旧版本的文件，从而导致无法预料后果。

2、浏览器缓存的策略

Expires:设置过期时间，如果http响应报文中设置了Expires，在Expires过期之前，我们就避免了和服务器之间的连接。此时，浏览器无需想浏览器发出请求，只需要自己判断手中的材料是否过期就可以了，完全不需要增加服务器的负担。

Catch-control:max-age

Expires的方法很好，但是我们每次都得算一个精确的时间。max-age 标签可以让我们更加容易的处理过期时间。是以秒计算

Last-Modified:服务器为了通知浏览器当前文件的版本，会发送一个上次修改时间的标签，这样浏览器就知道他收到的这个文件创建时间，在后续的请求中，浏览器会按照下面的规则进行验证：

1. 浏览器：Hey，我需要jquery.min.js这个文件，如果是在 Last-Modified之后修改过的，if-modified-since:Last-Modified所设置的时间，请发给我。
2. 服务器：（检查文件的修改时间）服务器：Hey，这个文件在那个时间之后没有被修改过，你已经有最新的版本了。返回304状态码

ETag：通常情况下，通过修改时间来比较文件是可行的。但是在一些特殊情况，例如服务器的时钟发生了错误，服务器时钟进行修改，夏时制DST到来后服务器时间没有及时更新，这些都会引起通过修改时间比较文件版本的问题。

ETag可以用来解决这种问题。ETag是一个文件的唯一标志符。就像一个哈希或者指纹，每个文件都有一个单独的标志，只要这个文件发生了改变，这个标志就会发生变化。

1. Ctrl+F5
2. 在浏览器地址栏中敲一个回车
3. 按F5刷新

你可知道这三种不同的操作方式，决定浏览器不同的刷新缓存策略？

浏览器缓存刷新

1. 在地址栏中输入网址后按回车或点击转到按钮

浏览器以最少的请求来获取网页的数据，浏览器会对所有没有过期的内容直接使用本地缓存，从而减少了对浏览器的请求。所以，Expires，max-age标记只对这种方式有效。

1. 按F5或浏览器刷新按钮

浏览器会在请求中附加必要的缓存协商，但不允许浏览器直接使用本地缓存，它能够让 Last-Modified、ETag发挥效果，但是对Expires无效。

1. 按Ctrl+F5或按Ctrl并点击刷新按钮

这种方式就是强制刷新，总会发起一个全新的请求，不使用任何缓存。

1. Cdn（内容分发网络）缓存：

1.CDN加速原理

　　通过动态域名解析，网友的请求被分配到离自己最快的服务器。CDN服务器直接返回缓存文件或通过专线代理原站的内容。

　　网络加速+内容缓存，有效提供访问速度

2.CDN节点数量

　　全国多个机房，每个机房多台服务器，CDN节点一般上百台

3.CDN缓存什么内容

　　缓存html、图片、css、xml等静态资源，不缓存含有？的动态地址、jsp、php，js文件也不缓存【除非特殊设置】

　　缓存原站返回HTTP状态为20\*或304，不缓存其他状态(例如404，500，503)

4.CDN缓存内容的更新

　　a）用户首次请求，CDN从原站抓取后缓存，直到文件过期后有用户请求再次更新

　　b）程序主动通知CDN抓取

5.CDN缓存内容的有效期

　　a）原站apache吐出的静态文件：由apache的expire和header模块控制

　　主要两项：last-modified，cache-control：max-age

　　apache缺省配置，所有静态文件在cdn只缓存3600s【需要我们按需求调整被加速服务器的apache设置】

　　3600s后cdn失效，用户访问时会重新请求原站，如果没有变化，缓存失效周期自动延长10%。

　　b）原站jsp或php吐出的动态内容（url形式必须是静态的）

　　由程序控制last-modified，cache-control：max-age public ，apache的设置将不起作用

　　cdn根据这两项判断是否需要到原站更新内容

6.CDN和应用的结合策略

　　a）变化不频繁的页面：例如图吧的图片显示页、车型页、已结束的比赛对阵页

　　在原站生成静态页面，原站apache上定义过期时间，例如1天。

　　原站上静态文件更新后，可以等待cdn过期。或者主动通知cdn更新（随着cdn节点越来越多，代价会非常高）

　　b）变化频繁的页面：例如足球库中的及时亚盘、及时欧赔、正在进行的比赛对阵页

　　不生成静态页面，由jsp或php定义过期时间，例如5s或60s。cdn过期后，如果有用户访问就从原站上抓取。

　　优点：相关页面内容更新后，不需要主动通知100个原站都来抓取，有效降低原站的压力。

　　如果页面内容没有变化，返回lastmodified不变，这样原站会直接返回304给cdn，cdn也会返回304给用户。减少网络传输和速度

　　比赛结束后，“正在进行的比赛对阵页”转换为第一类情况，再生成静态文件

　　c）特殊静态资源：例如图片库和某些大型产品库中的评论js

　　或者频繁访问、频繁更新的页面：例如足球赛事库的及时比分文件

　　通过apache nocache告诉IE不缓存，html中就不需要使用pinglun.js?123456这样的代码形式

然后用max-age告诉cdn缓存1s，这样避免每次用户请求都转到原站

1. **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。**

* 图片懒加载，在页面上的未可视区域可以添加一个滚动条事件，判断图片位置与浏览器顶端的距离与页面的距离，如果前者小于后者，优先加载。
* 如果为幻灯片、相册等，可以使用图片预加载技术，将当前展示图片的前一张和后一张优先下载。
* 如果图片为css图片，可以使用CSSsprite，SVGsprite，Iconfont、Base64等技术。
* 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。
* 如果图片展示区域小于图片的真实大小，则因在服务器端根据业务需要先行进行图片压缩，图片压缩后大小与展示一致**。**

**36、超链接访问过后hover样式就不出现的问题是什么？如何解决？**

被点击访问过的超链接样式不在具有hover和active了,解决方法是改变CSS属性的排列顺序: L-V-H-A（link,visited,hover,active）

1. **谈谈以前端角度出发做好SEO需要考虑什么？**

* 了解搜索引擎如何抓取网页和如何索引网页

你需要知道一些搜索引擎的基本工作原理，各个搜索引擎之间的区别，搜索机器人（SE robot 或叫 web crawler）如何进行工作，搜索引擎如何对搜索结果进行排序等等。

* Meta标签优化

主要包括主题（Title)，网站描述(Description)，和关键词（Keywords）。还有一些其它的隐藏文字比如Author（作者），Category（目录），Language（编码语种）等。

* 如何选取关键词并在网页中放置关键词

搜索就得用关键词。关键词分析和选择是SEO最重要的工作之一。首先要给网站确定主关键词（一般在5个上下），然后针对这些关键词进行优化，包括关键词密度（Density），相关度（Relavancy），突出性（Prominency）等等。

* 了解主要的搜索引擎

虽然搜索引擎有很多，但是对网站流量起决定作用的就那么几个。比如英文的主要有Google，Yahoo，Bing等；中文的有百度，搜狗，有道等。不同的搜索引擎对页面的抓取和索引、排序的规则都不一样。还要了解各搜索门户和搜索引擎之间的关系，比如AOL网页搜索用的是Google的搜索技术，MSN用的是Bing的技术。

* 主要的互联网目录

Open Directory自身不是搜索引擎，而是一个大型的网站目录，他和搜索引擎的主要区别是网站内容的收集方式不同。目录是人工编辑的，主要收录网站主页；搜索引擎是自动收集的，除了主页外还抓取大量的内容页面。

* 按点击付费的搜索引擎

搜索引擎也需要生存，随着互联网商务的越来越成熟，收费的搜索引擎也开始大行其道。最典型的有Overture和百度，当然也包括Google的广告项目Google Adwords。越来越多的人通过搜索引擎的点击广告来定位商业网站，这里面也大有优化和排名的学问，你得学会用最少的广告投入获得最多的点击。

* 搜索引擎登录

网站做完了以后，别躺在那里等着客人从天而降。要让别人找到你，最简单的办法就是将网站提交（submit）到搜索引擎。如果你的是商业网站，主要的搜索引擎和目录都会要求你付费来获得收录（比如Yahoo要299美元），但是好消息是（至少到目前为止）最大的搜索引擎Google目前还是免费，而且它主宰着60％以上的搜索市场。

* 链接交换和链接广泛度（Link Popularity）

网页内容都是以超文本（Hypertext）的方式来互相链接的，网站之间也是如此。除了搜索引擎以外，人们也每天通过不同网站之间的链接来Surfing（“冲浪”）。其它网站到你的网站的链接越多，你也就会获得更多的访问量。更重要的是，你的网站的外部链接数越多，会被搜索引擎认为它的重要性越大，从而给你更高的排名。

1. **什么是cookie隔离？(或者说请求文件的时候不要带上cookie)**

如果静态文件都放在主域名下，那静态文件请求的时候都带有的cookie的数据提交给server的，非常浪费流量，所以不如隔离开。

因为cookie有域的限制，因此不能跨域提交请求，故使用非主要域名的时候，请求头中就不会带有cookie数据，这样可以降低请求头的大小，降低请求时间，从而达到降低整体请求延时的目的。

同时这种方式不会将cookie传入Web Server，也减少了Web Server对cookie的处理分析环节，提高了webserver的http请求的解析速度。

## 如何修改chrome记住密码后自动填充表单的黄色背景 ？

input:-webkit-autofill,textarea:-webkit-autofill,select:-webkit-autofill {background-color: rgb(250, 255, 189); /\* #FAFFBD; \*/

background-image: none;

color: rgb(0, 0, 0);}

**34、什么是Css Hack？ie6,7,8的hack分别是什么？**

答案：针对不同的浏览器写不同的CSS code的过程，就是CSS hack。

#test{

     background-color:yellow;    /\*ie8\*/

     +background-color:pink;        /\*ie7\*/

     \_background-color:orange;       /\*ie6\*/    }

更好的方式是使用IE条件判断语句：

<!–[if lte IE 6]><![endif]–>

## 35、rgba()和opacity的透明效果有什么不同？

rgba()和opacity都能实现透明效果，但最大的不同是opacity作用于元素，以及元素内的所有内容的透明度，而rgba()只作用于元素的颜色或其背景色。（设置rgba透明的元素的子元素不会继承透明效果！）

## css中可以让文字在垂直和水平方向上重叠的两个属性是什么？

垂直方向：line-height

水平方向：letter-spacing

那么问题来了，关于letter-spacing的妙用知道有哪些么？

答案:可以用于消除inline-block元素间的换行符空格间隙问题。

## 如何垂直居中一个<img>?

## /\*<img>的容器设置如下\*/

## #container{

## display:table-cell;

## text-align:center;

## vertical-align:middle;}

## 38、BFC是什么？

BFC就是“块级格式化上下文”的意思，创建了 BFC的元素就是一个独立的盒子，不过只有Block-level box可以参与创建BFC， 它规定了内部的Block-level Box如何布局，并且与这个独立盒子里的布局不受外部影响，当然它也不会影响到外面的元素。

* BFC有一下特性：

内部的Box会在垂直方向，从顶部开始一个接一个地放置。

Box垂直方向的距离由margin决定。属于同一个BFC的两个相邻Box的margin会发生叠加。

每个元素的margin box的左边， 与包含块border box的左边相接触(对于从左往右的格式化，否则相反)。即使存在浮动也是如此。

BFC的区域不会与float box叠加。

BFC就是页面上的一个隔离的独立容器，容器里面的子元素不会影响到外面的元素，反之亦然。

计算BFC的高度时，浮动元素也参与计算。

* 如何触发BFC？

float 除了none以外的值

overflow 除了visible 以外的值（hidden，auto，scroll ）

display (table-cell，table-caption，inline-block, flex, inline-flex)

position值为（absolute，fixed）

fieldset元素

## 39.IE的双边距BUG：块级元素float后设置横向margin，ie6显示的margin比设置的较大。

解决：加入\_display：inline

## 40.HTML与XHTML——二者有什么区别？

**XHTML：**

1. 所有的标记都必须要有一个相应的结束标记

2. 所有标签的元素和属性的名字都必须使用小写

3. 所有的 XML 标记都必须合理嵌套

4. 所有的属性必须用引号 "" 括起来

5. 把所有 < 和 & 特殊符号用编码表示

6. 给所有属性赋一个值

7. 不要在注释内容中使用 "--"

8. 图片必须有说明文字

## **41、visibility:collapse;是干嘛用的?不同浏览器有什么区别？**

* 当一个元素的visibility属性被设置成collapse值后，对于一般的元素，它的表现跟hidden是一样的。
* 但例外的是，如果这个元素是table相关的元素，例如table行，table group，table列，table column group，它的表现却跟display: none一样，也就是说，它们占用的空间也会释放。
* 在谷歌浏览器里，使用collapse值和使用hidden值没有什么区别;
* 在火狐浏览器、Opera和IE11里，使用collapse值的效果就相当于display:none;

**42、position跟display、margin collapse、overflow、float这些特性相互叠加后会怎么样？**

* 若display:none,则position和float都不起作用；
* display不是none,当position:absolute或fixed的时候，float的计算值都为none,且display的计算方法为

inline-table ——> table

inline,run-in,table-row-group,table-column,table-column-group,table-eader-group,table-footer-group,table-row,table-cell,table-caption,inline-block ——> block

其他 ——> 同设定值

* display不是none,position不是absolute或fixed,当float不是none，即有设定值，则display按照上述计算方法计算；
* 当float是none，即没有设定值：

若元素时根元素，display按照上述计算方法计算，如不是，则应用设定值

**43、HTML5的离线存储怎样使用？**

* 在用户没有与因特网连接时，可以正常访问站点或应用，在用户与因特网连接时，更新用户机器上的缓存文件。
* 原理：HTML5的离线存储是基于一个新建的.appcache文件的缓存机制(不是存储技术)，通过这个文件上的解析清单离线存储资源，这些资源就会像cookie一样被存储了下来。之后当网络在处于离线状态下时，浏览器会通过被离线存储的数据进行页面展示。
* 如何使用：

1、页面头部像下面一样加入一个manifest的属性；

2、在cache.manifest文件的编写离线存储的资源；

CACHE MANIFEST

#v0.11

CACHE:

js/app.js

css/style.css

NETWORK:

resourse/logo.png

FALLBACK:

/ /offline.html

1. 在离线状态时，操作window.applicationCache进行需求实现。

**44、、浏览器是怎么对HTML5的离线储存资源进行管理和加载的呢？**

在线的情况下，浏览器发现html头部有manifest属性，它会请求manifest文件，如果是第一次访问app，那么浏览器就会根据manifest文件的内容下载相应的资源并且进行离线存储。如果已经访问过app并且资源已经离线存储了，那么浏览器就会使用离线的资源加载页面，然后浏览器会对比新的manifest文件与旧的manifest文件，如果文件没有发生改变，就不做任何操作，如果文件改变了，那么就会重新下载文件中的资源并进行离线存储。

离线的情况下，浏览器就直接使用离线存储的资源。

**45、iframe有哪些缺点？**

* iframe会阻塞页面的onload事件
* 搜索引擎无法解析这样的页面，不利于SEO
* iframe和主页面共享连接池，而浏览器对相同域的连接有限制，所以会影响页面的并行加载。
* 解决方案：如果要使用iframe最好使用js引入,动态添加src属性

**46、WebSocket**

**47、用纯CSS创建一个三角形的原理是什么？**

把上、左、右三条边隐藏掉（颜色设为 transparent）

#demo {

width: 0;

height: 0;

border-width: 20px;

border-style: solid;

border-color: transparent transparent red transparent;

}

**48、position:fixed;在android下无效怎么处理？**

fixed的元素是相对整个页面固定位置的，你在屏幕上滑动只是在移动这个所谓的viewport，原来的网页还好好的在那，fixed的内容也没有变过位置，

所以说并不是iOS不支持fixed，只是fixed的元素不是相对手机屏幕固定的。

<metaname="viewport"content="width=device-width,initial-scale=1.0,maximum-scale=1.0,minimum-scale=1.0, user-scalable=no"/>