**CSS面试总结**

说明：面试或笔试（具体是面试公司情况而定）。

特别说明：本篇总结为个人心得总结，仅完美适用于作者本人。如有异议，请自行消化或删除文档。

作者：陶福荣

备注：格式如果错乱，请使用正版offic2013

版权所有，非作者本人同意，不得上传于网络

# 1、css sprite是什么,有什么优缺点

## 1）概念

将多个小图片拼接到一个图片中【制作精灵图的工具】。通过background-position和元素尺寸调节需要显示的背景图案。

## 2）优点

1.减少HTTP请求数，极大地提高页面加载速度

2.增加图片信息重复度，提高压缩比，减少图片大小

3.更换风格方便，只需在一张或几张图片上修改颜色或样式即可实现

## 3）缺点

1.图片合并麻烦

2.维护麻烦，修改一个图片可能需要从新布局整个图片，样式

# 2、隐藏

1）visibiltily:hidden 完全隐藏，占位，不可见，鼠标不可操作 visible默认显示

2）display：none 被隐藏，不占位，不可见，鼠标不可操作

3）opcity：0-1 隐藏，占位，不可见，鼠标可操作

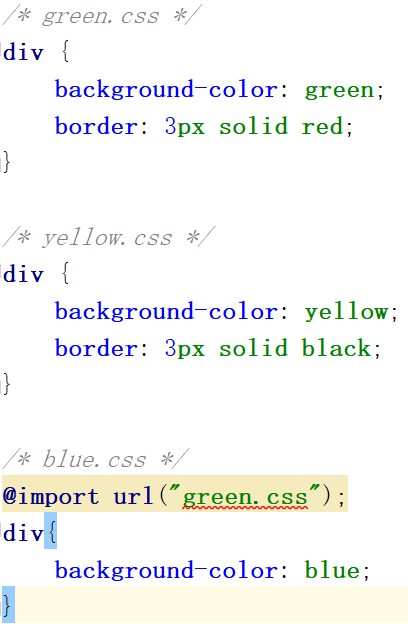
4）height：0；未隐藏，不占位，不可见，鼠标不可操作

5）overflow： didden隐藏溢出，visible默认显示，scroll滚动条，auto超出有滚动条，不超出没有滚动条。

# [3、link和@import](mailto:3.link和@import)的区别

## 1）从属关系区别

@import是 CSS 提供的语法规则，只有导入样式表的作用；link是HTML提供的标签，不仅可以加载 CSS 文件，还可以定义 RSS、rel 连接属性等。



## 2）加载顺序区别

加载页面时，link标签引入的 CSS 被同时加载；@import引入的 CSS 将在页面加载完毕后被加载。

## 3）兼容性区别

@import是 CSS2.1 才有的语法，故只可在 IE5+ 才能识别；link标签作为 HTML 元素，不存在兼容性问题。

## 4）DOM可控性区别

可以通过 JS 操作 DOM ，插入link标签来改变样式；由于DOM方法是基于文档的，无法使用@import的方式插入样式。

## 5）权重

两者实质没有权重概念，谁在后面就使用谁的样式。

# 4、什么是 FOUC（无样式内容闪烁）？你如何来避免 FOUC？

## 1）概念

如果使用import方法对CSS进行导入,会导致某些页面在Windows 下的Internet Explorer出现一些奇怪的现象:以无样式显示页面内容的瞬间闪烁,这种现象称之为文档样式短暂失效(Flash of Unstyled Content),简称为FOUC。

## 2）原因

 1．使用import方法导入样式表

2．将样式表放在页面底部

3．有几个样式表，放在html结构的不同位置。

其实原理很清楚：当样式表晚于结构性html加载，当加载到此样式表时，页面将停止之前的渲染。此样式表被下载和解析后，将重新渲染页面，也就出现了短暂的花屏现象。

## 3）解决办法

使用link标签将样式表放在文档head中；

# 5、BFC（块级格式化上下文）

BFC(Block formatting context)直译为"块级格式化上下文"。它是一个独立的渲染区域，只有Block-level box参与， 它规定了内部的Block-level Box如何布局，并且与这个区域外部毫不相干。

## box:css布局的基本单位

Box 是 CSS 布局的对象和基本单位， 直观点来说，就是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（一个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。让我们看看有哪些盒子：

block-level box:display 属性为 block, list-item, table 的元素，会生成 block-level box。并且参与 block fomatting context；

inline-level box:display 属性为 inline, inline-block, inline-table 的元素，会生成 inline-level box。并且参与 inline formatting context；

## 2）Formatting context

　　Formatting context 是 W3C CSS2.1 规范中的一个概念。它是页面中的一块渲染区域，并且有一套渲染规则，它决定了其子元素将如何定位，以及和其他元素的关系和相互作用。最常见的 Formatting context 有 Block fomatting context (简称BFC)和 Inline formatting context (简称IFC)。

## 3）BFC布局规则

1.内部的Box会在垂直方向，一个接一个地放置；

2.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会发生重叠；

3.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；

4.BFC的区域不会与float box重叠；

5.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。反之也如此；

6.计算BFC的高度时，浮动元素也参与计算；

## 4）哪些元素会生成BFC

1.根元素【html】

2.float属性不为none

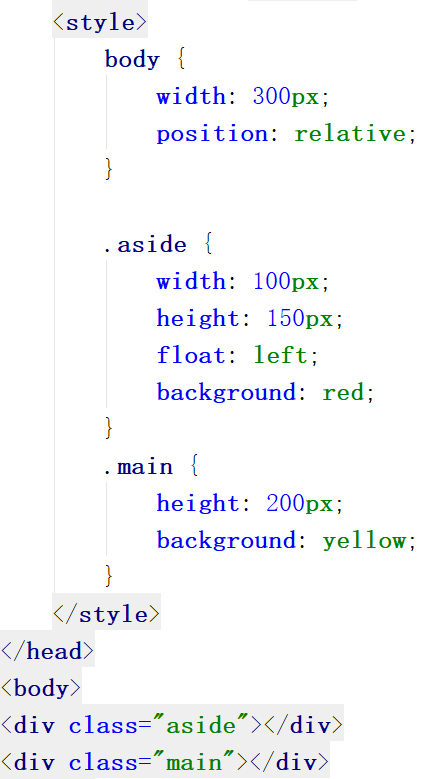
3.position为absolute或fixed

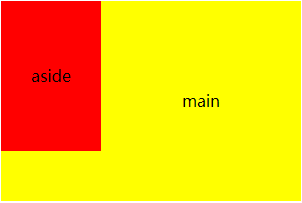
4.display为inline-block, table-cell, table-caption, flex, inline-flex

5.overflow不为visible

## 5）BFC的作用及原理-自适应两栏布局

下面为BFC的演示示例代码:定义了一个aside和一个main。其中aside开启了浮动，脱离了文档流，浮在了main的上面。





效果图

（1）根据BFC布局规则第三条

每个元素的margin box的左边，与包含块【body】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；

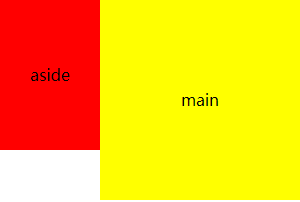
因此，虽然存在浮动的元素aside，但main的左边依然会与包含块【body】的左边相接触。

（2）根据BFC布局规则第四条

BFC的区域不会与float box重叠。

我们可以开启main的BFC，来使其和aside不重合。

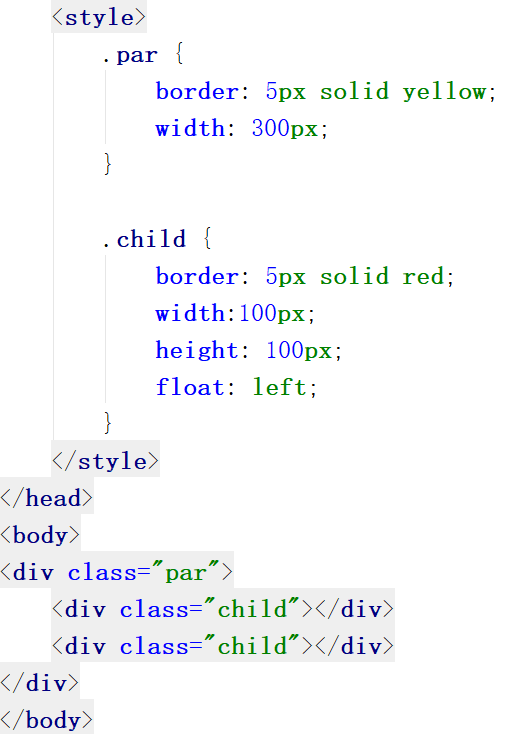
给main加一个 overflow:hidden;开始BFC

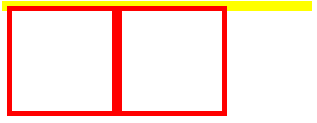


效果图

## 6）BFC的作用及原理-清除内部浮动

以下代码定义了一个包含块和两个子元素，这里我们主要解决高度塌陷的问题；





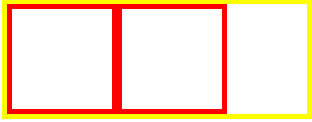
效果图

从效果图中我们不难看出，由于设置了子元素浮动，导致子元素脱离了文档流，使父元素产生了高度塌陷；

根据BFC布局规则第六条：

计算BFC的高度时，浮动元素也参与计算；

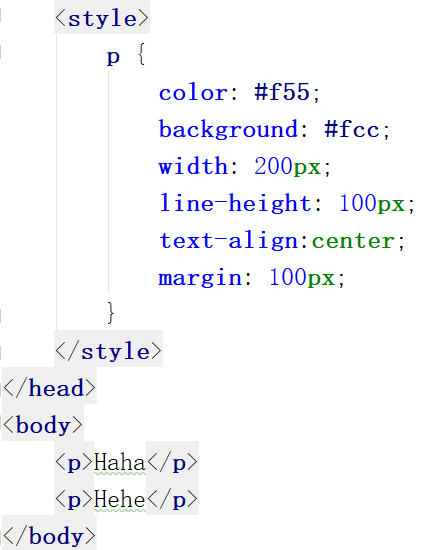
故给父元素par开启BFC，overflow:hidden；

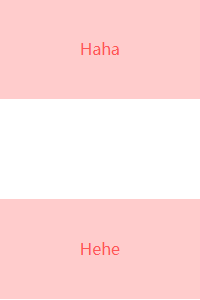


效果图

## 7）BFC的作用及原理-防止外边距合并

下面为一段示例代码，定义了两个p，他们的外边距都为100，理论上加起来是200，但是由于外边距合并问题，他们之间的实际距离为100；





效果图

根据BFC布局规则第二条：

Box垂直方向的距离由margin决定。属于同一个BFC 的两个相邻的margin会发生重叠；

我们可以给第二个p标签外面包裹一个div，然后开启BFC；overflow：hidden；



效果图

**总结：**以上几个例子都体现出了BFC布局规则的第五条，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；

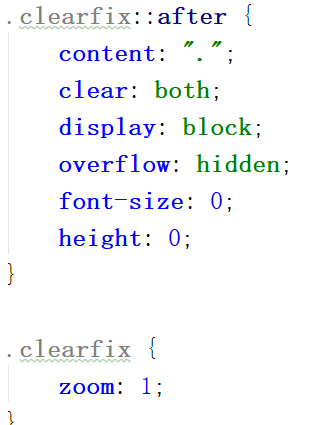
因为BFC内部的元素和外部的元素绝对不会互相影响，因此， 当BFC外部存在浮动时，它不应该影响BFC内部Box的布局，BFC会通过变窄，而不与浮动有重叠。同样的，当BFC内部有浮动时，为了不影响外部元素的布局，BFC计算高度时会包括浮动的高度。避免margin重叠也是这样的一个道理。

# 6. 清除浮动的几种方式，各自的优缺点

1）使用空标签清除浮动clear:both（缺点，增加无意义的标签）

2）使用overflow:auto（使用zoom:1用于兼容IE，缺点：内部宽高超过父级div时，会出现滚动条）

3）用afert伪元素清除浮动(IE8以上和非IE浏览器才支持，目前：大型网站都有使用



# 7、为什么要初始化css样式

CSS初始化是指重设浏览器的样式。不同的浏览器默认的样式可能不尽相同，所以开发时的第一件事可能就是如何把它们统一。如果没对CSS初始化往往会出现浏览器之间的页面差异。每次新开发网站或新网页时候通过初始化CSS样式的属性，为我们将用到的CSS或html标签更加方便准确，使得我们开发网页内容时更加方便简洁，同时减少CSS代码量，节约网页下载时间。

## CSS初始化示例代码



上面是一个css文件，保存了基本常见的css初始化代码；

# 8、css3有哪些新特性

1）新增各种CSS选择器 （: not(.input)：所有 class 不是“input”的节点）

2）圆角 （border-radius:8px）

3）多列布局 （multi-column layout）

4）阴影和反射 （Shadow\Reflect）

5）文字特效 （text-shadow、）

6）文字渲染 （Text-decoration）

7）线性渐变 （gradient）

8）旋转 （transform）

9）增加了旋转,缩放,定位,倾斜,动画，多背景

10）transform:\scale(0.85,0.90)\ translate(0px,-30px)\ skew(-9deg,0deg)\Animation

主要就是二阶段讲的那些特效动画什么的，挑着随便说一说，这里就随便列了几个。最好就是举个做过的例子【延长面试时间，不给面试官说话的机会】；

# 9. ****CSS3新增伪类有那些？****

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

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

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

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

5）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。

6）:after 在元素之前添加内容,也可以用来做清除浮动。

7）:before 在元素之后添加内容

8）:enabled 表单处于激活状态

9）:disabled 控制表单控件的禁用状态。

10）:checked 单选框或复选框被选中

# 10、display有哪些值？说明他们的作用

1）block 象块类型元素一样显示。

2）none 缺省值。象行内元素类型一样显示。

3）inline-block 象行内元素一样显示，但其内容象块类型元素一样显示。

4）list-item 象块类型元素一样显示，并添加样式列表标记。

5）table 此元素会作为块级表格来显示

6）inherit 规定应该从父元素继承 display 属性的值

# 11.介绍一下标准的CSS的盒子模型？低版本IE的盒子模型有什么不同的？

1）有两种， IE 盒子模型、W3C 盒子模型；

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

3）区 别： IE的content部分把 border 和 padding计算了进去;【怪异盒子模型】

普通盒子模型要转换为怪异盒子模型，在style样式里面加 box-sizing:boder-box;属性即可；

# 12、display:inline-block怎么去掉空格

在父级设置font-size:0; 去除换行符间隙，在IE6/7下残留1像素间隙，Chrome浏览器无效，其他浏览器都完美去除；

letter-spacing/word-spacing负值可以去除所有浏览器的换行符间隙，但是，Opera浏览器下极限是间隙1像素，0像素会反弹，换行符间隙还原。

# 13、 CSS选择符有哪些？哪些属性可以继承？

## 1）全局选择器

权重0

\*{

}

## 2）标签选择器

权重1

div{

}

## 3）类选择器/伪类选择器

权重10

.class{

}

## 4）ID选择器

权重100

#id{

}

## 5）行内选择器

权重1000

<p style=”font-size:20px;”></p>

PS:同权重下，则显示底下的样式。

## 6）超链接伪类选择器

1）a:link默认超链接效果

2）a:visited 点击后效果

3）a:hover 鼠标悬停效果

4）a:active 被激活后的【按住鼠标左键不放】效果

## 7）群组选择器

div,p,#id{

}

多个设置样式的选择器通过逗号隔开，使用同一样式

## 8）后代选择器

div div {

}

父级一级级下来，直至可确定对应的标签即可

## 9）子代选择器

div>div {

}

1)子代选择器与后代选择器的区别：后代选择器包含孙子代等，子代选择器只包含子代。

2)div.div1{ } 表示所有含有class选择器【.div1】的div标签。

3)div .div1{ } 表示div下的class选择器【.div1】。

## 10）相邻选择器

相邻选择器用“+”来连接。

Ex:

<div>

<div id=”div1”></div>

<div id=”div2”></div>

</div>

<hr/>

div>div+div{

}

上述代码就表示设置ID为div2的div标签的CSS样式。

## 11）兄弟选择器

兄弟选择器用“~”来连接。

<div>

<div id=”div1”></div>

<div id=”div2”></div>

<span></span>

</div>

<hr/>

div>div~span{

}

上述代码就表示span标签的CSS样式。

# 14.、css新增伪类

## 12）超链接锚点伪类选择器

a:target{

Color:red;

}

超链接的目标样式，当超链接被点击时，锚点目标样式变为目标样式

点击另一个时，目标样式切换。上次的恢复为原来的样式。

## 13）元素伪类选择器

1.:enable

匹配每个已启用的元素。

2.:disable

匹配每个已禁用的元素。

3.:checked

匹配每个已选中的元素。

## 14）结构伪类选择器

1. div: first-child { } div列表中第一个div 子元素

2.div:last-child { } div列表中最后一个div子元素

3.div:only-child{ } div列表中，div底下只有一个子元素

4.div:empty { } div底下为空，没有子元素

5.div:nth-child(3) div列表中第三个div子元素,第三个不能为其他元素，否则样式设置不成功

6.div：nth-of-type(3) div列表中第三个div子元素，其他元素插在中间并不影响

## 15）否定伪类选择器

<div>

<p></p>

<span></span>

<div></div>

</div>

div:not(span){

}

除括号中span标签外所有标签的样式。

## 16）伪元素

在CSS3中将部分伪类选择器更改为伪元素。:hover伪类选择器，::hover伪元素。

1. ::first-letter 第一个汉字或字母

2. ::first-line 第一行

3. selection 选中更改样式

## 17）内容生成器

1. ::before 在之前加元素

2. ::after 在之后加元素

hover::before 鼠标悬停显示。加的元素不能被选中。

a::before{

content:url();

}

可继承的样式： font-size font-family color, UL LI DL DD DT;

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

# 15、请解释一下CSS3的Flexbox（弹性盒布局模型）,以及适用场景？

详细的请参考第二阶段讲的。

# 16. 用纯CSS创建一个三角形的原理是什么？

假设以左边为底边，那么对应的右边就不做设置，然后左、上、下均设置一样的高度，且上下设置颜色为透明。其他边原理和这个一样。

# 17.浏览器兼容问题

## 1）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。

## 2）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:常见症状是ie6中后面的一块被顶到下一行

碰到频率：90%（稍微复杂点的页面都会碰到，float布局最常见的浏览器兼容问题）

解决方案：在float的标签样式控制中加入display:inline;将其转化为行内属性

备注：我们最常用的就是div+css布局了，而div就是一个典型的块属性标签，横向布局的时候我们通常都是用div float实现的，横向的间距设置如果用margin实现，这就是一个必然会碰到的兼容性问题。

## 3）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。

## 4）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（类似第二种）

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。

## 5）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，加上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

## 6）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，有很多情况下我们有这种需求。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。

## 7）透明度的兼容css设置

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 \* （包括上面问题6中的 !important也算是hack的一种。不过实用性较小。）

**详解：**

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。

# 18. 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？

行框的排列会受到中间空白（回车\空格）等的影响，因为空格也属于字符,这些空白也会被应用样式，占据空间，所以会有间隔，把字符大小设为0，就没有空格了。

# 19. 你用过媒体查询，或针对移动端的布局/CSS 吗？

通过媒体查询可以为不同大小和尺寸的媒体定义不同的css，适合相应的设备显示；即响应式布局

/\* 媒体查询 字号 分为20份 \*/

@media screen and (width:320px){

html{

font-size:320/20px;

}

}

@media screen and (width:360px){

html{

font-size:360/20px;

}

}

@media screen and (width:375px){

html{

font-size:375/20px;

}

}

@media screen and (width:414px){

html{

font-size:414/20px;

}

}

@media screen and (width:412px){

html{

font-size:412/20px;

# 20. 浏览器解析加载资源与渲染顺序

## 1）浏览器解析加载资源与渲染顺序

浏览器下载的顺序是从上到下，渲染的顺序也是从上到下，下载和渲染是同时进行的。

在渲染到页面的某一部分时，其上面的所有部分都已经下载完成（并不是说所有相关联的元素都已经下载完，比如图片）。

如果遇到语义解释性的标签嵌入文件（JS脚本，CSS样式），那么此时IE的下载过程会启用单独连接进行下载。

并且在下载后进行解析，解析过程中，停止页面所有往下元素的下载。此时渲染会被阻塞，必须等js、或css资源文件加载并解析完成之后才会继续后面的渲染

样式表在下载完成后，将和以前下载的所有样式表一起进行解析，解析完成后，将对此前所有元素（含以前已经渲染的）重新进行渲染。重新渲染师很耗费性能的，如果可以，尽量把所有的css样式文件都在head里面夹在，实在不行，把首屏的css样式内嵌在页面中，加快首屏显示速度，提升用户体验

## 2）浏览器对css选择器的解析规则

浏览器对选择器的解析规则是从右到左解析的，比如.box .left p,会在页面中找到所有的p标签，然后在p标签中找其父元素有.left类的p元素，再找祖父元素有.box的p标签。  
为了提升渲染速度，可以注意以下几点：  
 （1）缩小查找范围。也就是标签选择器范围太广，可以直接使用类选择器，比如.box .left .text 替代 .box .left p，当然ID选择器更快，但是这不符合服用原则

（2）减少层级关系。 层级嵌套太深不美观也增加查找成本

在 CSS 渲染效率中 id 和 class 的效率是基本相当的 class 会在第一次载入中被缓存，在层叠中会有更加好的效果，在根部元素采用id会具有更加好（id有微妙的速度优势）。

# 21. 什么是响应式设计？为什么要做响应式设计？响应式设计的基本原理是什么？

页面的设计和开发应当根据用户行为以及设备环境（系统平台、屏幕尺寸、屏幕定向等）进行相应的响应和调整。具体的实践方式由多方面组成，包括弹性网格和布局、图片、css media query的使用等。无论用户正在使用笔记本还是iPad，我们的页面都应该能够自动切换分辨率、图片尺寸及相关脚本功能等，以适应不同设备；换句话说，页面应该有能力去自动响应用户的设备环境。

响应式网页设计就是一个网站能够兼容多个终端——而不是为每个终端做一个特定的版本。这样，我们就可以不必为不断到来的新设备做专门的版本设计和开发了。

响应式设计的基本原理是通过媒体查询检测不同的设备屏幕尺寸做处理。页面头部必须有meta声明viewport：

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

# 22. 如果需要手动写动画，你认为最小时间间隔是多久，为什么？

多数显示器默认频率是60Hz，即1秒刷新60次，所以理论上最小间隔为1/60＊1000ms ＝ 16.7ms。

# 23. 什么是Cookie 隔离？（或者说：请求资源的时候不要让它带cookie怎么做）

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。

# 23. style标签写在body后与body前有什么区别？