目录

[常识 2](#_Toc2008179)

[网页与jsx区别 2](#_Toc2008180)

[给div设置border 3](#_Toc2008181)

[两个标签连在一起 3](#_Toc2008182)

[宽高 3](#_Toc2008183)

[特殊属性 3](#_Toc2008184)

[zoom 3](#_Toc2008185)

[clear 4](#_Toc2008186)

[position 4](#_Toc2008187)

[\* 5](#_Toc2008188)

[\_ 5](#_Toc2008189)

[特殊效果 5](#_Toc2008190)

[隐藏 5](#_Toc2008191)

[border三角形 5](#_Toc2008192)

[原理 5](#_Toc2008193)

[实现 6](#_Toc2008194)

[:hover技巧 6](#_Toc2008195)

[行内元素和块级元素 7](#_Toc2008196)

[清除浮动三种方法 7](#_Toc2008197)

[过去 7](#_Toc2008198)

[最新 8](#_Toc2008199)

[注意 9](#_Toc2008200)

[1.父级div定义 伪类:after 和 zoom 9](#_Toc2008201)

[2.父级div定义 overflow:hidden 9](#_Toc2008202)

[3.父级div定义 overflow:auto 10](#_Toc2008203)

[4.父级div定义 height 11](#_Toc2008204)

[5.结尾处加空div标签 clear:both 11](#_Toc2008205)

[css3动画 12](#_Toc2008206)

[原理 12](#_Toc2008207)

[transiton 13](#_Toc2008208)

[使用 13](#_Toc2008209)

[注意 13](#_Toc2008210)

[没有效果的情况 13](#_Toc2008211)

[animation 13](#_Toc2008212)

[使用 13](#_Toc2008213)

[属性 14](#_Toc2008214)

[border 14](#_Toc2008215)

[兼容性 14](#_Toc2008216)

[display 14](#_Toc2008217)

[布局 14](#_Toc2008218)

[间隙 14](#_Toc2008219)

[letter-spacing 14](#_Toc2008220)

[运算 15](#_Toc2008221)

[运算符 15](#_Toc2008222)

[calc 15](#_Toc2008223)

[媒体查询 15](#_Toc2008224)

[逻辑操作符and,not,only,逗号 15](#_Toc2008225)

[逗号 15](#_Toc2008226)

[not 15](#_Toc2008227)

[noly 16](#_Toc2008228)

[规则 16](#_Toc2008229)

[使用场景 16](#_Toc2008230)

[媒体特征 17](#_Toc2008231)

[orientation 17](#_Toc2008232)

[媒体类型 18](#_Toc2008233)

[bfc 18](#_Toc2008234)

[常见问题 18](#_Toc2008235)

[高度变成0 18](#_Toc2008236)

[绝对相对定位 19](#_Toc2008237)

[绝对定位 19](#_Toc2008238)

[高度设置 19](#_Toc2008239)

[移动端 19](#_Toc2008240)

[界面适配 19](#_Toc2008241)

[rem 19](#_Toc2008242)

# 常识

## 网页与jsx区别

<style>

.nav {

width:100%;

background-color:#cccccc;

}

</style>

不是驼峰

没有双引号

## 给div设置border

一定要border的宽度，样式，颜色都要设置才可以的！

.footer {

width:100%;

height:100px;

border:1px solid #c9e2cc;

}

**是空格不是，**

## 两个标签连在一起

.c1.c2 表示如果一个标签中**同时使用了这两个类（不论顺序、不论是否还有其他类），则响应此样式规则。**  
然而在 ie 中，第二个段落中除了有斜体还有下划线，说明它还把 .c1.c2 看作是 .c2。  
于是，考虑到兼容性的原因，尽量不要使用 .c1.c2 这样的选择器。

## 宽高

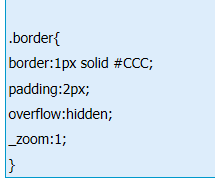
实际宽高包括padding width:100px;padding:20px 实际宽度140px

不包括margin，算入外层。

# 特殊属性

## zoom

Zoom属性是IE浏览器的专有属性，Firefox等浏览器不支持。它可以设**置或检索对象的缩放比例**。除此之外，它还有其他一些小作用，比如触发ie的hasLayout属性，清除浮动、清除margin的重叠等。   
  
**Zoom的使用方法：**zoom :﻿ normal | number   
  
normal :　 默认值。使用对象的实际尺寸   
  
number :　 百分数 | 无符号浮点实数。浮点实数值为1.0或百分数为100%时相当于此属性的 normal 值用白话讲解就是zoom:后面的数字即放大的倍数，可以是数值，也可以是百分比。如：zoom:1，zoom:120%。而这个属性只要在IE中才起作用，所以很少用到它的实际用途，而最经常用到作用是清除浮动等，如：



\_zoom是CSS hack中专对IE6起作用的部分。IE6浏览器会执行zoom:1表示对象的缩放比例，但这里   
overflow:hidden;和\_zoom:1;是连起来用的，作用是清除border内部浮动。   
  
同理，还可以使用同样方法清除margin属性在IE浏览器中的重叠问题。

## clear

清除浮动，如果清楚两侧，任意一侧有浮动，则这一个元素本身下移，这样就避免了不同position元素相互覆盖。（绝对定位也是一种浮动？）

<div class="fn-clear"></div>

.fn-clear:after {

    content: " ";

    clear: both;

    display: block;

    font-size: 0;

    visibility: hidden;

    height: 0;

}

.fn-clear {

    zoom: 1;

}

## position

如果在相对布局中有一个绝对布局元素，那个元素最好是相对顶部而不是底部，否则当页面高度大于窗口时会错误。

能不用absolute尽量不用，使用float或者relative+top，left

## \*

和!importan一样用来设置优先级（用在属性外表示所有 \*{}）

IE都能识别\*;标准浏览器(如火狐)不能识别\*；

IE6能识别\*，但不能识别 !important;  
 IE7能识别\*，也能识别!important;   
 FF不能识别\*，但能识别!important;

所以最好使用!important

## \_

也是设置优先级

下划线"\_", IE6支持下划线，IE7和firefox均不支持下划线。

# 特殊效果

## 隐藏





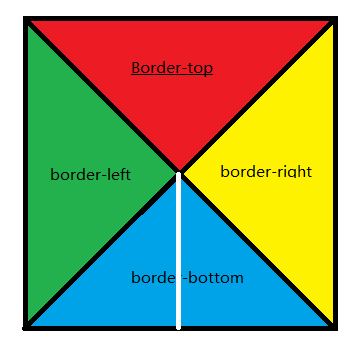
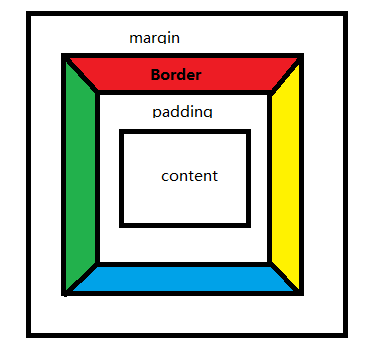
1:隐藏不占位

4 隐藏占位

## border三角形

### 原理

实际内部空间为0时的极限值。



### 实现

width:0px;

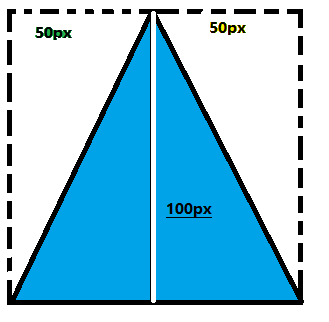
height:0px;

border-left: 50px solid transparent;

border-right: 50px solid transparent;

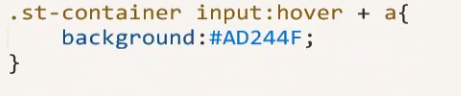
border-bottom: 100px solid red;

简写：**border-width:0,50px,100px(上，右左，下)**

****

## :hover技巧

被选中的选项获得焦点时不应该变化





# 行内元素和块级元素

1.行内元素与块级元素直观上的区别

行内元素会在一条直线上排列，都是同一行的，水平方向排列

块级元素各占据一行，垂直方向排列。块级元素从新行开始结束接着一个断行。

2.块级元素可以包含行内元素和块级元素。行内元素不能包含块级元素。

3.行内元素与块级元素属性的不同，主要是盒模型属性上

**行内元素设置width无效，height无效(可以设置line-height)，margin上下无效，padding上下无效**

# 清除浮动三种方法

## 过去

 方法一：添加新的元素 、应用 clear：both；

下一标签直接清浮动兄弟标签浮动时，在下一标签的属性中直接写入清除clear:both;这样就可以清除以上标签的浮动而不用加入空标签来清除浮动。

方法二：父级div定义 overflow: auto（注意：是父级div也就是这里的  div.outer）

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亲）

第一种可能出问题

.fn-clear{

    content: " ";

    clear: both;

    display: block;

    font-size: 0;

    visibility: hidden;

    height: 0;

}

.fn-clear {

    zoom: 1;

}

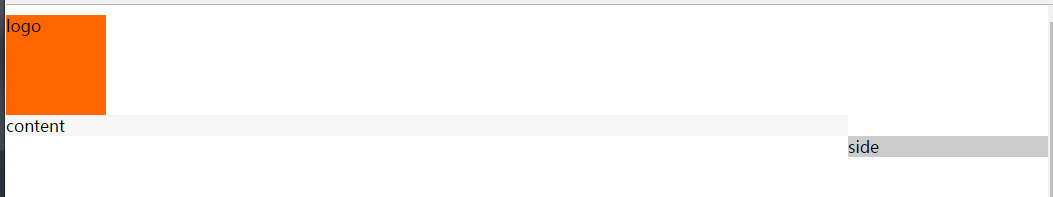
<div class="header">

    <div class="logo">logo</div>

    <div class="username">username</div>

</div>

<div class="fn-clear"></div>



username没有显示出来！

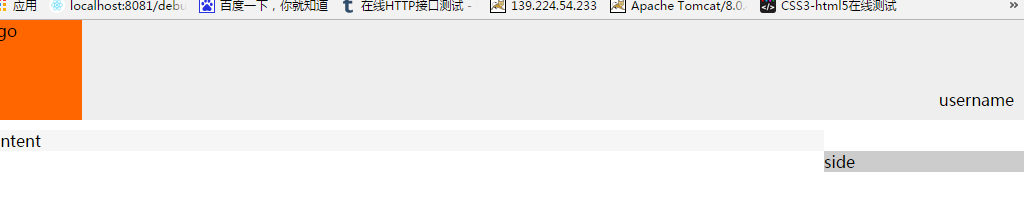
<div class="header fn-clear">

    <div class="logo">logo</div>

    <div class="username">username</div>

</div>

.fn-clear::after{



## 最新

原理：浮动会造成宽高塌陷，浏览器无法自动计算浮动元素宽高，导致其它元素和浮动元素重叠，解决方法的原理都是让浏览器重新计算宽高。

overflow：不指定高度时，宽高塌陷，迫使浏览器重新计算，从而顺便清除float的影响。

1.BFC 全称是块级排版上下文，用于对块级元素排版，默认情况下只有根元素（body)一个块级上下文，但是如果一个块级元素 设置了float:left，overflow:hidden或position:absolute样式，就会为这个块级元素生成一个独立的块圾上下文，就像在window全局作用域内，声明了一个function就会生成 一个独立的作用域一样，新开僻的BFC像是开僻了一个新的小宇宙，使这个块圾元素内部的排版完全独立隔绝。  
2.独立的块级上下文可以包裹浮动流，全部浮动子元素也不会引起容器高度塌陷，也就是说包含块会把浮动元素的高度也计算在内，所以不用清除浮动来撑起高度。

### 注意

absolute不设置left之类的话，和正常元素一样显示。（相当于absolute的bfc里面默认值就是正常的位置。），

新bfc对之前的元素无影响。对之后的元素有影响，不能计算之后元素的宽度。

一般的清除浮动,未指定left,top的absolute。

1.父级div定义 伪类:after 和 zoom  
代码：

<style type="text/css">  
.div1{background:#000080;border:1px solid red;}  
.div2{background:#800080;border:1px solid red;height:100px;margin-top:10px}  
.left{float:left;width:20%;height:200px;background:#DDD}  
.right{float:right;width:30%;height:80px;background:#DDD}  
/*清除浮动代码*/  
.clearfloat:after{display:block;clear:both;content:"";visibility:hidden;height:0}  
.clearfloat{zoom:1}  
</style>

<div class="div1 clearfloat">

<div class="left">Left</div>

<div class="right">Right</div>  
</div>

<div class="div2">  
 div2  
</div>  
原理：IE8以上和非IE浏览器才支持:after，zoom(IE转有属性)可解决ie6,ie7浮动问题  
优点：浏览器支持好、不容易出现怪问题（目前：大型网站都有使用，如：腾迅，网易，新浪等等）  
缺点：代码多、不少初学者不理解原理，要两句代码结合使用才能让主流浏览器都支持。  
建议：推荐使用，建议定义公共类，以减少CSS代码。

2.父级div定义 overflow:hidden  
代码：

<style type="text/css">  
.div1{background:#000080;border:1px solid red;width:98%;overflow:hidden}  
.div2{background:#800080;border:1px solid red;height:100px;margin-top:10px;width:98%}  
.left{float:left;width:20%;height:200px;background:#DDD}  
.right{float:right;width:30%;height:80px;background:#DDD}  
</style>

<div class="div1">

<div class="left">Left</div>

<div class="right">Right</div>  
</div>

<div class="div2">  
div2  
</div>  
原理：必须定义width或zoom:1，同时不能定义height，使用overflow:hidden时，浏览器会自动检查浮动区域的高度  
优点：简单、代码少、浏览器支持好  
缺点：不能和position配合使用，因为超出的尺寸的会被隐藏。  
建议：只推荐没有使用position或对overflow:hidden理解比较深的朋友使用。

3.父级div定义 overflow:auto  
代码：

<style type="text/css">  
.div1{background:#000080;border:1px solid red;/*解决代码*/width:98%;overflow:auto}  
.div2{background:#800080;border:1px solid red;height:100px;margin-top:10px;width:98%}  
.left{float:left;width:20%;height:200px;background:#DDD}  
.right{float:right;width:30%;height:80px;background:#DDD}  
</style>

<div class="div1">

<div class="left">Left</div>

<div class="right">Right</div>  
</div>

<div class="div2">  
div2  
</div>  
原理：必须定义width或zoom:1，同时不能定义height，使用overflow:auto时，浏览器 会自动检查浮动区域的高度  
优点：简单、代码少、浏览器支持好  
缺点：内部宽高超过父级div时，会出现滚动条。  
建议：不推荐使用，如果你需要出现滚动条或者确保你的代码不会出现滚动条就使用吧。

4.父级div定义 height  
代码：

<style type="text/css">  
.div1{background:#000080;border:1px solid red;/*解决代码*/height:200px;}  
.div2{background:#800080;border:1px solid red;height:100px;margin-top:10px}  
.left{float:left;width:20%;height:200px;background:#DDD}  
.right{float:right;width:30%;height:80px;background:#DDD}  
</style>

<div class="div1">

<div class="left">Left</div>

<div class="right">Right</div>  
</div>

<div class="div2">  
div2  
</div>  
原理：父级div手动定义height，就解决了父级div无法自动获取到高度的问题。  
优点：简单、代码少、容易掌握  
缺点：只适合高度固定的布局，要给出精确的高度，如果高度和父级div不一样时，会产生问题  
建议：不推荐使用，只建议高度固定的布局时使用

5.结尾处加空div标签 clear:both  
代码：

<style type="text/css">  
.div1{background:#000080;border:1px solid red}  
.div2{background:#800080;border:1px solid red;height:100px;margin-top:10px}  
.left{float:left;width:20%;height:200px;background:#DDD}  
.right{float:right;width:30%;height:80px;background:#DDD}  
/*清除浮动代码*/  
.clearfloat{clear:both}  
</style>

<div class="div1">

<div class="left">Left</div>

<div class="right">Right</div>

<div class="clearfloat"></div>  
</div>

<div class="div2">  
div2  
</div>  
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  
建议：不推荐使用，但此方法是以前主要使用的一种清除浮动方法

# css3动画

## 原理

transition和animation是动画。

transform是效果，translate是其中一种效果（平移）

通常动画和效果要结合使用。

transition的动画需要事件去触发（如鼠标经过、点击）  
animation的动画不需要触发事件就可发生

## transiton

## 使用

#st-control-1:checked ~ .st-scroll{

-webkit-transform:translateY(0%);

-moz-transform:translateY(0%);

**transform:translateY(0%);**

}

.st-scroll{

left:0;

top:0;

-webkit-transform:translate3d(0,0,0);

-moz-transform:translate3d(0,0,0);

**transform:translate3d(0,0,0);**

-webkit-backface-visibility:hidden;

-webkit-transition:all 0.6s ease-in-out;

-moz-transition:all 0.6s ease-in-out;

**transition:all 0.6s ease-in-out;**

}

### 注意

### 没有效果的情况

display:none

width/height: auto

无法计算宽高

## animation

### 使用

#st-control-5:checked ~ .st-scroll #st-panel-5 h2{

-webkit-animation:moveDown 0.6s ease-in-out 0.2s backwards;

-moz-animation:moveDown 0.6s ease-in-out 0.2s backwards;

-o-animation:moveDown 0.6s ease-in-out 0.2s backwards;

-ms-animation:moveDown 0.6s ease-in-out 0.2s backwards;

**animation:moveDown 0.6s ease-in-out 0.2s backwards;**

}

@**-ms**-keyframes **moveDown**{

0%{

**-ms-transform:translateY(-40px);**

opacity:0;

}

100%{

-ms-transform:translateY(0px);

opacity:1;

}

}

# 边界重叠

# 属性

## border

宽度属于外边距

# 兼容性

## display

display: inline-block;

\*display:inline;

\*zoom:1;

# 布局

## 间隙

### letter-spacing

0是默认宽度，可以用负数。

# 运算

## 运算符

css原生不支持运算符，使用less/sass后支持。

## calc

css3的属性，用于运算。

 使用“+”、“-”、“\*” 和 “/”四则运算；

 可以使用百分比、px、em、rem等单位；

 可以混合使用各种单位进行计算；

 表达式中有“+”和“-”时，其前后必须要有空格，如"widht: calc(12%+5em)"这种没有空格的写法是错误的；

 表达式中有“\*”和“/”时，其前后可以没有空格，但建议留有空格。

# 媒体查询

当媒体查询返回false，<link> 标签指向的样式表也将会被下载(但是它们不会被应用)

媒体查询是大小写不敏感的，包含未知媒体类型的查询通常返回假。

如果浏览器运行的设备上没有某个属性值，包含这个属性值的表达式一般返回假。例如，在语音合成器上查询屏幕长宽比总是返回假。

## 逻辑操作符and,not,only,逗号

### 逗号

逗号相当于or

@media (min-width: 700px), handheld and (orientation: landscape) { ... }

### not

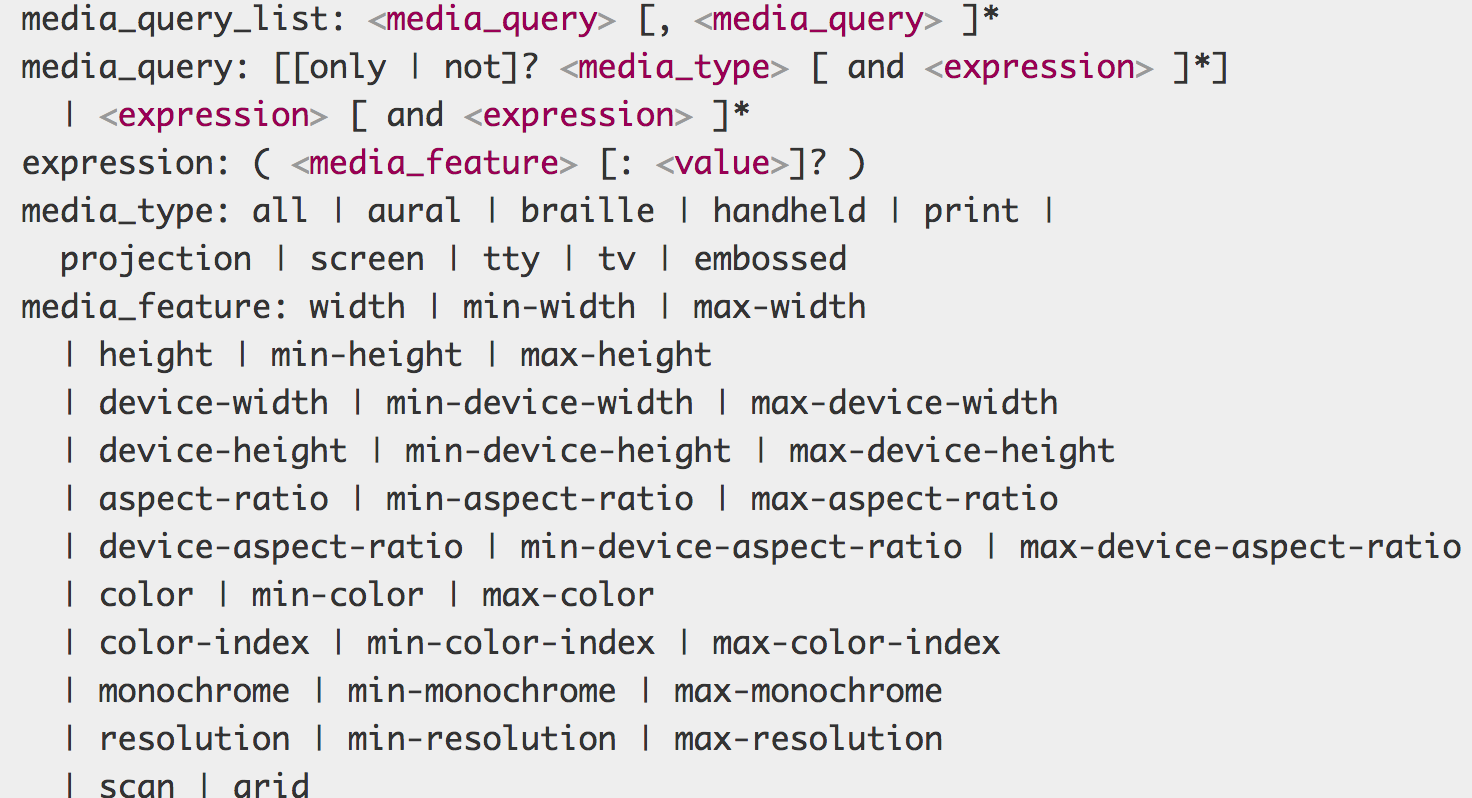
not关键字仅能应用于整个查询，而不能单独应用于一个独立的查询。不包括逗号。

### noly

指定某种特定的媒体类型，可以用来排除不支持媒体查询的浏览器。比如only screen and xxx表示只在支持screen查询的设备上继续查询，防止老旧的浏览器不支持带媒体属性的查询而应用到给定的样式

## 规则

表达式需要用圆括号。没有使用的会引起错误。



## 使用场景

方法一

<link href="style.css" media="screen print" ...

方法二

<?xml-stylesheet media="screen" href="style.css"...

方法三

@import url("style.css") screen;

方法四

<style media="screen">

@import url("style.css");

</style>

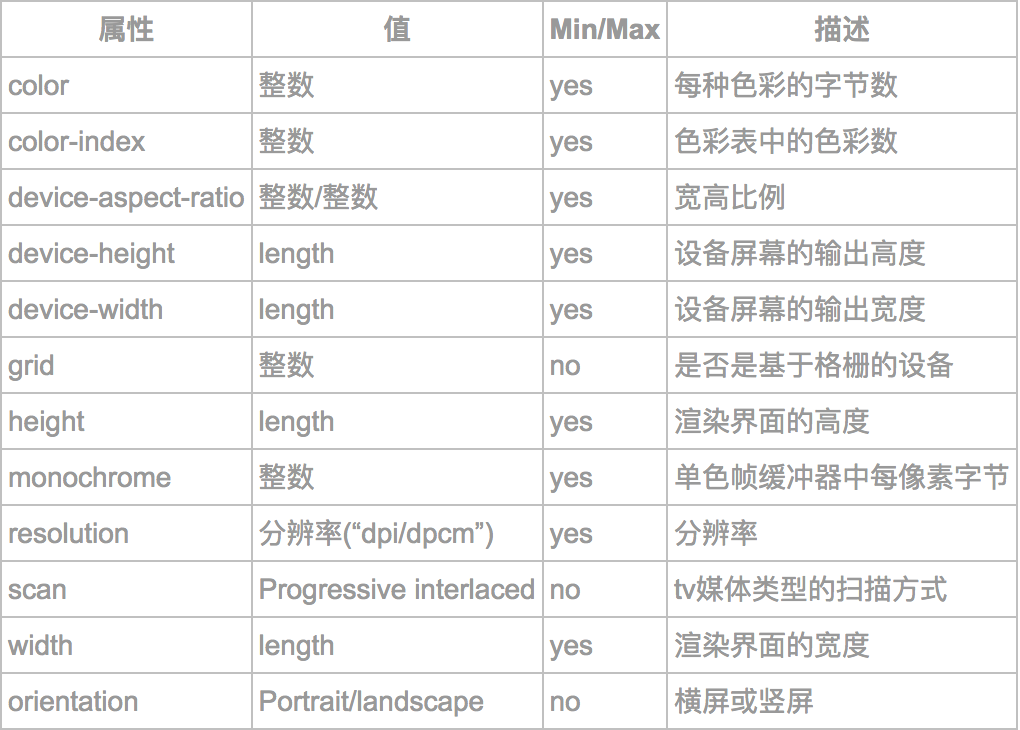
方法五

@media screen{

selector{rules}

}

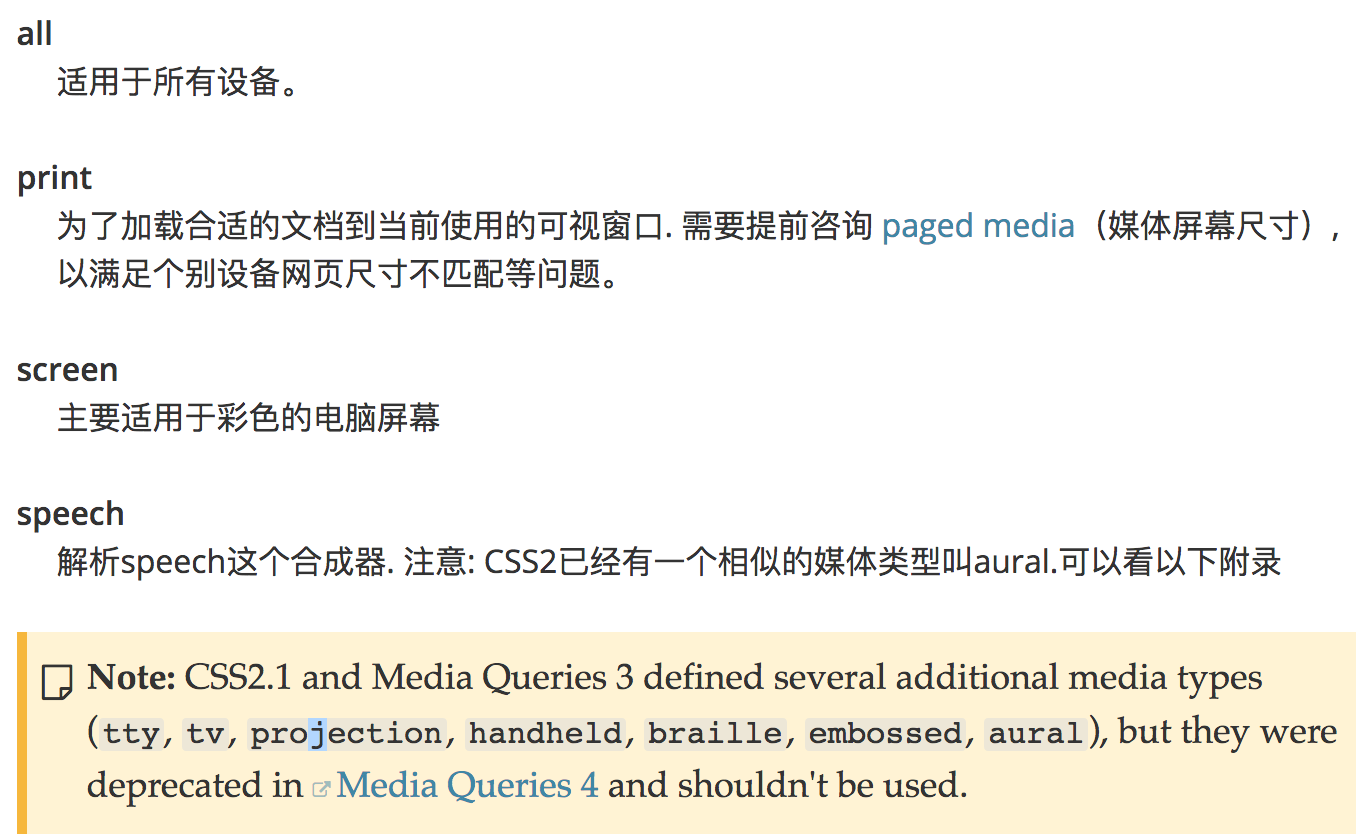
## 媒体特征



### orientation

值：landscape | portrait

## 媒体类型





# bfc

## 常见问题

### 高度变成0

父节点不设高度，子节点脱离文档流，（常见父节点relative，子节点absolute），导致无法计算子节点高度，也就无法撑开父节点，所以父节点高度为零。

**解决方法： 部分absolute**

# 绝对相对定位

## 绝对定位

### 高度设置

能够为绝对定位设置高度吗?

# 移动端

## 界面适配

### rem

原理是百分比

比如通过js拿到宽度xpx , 设计稿750px，1px / 750px = 1rem \* q / x => q = 1/750 \* x = x /750

(function (win,doc){

if (!win.addEventListener) return;

var html=document.documentElement;

function setFont()

{

var cliWidth=html.clientWidth;

html.style.fontSize=100\*(cliWidth/750)+'px';

// 乘了100，对应的rem就缩小100

}

win.addEventListener('resize',setFont,false);

setFont();

})(window,document);