CSS初探

2017年3月23日 9:02

CSS 层叠样式表

冲突的声明要通过这个层叠过程排序，并由此确定最终的文档表示。

这个过程的核心是 选择器、相关声明的权重、以及继承机制。

专为UI开发,css的出现解决了UI问题。

以至于不少非web平台也开始支持css的一部分子集了

微信小程序 ：wxml wcss（ 内部还是转换为css ） js

FireFox的Chrome：早期没有谷歌浏览器的时候，chrome指代的是浏览器自己的UI

CSS Zen Garden：同一套html用不同的css实现了完全不一样的页面效果，对css进行了很好的推广。

px是一个长度单位，当px与你屏幕分辨率一样时，一个px就是你屏幕上的一个点。

Css不区分大小写

替换元素：内容被其他不在文档里的内容替换了的元素，如 img 、 video/audio、 checkbox、 input、 iframe 、 canvas

(<img src="a.jpg">、 <video src="a.mp4"></video>、 <input type="checkbox">)

没有后代元素(html, js)

没有伪元素(css)

有内在宽高

当iframe元素子元素生效时，说明浏览器不支持该元素。

非替换元素：其内容直接出现在其标记中，其他大部分元素都是非替换元素。

块级元素：会占满父级元素的宽度，不会让其他元素在他的旁边。有p、 div、 h1

display属性可以改变元素的显示角色， inline/block

行内/内联（ inline ）:即在一行文字内产生的元素框，如em,a,strong标签都是行内元素

他们不会在其前后产生换行于是也就不会打断所在行的文字

在以前某些版本的html中，默认情况下内联元素是不能作为默认情况下块级元素的祖先元素的。如   ，在css中没有这个限制。

link： media属性 ，<link rel="stylesheet" href="a.css" media=print> ，意思即为这个样式表仅仅在打印的时候生效。比如一些超链接也可以在打印的时候显示。

media值：screen、 tty(字符显示设备/终端)、 print。可用逗号分隔这些属性media="screen,tv"

内联样式：style直接在标签里。

@import指令：必须出现在文件头/顶部，在css文件中。

@import "../a.css" ，相对路径

style标签里也可以用import，如<style>@import "a.css"</style>，该a.css即为在该代码目录下。

@import会形成一个树状依赖，即如果a.css里面引用了b.css，则需要先下载a.css，然后在下载b.css

css文件注释：/* */

也可以通过改变属性名称，让浏览器无法识别该属性而自动忽略。

规范：css每个属性后都加分号，以及文件最后加个空行。

选择器、优先级

2017年3月23日 11:18

选择器

把同一组属性应用于多种元素：p,div,h1,*{color:red}，*通配选择器

选择器对大小写敏感，但有些老浏览器不敏感。

标签/元素选择器 ：span{}

类选择器：.class ，*.foo简写为.foo，通配选择器可以忽略，所有选择器前面都省略了通配选择器

复合类选择器：p.class1.class2 即需要p满足 123

如p.a.a {color:red}

p.a{color:blue}，前面的优先级高于后者。

ie7之前不支持，只会有最后一个类生效，即p.a.b相当于p.b；

id选择器：一个元素只能有一个id,不存在复合id选择器

p.a.b#foo需要满足p标签拥有a,b类，并且id为foo

属性选择器：七种。

是否存在：p[name]{color:red} 即要求p标签为 123

属性名与值完全匹配：p[class="a b"]{color:red} 即要求p标签为

（css3）以开始： [name^="abc"]{color:red}，即name属性值要以abc开头，

以结束：[href$=".pdf"]{color:red}，即href属性值要以.pdf结尾，

包含：[href*="abc"]{color:red}，即href属性中包含abc即可，

属性值列表里包含：[data-like~="orange"]{color:red}，data-like属性列表里包含orange这个值

属性值等于或以开头并加一个"-"：[lang|="zh"]{color:red;}，lang属性以'zh-'开头或者为'zh'， 123

其他：

属性值不区分大小写：[attr="abc" i] case insenstive 大小写不敏感，属性名称默认不区分大小写。

后代选择器：h1 em {} h1所有后代里的em

子元素选择器 ：h1 > strong ，h1里所有作为第一代的strong，直接子元素，即若strong是包含在其他元素内的，无法选中

邻接选择器：h1 + p ,意思为选择所有跟h1位兄弟节点的p节点，如 <h1></h1> fsfe ，只能向后选，不能向前选。

只能选择两个相邻兄弟中的第二个元素。

h1~p ，h1后面所有的p元素，并且跟h1是兄弟关系

例子：

```
div ul >li~li {
 color:red:
 }
```

效果：（li~li意思为li前面必须有个li，选中的必须是作为弟弟的li）

```
<div>
  <ul>
    <li>1</li>
    <li>2</li>
    <li>3</li>
  </ul>
 </div>
```

注意区分：

.a .b .c {} 表示选择a类元素的所有b类元素的所有c类元素

.a.b.c{} 同时满足.a.b.c

.a,.b,.c{}表示三个选择器

伪类选择器（pseudo class）：需要满足某个条件时才生效， ":" 是伪类或伪元素的名片。

a伪类 ie6只有a标签支持这四个选择器

当设置的是同一个属性，a要按照如下顺序（lv ha）写才能有作用，否则无关顺序

--href属性有值，并且未访问的时候

```
a:link{
  color:yellow;
```

```
}
--访问过之后，visited里只有color/bg-color属性可以改（除了透明色），涉及到安全隐私性，因为js能探测其他变化，但无法探测颜色的变化。
 如：a span{display:none}; a:visited span{display:inline}无效，且不能渐变，不能设置opacity
a:visited{
color:grey;
}
--移动到a上时
a:hover{
   color:red;
}
--点击的时候
a:active{
  color:aqua;
}
```

结合伪类：

a:link:hover{color:red} 鼠标移动到未访问过的a标签上显示红色，顺序无关

注：不能把互斥的伪类结合在一起使用，如a:link:visited{}，一个链接不能同时是已访问和未访问

focus伪类

所有只要获取到焦点的元素就会生效

```
:focus{
backgound:red;
}
```

位置伪类：（css3）

:first-child 第一个子节点，如

p:first-child 选择的是作为第一个子节点的p元素

p>:first-child 选择的是p的第一个子节点

p :first-child p标签下的所有作为第一个子元素后代

a:last-child 最后一个子节点且为a标签

a:last-of-type 选择所有a元素里面最后一个a元素

a:first-of-type 选中某个元素中第一个为a的元素

:nth-child(2) 第二个p元素 ，（从1开始）

:nth-child(odd) 奇数的子元素

:nth-child(even) 偶数的子元素

:nth-child(3n) 取3的倍数的子元素，里面可以为任意函数

:nth-child(3n+1) 取3的倍数+1的子元素，n可以从0开始

:nth-last-child(3) 倒数第3个元素

p:only-child 父元素中唯一的p标签 ， <a>

p:only-of-type 父元素里有且仅有一个p标签

p:nth-of-type(2) 选择p元素中第二个p元素

例子：

```
/* 有一个ul里有若干个li，使用选择器实现当且仅当li个数为8个时，li的颜色全部为红。 */
li:nth-child(1):nth-last-child(8),
li:nth-child(1):nth-last-child(8)~li{
  color:red;
}
```

选择器取反

:not([value]) {}　　取没有value属性的

选择文档的跟元素

:root {}，在html里面相当于是html根元素

空的div元素

div:empty{} ，如

target伪类

:target{color:red} 选择id的值与地址里#后面的内容相同的元素，一般用来做页面内的高亮。
即若页面中跳转到第五章内容，则第五章内容就变成你设置的样式
input伪类
input:valid{border-color:green} 内容输入正确的时候，如<input type="text" minlength=5 required>输入不小于五个长度时生效
input:invalid{border-color:red} 内容输入不合法的时候
input:required{} 内容要求必填的时候
input:optional{} 可选填的
input:enabled{} 启用的时候
input:disabled{ cursor:not-allowed;} 禁用的时候鼠标状态变成禁用
input:checked{zoom:2} checkbox被选中之后放大两倍
例子：input 选中之后span才显示

```
input+span{display:none}
input:checked+span{
display:inline;
}
```

选中内容之后的伪类
::selection{color:red} 被用户选中部分的内容/文字，颜色变成红色，基本上只能设置颜色
语言伪类
:lang(fr){font-style:italic;} 把所有法语元素变成斜体

更多伪类伪元素：
:empty{}空元素时,
:only-child{}一个子元素时
更多可查看： https://www.smashingmagazine.com/2016/05/an-ultimate-guide-to-css-pseudo-classes-and-pseudo-elements/

伪元素选择器：能够在文档中插入假想的元素<before>，<after>，从而得到某种效果。(: 或 ::都可以)
作为元素的子元素存在。
替换元素没有伪元素，因为替换元素没有子元素。
一个选择器中只能出现一次，并且只能在选择器最后出现。即p::before:hover{} 无效，可以为 p:hover::before{} 即意为鼠标移动时才生效

设置首字母样式
p:first-letter{color:red;} 使每个p的第一个字母变成红色
注：连带选中前面的标点符号，第二个字母前所有的内容，可以浮动（实现报纸首字母很大的效果）
元素的display计算值必须是 block, inline-block, table-cell, list-item或者table-caption才生效，span就没用！！
具体可查看文章：http://www.zhangxinxu.com/wordpress/2016/09/css-first-letter-pseudo-element/
设置第一行样式
p::first-line{color:purple} 使每个p的第一行文本都变成紫色
设置之前元素的样式，
h2::before{content:"lala";color:silver;} 选中h2前的before元素，所有h2前都会插入lala字符串
h2::before{content:"lala" url(a.jpg) '123123';} 内容中增加图片
a::before{ content:'(' attr(href) ')';} 显示a链接的href
a::before{ content:'\A(' attr(href) ')';white-space:pre; } 换行显示a链接的href
设置之后的元素样式
body::after{content:" the end ";} 选中body后的after元素，body最后会加上the end字符串
选择器的优先级：四位的无穷进制数，永远不进位！ 即0,0,1,0 > 0,0,0,17
标签选择器、伪元素：0,0,0,1
类选择器、伪类：0,0,1,0
属性选择器：0,0,1,1
id选择器：0,1,0,0,
内联样式/行间样式：1,0,0,0，既样式写在标签里，内联样式表（标签内部）> 嵌入样式表（当前文件中）> 外部样式表（外部文件中）
*通配符：0,0,0,0，div p 与div * p 优先级一样，前者为所有后代p,后者为所有孙子代p
继承上级选择器标签优先级为空
当选择器中优先级相同（如同个选择器）的时候，后出现的优先级高
> + - 等结合符不参与优先级计算

继承：元素会把css的值继承到后代元素，但不会向上传播，有一个例外，应用到body元素的背景样式可以传递

相应地可以定义其画布。某些属性不能继承，如border，大多数框模型属性（外边框、内边距、背景和边框）都不能继承，否则会造成结构混乱。

例子
p>span{} 0002
p span{} 0002 所以相同

.b.b{} 0020
.b{} 0010 所以.b.b大于.b

!important 优先级最高
p{color:green !important;} 总是放在声明的最后，分号前面，否则无效。

浏览器优先级：
每个浏览器都有自己默认的样式，并且可以修改，也可以选择自己的css来决定浏览器的样式。
没有important，Author网站作者 > Customize用户 > UserAgent 浏览器(用户代理)
有important ，Customize > Author > UserAgent

不来自css的样式优先级
如font标签，优先级为0
并且出现在作者样式的开头，会被作者样式和读者样式覆盖，但不会被ua（浏览器）样式覆盖
123 该color非内联样式，内联样式是写在style里的
<style>*{color:green;}</style> 最后为绿色

优先级例题：问p最后字体是多大？

```
<style>
 #a{font-size: 12px;}
 div p{font-size: 13px;}
 div .c{font-size: 14px;}
 .a .b .c{font-size: 15px;}
 #b{font-size: 16px;}
</style>
<div id="a" class="a">
 <div id="b" class="b">
  <p id="c" class="c">Hello world!</p>
 </div>
</div>
```

答：因#a,#b并未选中p元素，都是继承下来的样式，所以优先级为0
div p优先级为0002，div .c优先级为0011，.a.b.c优先级为0030，所以字体为15px!

练习：
div:hover:after{}
div在被hover时的after伪元素

body :hover{}
body里鼠标正按下去的那个元素，不限于可交互元素
注：ie6里，:hover伪类只用在a标签上

input +ul+p +span{}
input后面的一个ul后面的一个p（此三个必须存在）后面所有的同级span

#some #thing .not:hover .abc:hover{}
优先级为 0240

第8个子结点之后，倒数第5个子结点之前的li结点
li:nth-child(n+9):last-nth-child(n+6){};

【类名】以“damiao-”开头的元素
[class*=" damioa-"],[class^="damiao-"]{}

【类名】以“damiao”结束的元素
[class*="damiao "],[class$="damiao"]{}

rel属性中有nofollow这个单词的标签
[rel~="nofollow"]{}
同
[rel^="nofollow "],
[rel*=" nofollow "],
[rel$="nofollow "]{}

值与单位

2017年3月24日　　10:19

数字：如line-height:2;这里2不等价与200%
百分比：width:60%;需要知道基数是什么
百分比与纯数字不可互换
颜色：　#RRGGBB #(0-255)(0-255)(0-255)　--->16700000
　　#abc -> #aabbcc　十六进制　如#ffffff 即 #fff白色，#000黑色
　　rgb(r,g,b)　　如 rgb(100%,100%,100%)白色　　rgb(0%,0%,0%)黑色
　　rgb(0-255,0-255,0-255)　如　rgb(255,255,255)白色　rgb(0,0,0)黑色
　　rgb(r%,g%,0-100%)
　　rgba(r,g,b,alpha) ,alpha为透明度，范围为0-1
　　green/lightgreen/darkblue
　早期电脑只支持256种颜色，web safe 颜色：216种，在计算机系统上能避免抖动的颜色。
color:rbga(...)
HSL、HSV色彩空间
hsl:
　H色相hue，即颜色类型，0—360，其中每一个值代表一种颜色
　S饱和度saturation，指色彩的纯度，越高色彩越纯，低则逐渐变灰，取0-100%的数值
　L亮度lightness，取0-100%
　如color:hsl(55,50%,50%)
hsla:色相，饱和度，亮度，透明度
hsv:色相，饱和度，明度或色调（value）

若需要把彩色图片转换为黑白，就需要调节这个属性，不过需要先把rgb转换为hsl，好了之后再转换为rgb。待了解rgb与hsl颜色之前的转换。

绝对长度单位：
in（ch）英寸，
cm 厘米 1in是2.45cm 1cm=0.394in
mm 毫米
mozmm(火狐专用)，
pt 点point，印刷度量单位 72分之一in
pc 派卡pica，印刷术语，6分之一in。
存在的问题：大部分时候不准，取决于你的分辨率以及系统设置，因此用的比较少，但在打印的时候可以比较准。

相对长度单位
　　px：一个css像素，显示器的分辨率与操作系统分辨率一致时，网页不放大的情况下，一个px就是你屏幕上的一个点。反之，需要算出你显示器分辨率对比操作系统的值为显示器的物理像素，如果网页未放大，得到的值就为px的值。

　　指定的图片的大小一般用px，要不然图片会被变形拉伸，因为图片的尺寸大多数时候是以px来丈量的。
　　坑：ie7之前的浏览器，放大时以px指定的文字不会放大，不过已经不在考虑范围了。

　　例：显示器的物理分辨率为1920x1080，操作系统设置的分辨率为1280x720，网页的放大倍数为110%，请计算一个CSS像素对应多少个显示器物理像素（面积与长度）？
　　width:1920/1280=1.5
　　height:1080/720=1.5
　　放大后：w=1.5*1.1=1.65

s=1.65^2=2.72

em：相对于当前元素font-size的大小，层层继承，若其中一层修改，则其子元素会受影响，p{font-size:1.5em}
rem：root element's em，相对于html（根）元素的字体大小（若html{font-size:2rem}，则为相对浏览器默认字体大小的两倍，谷歌为16px）。只需修改html标签的字体大小，所有其他为rem的设置都会变
ex：x字符的高度，几乎没用，ex=0.5em,因为不同的浏览器字体高度都不一样，有些浏览器会把它计算成0.5em
ch：0字符的宽度
vw/vh：viewpoint width /viewpoint height,相对于适口高度的大小，视口宽/高的100之一,手机上支持的比较好；若页面有滚动条，则包含滚动条宽度
vmax/vmin：vmax视口宽或者高较大的那一个的100之一
vmin视口宽或者高较小的那一个的100之一
calc(100%-5px)：按照实际情况计算，都转换为px
calc(100vw-100%) :100vw为包含滚动条时的宽度，100%为不包含滚动条页面的宽度，所以得到滚动条的宽度
注：为什么不用cm,mm等长度单位
这些单位都是绝对物理单位
但是，它们在css里渲染不到跟物理世界一样的尺寸，所不常用
px本就是相对单位
dip(device independent pixel)

角度单位
degree 角度 45deg
grad 梯度 100grad=90deg=1.57rad
radian 弧度 3.14rad=180deg 即π rad = 180deg
turn 圈 1turn=360deg=6.28rad

时间单位
1s
1.2s
.2s == 0.2s 前面的0可以省略
1ms

URL :url(path)，相对路径，绝对路径
css关键字： display:none ，font-size:inherit
inherit，使一个属性的值与其父元素的值相同，这个关键字所有属性共有
例子：

```
<div id="toolbar">
 <a href="1">1</a>
 <a href="2">2</a>
</div>
```

```
#toolbar{
 background:blue;
 color:white
}
#toolbar a{  --为保证链接的文字不因浏览器默认的样式也显示蓝色而看不到
 color:inherit;
}
```

字符串：content:'123';
取属性的值
content:attr(href); attr函数可以获取href的值

字体

2017年3月24日 16:04

字体 font-family
字体系列：serif、sans-serif、monospace、cursive、fantasy,每个字体系列都包括大量字体。
serif 衬线字体 （有三角状、上下短线），具体有Times、Georgia、New Century Schoolbook
sans-serif 非衬线字体，具体有Helvetica、Geneva、Verdana、Arial、Univers
monospaceliw 等宽字体，具体有Courier、Courier New、Andale Mono
cursive 类似人手写的字体，会有小弯曲，具体有Zapf Chancery、Author

body{font-family:sans-serif} ：浏览器会自动选择一款非衬线字体
h1{font-family:Verdana,Geneva,Arial,sans-serif;}
设置候选字体，按照顺序会根据用户电脑上的字体进行选择。
sans-serif不是字体，如果前几个字体找不到，浏览器就会选择一个实际字体来代替sans-serif，取代他的字体是浏览器定义的该字体系列的默认字体。

字体名称需要加引号，字体系列不需要加引号，否则会被当成字体名称。

为中英文设置字体
p{font-family:'Helvetica', '微软雅黑'}
一般英文字体无法识别中文，如:Helvetica这种字体在自己的字体库中没找到中文,只找到英文,所以,这种字体只能渲染英文数字和一些特殊符号,而页面中的中文就会自动调用第二种字体 '微软雅黑'
把英文字体放在前面，中文字体放在后面，否则英文字体将无效（不包括微软雅黑这种本来就包括中文字体跟英文字体的）

font-family：linux,mac,windows ，一套字体想在三个系统上使用，一般顺序linx mac windows

字重：font-weight
- normal
- bold
- bolder
- lighter
- 100-900 100最细，900最粗，400等价于normal，700等价于bold
- inherit

字号：font-size
继承的是计算后的值，而不是书写时的值，即若其值是10%，则他的字体大小为父级的字体大小*10%
绝对大小：xx-small x-small small medium large x-large xx-large
根据规范，一个绝对大小与相邻的绝对大小的缩放因子是向上1.5及向下0.66，如果medium是10px,那么large是15px
但因不同浏览器设置的缩放因子可能不一样，这个值开发者也没有办法更改，所以这几个关键字基本上不怎么用。
相对大小：larger smaller,相对于其父元素的大小
注：当用em设置到字号较小时，会发现浏览器上字号不在跟随父元素字号变化，原因可能是浏览器的默认最小字号设置了，所以需要调节浏览器的字号。

百分比：同em，相对于父亲元素字体的大小，120%跟1.2em效果基本一样
字体样式：font-style
normal
italic 倾斜，假如原来字体是宋体，italic会找到一个新字体宋体italic ，专门设计出来的斜体字
oblique 倾斜 ，在正体字体的基础上变倾斜

字体变形：font-variant
normal 默认
small-caps 把小写字母显示成小号的大写字母，有些字体专门为小写字母设计了这种样式，而不是单纯的把大写字母显示的小一点，当字体没有提供这种样式的时候浏览器当然就把大写字母缩小了

注意区分：text-transform:uppercase 把小写字母转换为大写，capitalize首字母大写

字体拉伸：font-stretch

normal wider narrower等

字体大小调整：font-size-adjust

数值|none|inherit

可以简写font:{} ,font-style,font-variant,font-weight 顺序不重要，如果值为normal，可以省略；

font-size ,font-family 顺序要确定，必须出现， font-size/line-height，可以跟行高一起写

如font:25px/1.2 "宋体";

例子：

h1,h2{font:italic small-caps 250% sans-serif;}

h2{font:200% sans-serif;}

h3{font-size:150%;}

此时h2不会产生斜体也不是小型大写字母，原因是因为

h2等价于{font:normal normal normal 200% sans-serif;}

所以h1的特性被覆盖。h3则不会有影响，因为h3用的是font-size

字体文件的保存格式

TrueType字体:.ttf

OpenType字体：.otf ，OpenType建立在TrueType基础之上

Embedded OpenType字体:.eot，Embedded OpenType是OpenType的一种压缩形式，这个格式是专用的（Microsoft）,仅IE提供支持

SVG字体： .svg ，Scalable Vector Graphics是一种通用图像格式

Web开放字体格式（最广泛）：.woff，建立在TrueType基础之上，大已经发展为Web字体的一个标准，

@font-face,内置的css规则，可以用来设置你要的字体。

可以指定多个，名字要唯一。加载比较耗时，移动设备和小型设备不支持web字体

```
@font-face{
font-family:"my font";
src:url("font.woff"),
    url("font.ttf");
}
h1{
font-family:"my font";
}
```

文本属性

2017年3月25日 15:37

text-indent：文本缩进，如text-indent:2em/20%，text-indent:-9999em可以用于隐藏文字
应用于块级元素
只应用于一个元素的第一行，即使插入了行分隔符
继承：从父元素计算，而不是在子元素上计算。

text-align：水平对齐，left(默认) center right justify
应用于块级元素
注意center与 元素的区别，前者只居中标签里的内容，后者会把整个元素居中
justify 两端对齐
为单行设置两端对齐：

```
p:after{
  content:'';
 display: inline-block; //要设置宽度必须设置为inline-block
  width:100%;
  height: 0px;
}
p{
  text-align: justify;
}
```

或在p内增加

line-height：行高 基线与基线之间的距离，(两行文本）
垂直居中：line-height值为height的值
继承：1em继承自父元素的font-size，1按照自己的font-size
例子：

```
body{font-size:20px;}
div{
  background-color:yellow;
  line-height:1em; p的line-height继承为20px ——> line-height:1; 则p的line-height继承1，即为30px
}
p{font-size:30px;}
<div>the first line i test lalalalalala
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Beatae impedit ex saepe amet a nulla consequatur recusandae aperiam?</p>
</div>
```

因为继承的原因导致字体重叠，
指定一个数时，缩放因子将是继承值而不是计算值，各元素会根据自己的font-size计算
，即改为 line-height:1;
注意：行高有让文字居中的特性，所以当行高为0时，则其元素实际占据的高度为0，而这个0的计算位置是content area的中线处。
https://jsbin.com/duqevemomu/edit?html,css,output

```
.dib-baseline {
  display: inline-block; width: 150px; height: 150px;
  border: 1px solid #cad5eb; background-color: #f0f3f9;
}
```

```
<span class="dib-baseline"></span>
<span class="dib-baseline">x-baseline</span>
```

vertical-align 垂直对齐 baseline top middle bottom继承自父元素的line-height

baseline 默认，有基线则与其父元素基线的那一行对齐，没有基线底端与其父元素的基线对齐

默认只应用于行内元素和替换元素（图片，文字、单元格、按钮）

例子：调整内容中的checbox的位置与底部对齐

```
input{
vertical-align:-4px;
margin:0;
}
```

因input中默认有margin，不为0的情况下回影响整个行高，所以要设置为0

IE6/IE7下vertical-align百分比值不支持小数line-height

注意问题：需要把父元素行高设为父元素的高度才够里面的元素居中，否则行框高度为父元素顶部到图片底部，无法再在父元素内部居中

```
img { vertical-align: middle; }
```

图片明明vertical-align:middle了，为什么不垂直居中啊？

行高设为容器高度值

不是vertical-align:middle没起作用，而是太短，不够居中。

```
<p><img src="小公主.jpg"></p>
```

```
p { display: table-cell; vertical-align: middle; }
```

图片近似垂直居中！

```
<div class="test-list">
    <span>文字</span>
    <img src="小公主.jpg">
</div>
```


```
.test-list > span { display:inline-block; width:210px;
vertical-align:middle; }
.test-list > img { vertical-align:middle; }
```

word-spacing 字间隔

leter-spacing 字母间隔

text-transform 文本转换

uppercase

lowercase

capitalize 首字母大写，复制粘贴后也为大写 ,类似的font-variant 复制粘贴后无变化

none

text-decoration 文本装饰

line-through 删除线 同<del>标签

underline 下滑线 同<ins>标签

overline　上划线
none

不会累加，需要对多个属性合并，才有效，如
em{text-decoration:underline overline}

怪异：因为不继承，在一个元素上设置的文本装饰意味着整个元素都有同样的颜色装饰，即使子元素有不同颜色。
解决办法：在外层嵌套一个元素，设置外层元素的装饰与颜色，然后再设置内层元素装饰与颜色。
且在子元素上设置装饰为none无效（父元素有装饰）。
如：p为section的子元素

```
section{
  color:yellow;
  text-decoration:underline;
}
p{
  color:red;
  text-decoration:underline;
}
```

text-shadow 文本阴影
text-shadow: x y r color； y为向下为正，r为模糊半径，可以省略；可以写多组
可以利用文本阴影来处理内容不被复制，即只显示阴影，不显示原文本，阴影文本无法被选中。

-webkit-text-stroke 空心字

```
span{
   font-size:10em;
   color:white;
   -webkit-text-stroke: 1px red;
}
```

box-shadow　盒子阴影
box-shadow:inset x y r1 r2 color； r1为模糊半径，r2为扩散半径
单向阴影 ,x或y为0, 使用负的扩散半径，正好跟模糊半径为相反数

white-space 处理空白符
nowrap强制不换行，可以用于tooltip提示框不换行
word-break:break-word 自动换行，一行显示不下的时候
overflow-wrap:原名叫word-wrap，同word-break;
word-wrap:break-word; 单词折断　变为

direction 控制文字书写方向
unicode-bidi　根据unicode控制文字顺序

writing-mode:文字排版方向　方案
默认值 horizontal-tb ,从左往右水平排版
vertical-rl 从右往左垂直排版，可用于诗歌

注意问题

css代码：

```
p{
  background-color: red;
}
img{
  width: 200px;
}
/* img{
  vertical-align:middle; //方法一，改变默认对齐方式基线对齐
}*/
/* img{
  display:block;   //方法二，改为块状元素，此时vertical-align对块状元素失效
  margin:auto;
} */
/*  p{
   line-height:0;//方法三，行高为0，基线跑到上面，不会撑开空白
 } */
/* p{
  font-size:0; //方法四，字体为0即行高为0
} */
```

html代码

```
 <p>
  <img src="http://img07.tooopen.com/images/20170314/tooopen_sy_201828895498.jpg" alt="">
</p>
```

ie7以下，否则无效

```
<p><img src="mm1.jpg"><!-- 这里要折行或空格 -->
</p>
```

基本视觉格式化

2017年3月29日 8:38

css3里边框可以为图像。

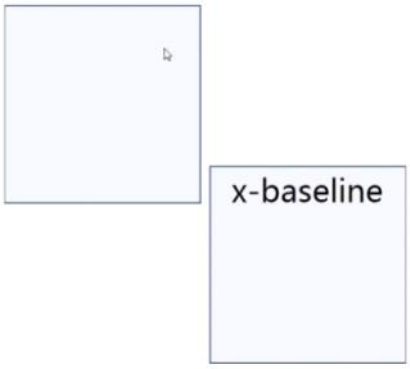
内边距、边框、内容宽度、高度不能为负值
只有外边距可以小于0。
边框的宽度不能是百分数，只能是长度。

水平格式化：
box-sizing:content-box/border-box
margin与padding取得都是包含块的宽度
替换元素为block时，如果width为auto，元素的宽度则是内容的固有宽度。

垂直格式化：
不声明border-style的情况下，上下内边距和边框默认为0
正常流中一个块元素的margin-top或margin-bottom设置为auto,则会自动计算为0
包含块高度在确定的情况下，内部的元素的百分数才生效。
常规流垂直方向上的块级元素的margin重合时会合并，合并结果为最大的那个margin。如果都是负值，则取绝对值最大的。

行内布局

基线：
对于文字来说，基线为x下方
替换元素来说，基线为margin-bottom下方
非替换元素
有内容：最后一行内容文字的基线
(当一个行内替换元素是块级元素或表单元格元素中的唯一后代，会默认块元素里有内容/文字的情况，基线以文字底部为准，如把图片放到div里，图片默认跟div里有文字时候的基线对齐)
无内容：基线为margin-bottom下方，同替换元素



最高点与最低点的确定：
行内：line-height-box的最上与最下
其他：margin-top的外边缘和margin-bottom的下边缘

对盒模型的理解
box-sizing属性会影响盒模型的计算方式
border-box 内容区域 的宽/高 是 width/height -padding -border
content-box 边框盒子的大小是 width/height+padding+border
当页面不申明doctype的时候，在ie低版本下，相当于box-sizing为border-box

为什么ie7份额比ie6小？
ie7只能装在xp上，而xp自带ie6，然后使用xp的用户一般不会升级浏览器
具体搜索 caniuse

页面有无doctype声明会有什么区别
无doctype页面会以怪异模式渲染，quirk mode
ie低版本下，box-sizing行为是border-box body
元素高度默认是视口高度

行内块元素 inline-block

如果width未定义，或者显示声明为auto,元素框会收缩以适应内容

对于他的兄弟元素来说，就是个Inline元素（从外看是inline元素，从内看是个块级元素，所以对他设置右margin时，动的是他的兄弟元素（父元素的行内元素也算），而不是它本身）
对于处于他里面的元素来说，他们认为自己处于一个块级元素里
i-b扮演了一个inline的元素，可以在行内水平排列

注意点：
baseline，有内容无内容，baseline不同
有文字时，最后一行与基线对齐；
无文字时，下margin的下边缘与基线对齐
用它做布局的时候，需要注意空格/字号问题
inline-block 元素开启了一个新的块级格式化上下文

后出现的元素不会影响前面已经出现的元素的渲染，后出现的行内元素会覆盖先出现的行内元素（当前面的内容超出他的父元素时），后出现的块元素不会覆盖先出现的行内元素！！

内边距、边框、外边距

2017年4月6日　　9:24

margin/padding/border
默认值都为0
margin/padding写百分比时都是相对父包含块内容区的宽度
margin不可见

长度为0px可以直接省略写为0（margin/padding/border）

min-width
只是设置元素的最小宽度，而不是视口的宽度，视口变小的时候元素下方会出现滚动条，css无法调节视口的宽度
视口变小到一定程度不能再变小，这是浏览器自身的设置，跟网页无关

负margin
例子：利用负margin设置最后的li无border(不用伪类)

```
li{border-bottom:1px solid;
  margin-bottom:-1px;
}
ul{
overflow:hidden;    /*因为负margin已经超出边界，所以用overflow:hidden就可以隐藏最后的border*/
}
```

padding
利用padding把元素设置在div水平中间

```
span{
display:inline-block;
min-width:x;
padding:0 calc( (100% - x)/2 );
}
div{
width:100%;
}
```

可通过padding-bottom设置为百分比来保持一个元素宽度的比例

border:
实现梯形、三角形
border足够粗，且没有高度，指定某一边border颜色，其他为透明(transparent)即可
如三角：

```
div{
  border:41px solid red;
  border-color:transparent;
  border-left-color:red;
  transform:translateY(-300px) rotate(270deg);
}
```

border-radius: a b c d;
border圆角，从左上角开始
border-top-left-radius:10px;
border-top-left-radius:10px 20px; 水平10px，垂直20px，椭圆

border-radius:0 20px; 上左跟下右为0，其余为20
注意 E { border-radius: [top-left] [top-right] [bottom-right] [bottom-left]; } 四个值
E { border-radius: [top-left] [top-right & bottom-left] [bottom-right]; } 三个值，上左，上右与下左，下右
E { border-radius: [top-left & bottom-right] [top-right & bottom-left]; }两个值，上左下右，上右下左
E { border-radius: [top-left & top-right & bottom-right & bottom-left]; }一个值

border-radius：50%；椭圆（w!=h）,圆（w=h）
border-radius:10px / 20px ; 水平方向四个角都是10px，垂直方向四个值都是20px;

translateY(-300px) 在Y轴上的距离

transform:rotate(20deg) 旋转

注：
边框过粗会盖住前面的元素，但是会被后面的元素盖住
因为浏览器是从上往下渲染，所以先画的会被后画的盖住

margin规则 都是相对宽度,只对块状元素
父width为400px
水平方向

margin-left	margin-right	width
auto=>200	100	100
100	100=>auto 200	100
auto=>150	auto=>150	100 ==>居中
auto=>0	100	auto=>300
auto=>0	auto=>0	auto=>400
		auto=> 内容宽（img）

垂直

父height	margin-top	margin-bottom	height 默认为内容高度
	auto=>0	auto=>0	
width=height	25%	25%	50% 居中
auto			50% =>auto

2017年4月6日 15:42

verdor prefix
厂商前缀，一些厂商实现了一些私有的css属性时，加上这个前缀
一些厂商提前实现了尚未进入标准的属性
为了避免与【未来的标准】产生不兼容

浏览器的私有属性

避免在属性进化的过程中浏览器识别不了

-ms-box-sizing ie
-o-box-sizing opera
-webkit-box-sizing chrome/safari
-moz-box-sizing firefox

IE hack 使用错误的css语法，只让IE的某些版本能够认识，用于处理ie特有css兼容性

```
a{
color:red;
*color:blue;    --IE6识别
+color:blue;    --IE6识别
color:yellow\9; --IE9识别IE6不识别
}
```

具体可参看：www.css88.com/archives/tag/css-hack

条件注释 conditional comment html的注释里写上特定的语法，让此段注释值在某些浏览器下生效，ie10已经不支持ie条件注释

```
<!--[if !IE]>
   <link rel="sytlesheet"  ref="b.css">
<![endif]-->

<!--[if IE 6]>
   <link rel="sytlesheet"  ref="a.css">
<![endif]-->
```

颜色、背景

2017年4月7日 10:12

color:

默认继承

border,box-shadow,text-shadow如果不指定颜色，都是使用color的值

currentColor 当前元素的颜色，如background-image:linear-gradient(currentColor,green)

<img src="" alt="description"> alt里面文字的颜色为color的颜色

背景图片一些属性设置：

```
background-color/image/position/size/origin/clip
repeat-x.y/attachment
```

background 多组属性同用

a覆盖b,最后的bgcolor在最底层 bg-position bg-size bg-origin bg-clip bg-color

background:url(a.jpg) url(b.jpg) no-repeat 50% 50% / 50% padding-box content-box #000;

bg-size在bg-position后，且用斜杠分开，bg-clip在bg-origin后

background-image:

linear-gradient(currentColor,green);渐变背景图片，从当前的颜色渐变到绿色

linear-gradient(-50deg, #07beea 20%,red, #444); 从某个角度开始渐变

linear-gradient(to left, #07beea 20%,red 73%,black 95%, #444); 从右边到左边渐变

linear-gradient(to top right, #000, #f00 50%, #090); 从左下角开始渐变到右上角

多组图，前面的覆盖后面的。如background-image:url(a.jpg),url(b.jpg); a.jpg覆盖b.jpg

背景图片：在背景颜色之上，文字之下

background-repeat:no-repeat;

repeat-y

space 不裁剪，等距(css3)，此时设置bg-p无效

round 不裁剪，无距，拉伸，整数个(css3)

如：bg-repeat:round space; 水平方向整数个，垂直方向有距离

background-repeat:no-repeat;

background-repeat:repeat-y;

background-color

background-size 图片大小

contain 保持其比例、完整显示，尽量大放进元素，等比例放大到某个方向上把元素盖满，不超出

cover 保持其比例，变大到足以盖住元素，从左上角开始渲染，覆盖，等比例放大到某个方向上把元素盖满，超出

background-size:40px 10px;

图片长宽，百分数则取的是paddingbox的比例，如果改变background-clip，相对box改变

如果只设置一个值，另一个为auto，即按原始比例缩放！

扩展，针对img，效果类似bg

object-fit:contain/cover/fill(默认)

object-position:top left;

background-position 图片定位

center 图片居中

bg-position:left 3px bottom 5px; 从左往右偏3px，从下往上偏5px

background-position:10px 10px; 默认相对父元素的padding-box！！！让图片在水平和垂直方向偏移

bg-position:50% 50%; 背景图位于元素中间，百分值是减去背景图片本身的size之后的值，
背景图片的这个点与元素的这个点对齐，同时作用于元素与图像！
当背景图片大于元素时，设置left百分数会向左移动，等同于元素大小减去背景size得出的负数值乘以百分比得到的px为负，所以向左移

图片中心与元素中心保持一致：
background-size:cover;
background-position:center;

css sprite 肯定用px单位（利用元素size,bg-size,bg-p）css精灵 css雪碧
从大图里切小图，把所有图放到一个图片里（一定是png格式），然后用background来切图
一般是用于站点的所有图标（复用程度很广的小图）
优点：节省图片流量/宽带，加快页面加载速度，不用多次加载图片，所有图片一次性出现
缺点：定位麻烦，图片制作麻烦，维护麻烦，图片放大麻烦，定位也随之需要改变

background-origin:content-box; 背景从哪里开始渲染，改变的是bg-position的相对位置！！
content-box ;
padding-box;(默认)
border-box;

background-clip 背景可以覆盖到什么范围，以bg-o为前提
content-box ;
padding-box;
border-box;(默认)，背景颜色默认渲染到border下面
text; 切出文字，-webkit-background-clip:text;（暂时只有谷歌浏览器支持），
设置为文字的背景，需要把文字颜色改为透明
注：当设置bg-clip为border-box时，因为bg-o默认从padding-box显示，所以不会有效果！

background-attachment 图片与内容的关联
关键字：
fixed 背景图片位置在整个视口固定，不随着内容滚动而滚动，文字滚动背景不动
且用background-size时，取的尺寸是视口的尺寸
螺旋效果：http://meyerweb.com/eric/css/edge/complexspiral/glassy.html

scroll 内容部分滚动条文字不随着背景一起滚动
local 内容部分滚动条文字随着背景一起滚动，类似于文字本身就嵌在背景里的效果，但是在页面上滚动条的不随滚动条滚动

opacity:1；不透明度为1，即不透明

2017年4月10日 14:18

浮动元素同时处于流内和流外

浮动元素周围的外边距不会合并

浮动和定位一起用
position:relative时，都生效
其他，position生效

浮动元素自动会生成一个块级框，所以不需要声明display:block

浮动元素的边界为margin边界

浮动元素摆放规则：
1.完全不与其他浮动元素重叠
2.尽量往上
在无法处于上方时，会下移
下移后不能因需要尽量往上（即使有空间）再重新上移
3.尽可能往浮动方向移动
4.上左右方向尽可能不超出包含块
当宽度大于包含块的宽度时，会超其包含块的左右
5.高度不能超过前面所有浮动元素的最高点
6.高度尽可能不超过所在行的行框最高点
7.左浮动块的高度不能超过之前所有左浮动块的高度
8.当父包含块也设置浮动时，父元素即可正好包含其浮动的子元素
9.行内元素会绕着浮动元素渲染
会在左浮动元素的右方、右浮动元素的左方渲染
块级元素会当浮动元素不存在（不考虑clear）
所以仅有浮动元素的包含块默认高度是塌陷的
行内元素会绕着浮动元素渲染

清除浮动
clear只用于块级元素！！！

被设置clear属性block元素会往下移动至其border-box的上边缘与某方向上的浮动元素margin-box的下边缘在同一高度
此现象不是margin collapse
对于左浮动元素来说，如果其右边也是左浮动元素，为其清除右方浮动将是无效的，因此clear:left相当于clear:both
clear:left让我的左方没有浮动，右方的浮动还是会随着它本身一起移动
clear:both让我的左方没有浮动，右方没有浮动

闭合浮动的方案: enclosing float ，使包含块变高以其能够将其浮动后代包在其盒子范围内
1、让父元素float:left;触发bfc（不常用，会导致父元素的父元素也需要浮动）
2、即让常规流的元素把包含块撑大，给包含块增加一个处于最后的块级子节点，为此子节点清除浮动，因此该子节点会下移，同时撑高了包含块
具体方法：放一个真实的节点；
增加父元素的块状伪元素，让块元素撑大父元素；

```
.clearfix::after{
  display:block;
  clear:both;
  content:"";
}
```

3、触发包含块的BFC
父元素增加 overflow:hidden
BFC,block format context，块级格式化上下文

其内部元素一定会渲染在矩形区域内，不会跑到元素外面，如闭合浮动时

此上下文的渲染不会影响其外部，反之亦然
子元素的margin不会超出BFC元素的外面，
比如p的margin会超到父元素外，可以在父元素上增加即可
利用BFC可以让浮动元素的兄弟div元素不环绕浮动元素，div会独立

4、增加父元素的 display:table;
因为表的布局也是个独立上下文，类似BFC
5、用inline-block触发bfc
6、display:flow-root 重新启动渲染，最新属性，需要考虑兼容性，IE不支持

overflow
scroll 默认显示滚动条
auto 内容有多的时候才显示滚动条
hidden 隐藏多的内容，
自己作为包含块且高度确定时才起作用
否则高度不确定时，元素还是被内容撑大，触发BFC

overflow:hidden
当子元素position为fix时，父元素overflow:hidden失效
解决办法：clip:rect(auto, auto, auto,auto) ，将其剪切到元素的内部边界边缘，auto意为裁剪到内部边界
，并且父元素需要绝对定位 只适用于 position:absolute or position:fixed.的元素
裁剪 上 右 下 左
https://stackoverflow.com/questions/12463658/parent-child-with-position-fixed-parent-overflowhidden-bug

clip/clip-path
clip 裁剪
rect(top,right,bottom,left)
rect(0,auto,auto,0) 指定裁剪区域内不做修改

clip-path 多边形裁剪
polygon(.....)
circle
rect

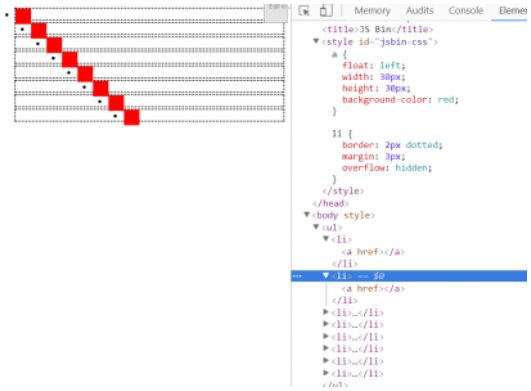
visibility
visible 默认值 可见
hidden 不显示，占位，会影响布局，支持动画，让一个元素延迟消失或出现，transition-delay:1s;transition-property:visibility
与display:none区别： 不显示，不占位，脱离文档流，不影响布局，不支持动画 ，不完全消失（<input type="text" required>时，不显示会一直提交不了）
collapse 用于非表元素时，等同于hidden

z-index
应用于定位元素，默认值auto
z-index:-999
把块放到最底层，必须在有定位的元素上才生效

设置定位或z-index的元素，相当于在一张纸上渲染再重叠，因此不会被覆盖
如九宫格中设置position:relative就可解决hover时，border只出来前两条的问题

```
li:hover{
    border-color:red;
    position:relative;
}
```

例题



解释如下代码渲染结果的成因：https://jsbin.com/dimaxip

li有高度是因为列表标号，是块级元素，当浮动元素不存在，所以li没有被a撑开

浮动元素要尽量往上但不能高出其父元素

阶梯的成因

浮动元素要往左，由于前面一个浮动元素跑到后一个li内部，导致此li内部的浮动元素最多只能往左到与前一个浮动元素挨着

table

2017年4月12日 8:53

布局层次：
单元格<-- 行<-- 行组<-- 列<-- 列祖<-- 表

display:table

colspan
rowspan
注：跨列可以超出，跨行超出无效

border-collapse 表单元格边框合并
separate 默认值，此时只有td可以设置边框样式
collapse 合并

border-spacing 边框间隔，与border-collapse不能合用
border-spacing(x,y)

empty-cells 空单元格，应用于display为table-cell的元素，即td,th
show 默认值
hide 隐藏

回车、换行、tab、空格也当做空单元格，只是设置没有实质内容的td/th
注意与 :empty伪类区分，此选择器只能选择没有任何内容（包括空格、回车）的元素

table-layout 表格大小
auto 自动，由内容撑大
fixed 固定，即设置宽高

col不接受伪类

注：table下结构为thead/tbody，tbody>tr>td，用子元素选择器时，table>tr是无效的，要table>tbody>tr

caption-side
top
bottom

表布局中边框合并的原则（p362）
hidden 优先级最高
none 优先级最低
看颜色、样式、来源

other

2017年4月12日 22:07

可保持状态
tabindex=-1 ，使div的focus可以捕获
input checkbox
input radio
:target选择器

outline 外框
类似border，focus时会出现，但是只能整个设置，如outline:1px solid red;
不影响布局
框住元素的轮廓
可以不为矩形
outline-offset:5px; 离元素的距离
注：border-radius不影响它，只影响元素

pointer-events:none;
不与鼠标交互，即鼠标点击的时候作为不存在。支持性还不是很好。

cursor:pointer;
鼠标移动到某个元素上，显示成手的形状
cursor:not-allowed;鼠标变成禁止形状

user-select:none; 不允许用户选中文字，当前只有webkit支持
none
drag

```
<lablel>
<input type='checkbox'>
<span role='checkbox'  aria-checked='false'></span>
</label>
```

hover抖动问题

```
span:hover{
  display: none;
}
```

原因：在span上hover时，span消失，hover动作就失效，一失效，span又显示，这样重复导致抖动的效果
解决办法：不要在具有位移或消失显见的元素上做hover动作，可以hover到父元素，然后让子元素动作！！

为什么要在文件的最后一行加一个回车

git diff信息更清楚，diff只有一个即你新增的内容，否则回车也将会是一个diff
合并文件更小的出错可能
更容易光标移到文件末尾
行业通用规范，一些软件会在没有回车结尾时给出提示

各种语言下的Unicode转化：

css	\00AB
html	&#56 或 &#x00AB
js	\U4e2d

transform 二维变化

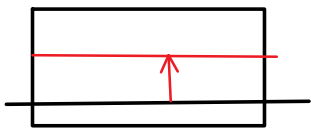
只适用于块级元素

scale(2) 放大2倍，像素级别的放大，zoom只是网页放大

scaleY(-1) 块级元素垂直方向倒置

rotate(90) 顺时针旋转90度

translate(x,y) 平移，y向下



布局总结

2017年4月13日 10:36

常规流/正常流

1. 所有没有浮动，没有定位的元素，就处于常规流
2.所有元素/内容从上到下，从左到右显示，前面的内容会把后面的内容往后顶，后面的元素的位置会被前面的元素往后顶
3.正常流里的元素不会与其他正常流里的元素产生重叠（不使用负margin）

脱离正常流

一个元素完全不影响后面元素的布局

包含块 containing block

1. 一个元素的布局很大程度上依赖于其所在的包含块
2.大多数时候百分比都是取自包含块content-box的宽度或高度
3.如何确定包含块？
*常规流
一个元素的包含块就是离其最近的块级祖先
*浮动元素
同常规流
*定位元素
fixed 视口
relative 相当于元素在正常流
absolute 离其最近的定位祖先
sticky 不在sticky状态时同常规流。在sticky状态时，定位同fixed,但原位置保留，依然影响布局
*根元素
其包含块是初始包含块，即视口

布局/格式化/上下文

什么是布局？

在何种情况下，何种元素将以何种方式摆放在何种位置，以及对其他元素的影响

格式化分类

块级格式化

*水平方向
*块级元素会在其前后换行，使其独占一整行
*块级元素水平方向的尺寸一定会等于其保护块的可用宽度 即（marginleft,bl,pl,w,pr,br,mr相加为available width of containing block）
*margin 与width都可以为auto
width会尽量宽
margin会尽可能的平均分配
除非元素过分受限
过分受限时，某一方（取决于语言环境）的margin会被重置为auto
*垂直方向
*块级元素会垂直摆放（stacking）
*常规流里块级元素的高度等于（当元素无padding和margin时）
最高元素的border上边缘到最低元素border的下边缘
*常规流里块级元素垂直方向的margin会合并
*给常规流里的块级元素设置百分比高度往往是无效的
除非他的包含块的高度不依赖于它
具体原因，是会产生逻辑矛盾，子元素会撑大父元素

行内格式化

就是一个块级元素的每一行内容都是如何布局和生成的
水平方向
由浏览器来计算哪些元素会放在同一行里，然后在行内元素的相应位置剪断，然后像块级元素一样堆叠摆放
行内元素水平方向上的m,p,b,w都会影响布局，垂直方向无效
垂直方向
行内非替换元素（span）
em框，字符框，高度即font-size，对于英文高度不确定，中文一般都一样
值得注意的一点：文字的内容有可能超出em框
内容区/框

行内元素：em框的集合即为内容区
非行内元素：marginbox
行内框：
非替换元素
内容区的上下分别加上半行间距（行间距：line-height 与 font-size的差）
替换元素
就是其margin-box
更严格来说，其行内框是从margin-top的边缘到margin-bottom的边缘
所以有可能一个行内框的高度为负（margin-top为1负值绝对值大于margin-bottom负值绝对值）
行框
能够包含该行所有行内框的最小矩形
是由该行所在行的行内框最高点
以及该行所在行内框的最低点确定
行内框在垂直方向的位置由vertical-align确定

浮动格式化
浮动元素会自动变成block
浮动不能与除position: static/relative;同时使用
浮动元素的摆放
浮动元素会完全不与其它浮动元素重叠
浮动元素会尽量往上
在无法处于上方时，会下移
下移后不能因需要尽量往上（即使有空间）再重新上移
浮动元素会尽量往浮动方向移动
浮动元素会在上，左，右方不超出包含块
当浮动元素的宽度大于包含块的宽度时，会超其包含块的左右
浮动元素的高度也不能超过其前面的任意浮动元素
浮动元素的高度还不能超其所在行行框的最高点
块级元素会当浮动元素不存在（不考虑clear）
所以仅有浮动元素的包含块默认高度是塌陷的
行内元素会绕着浮动元素渲染
会在左浮动元素的右方、右浮动元素的左方渲染
清除浮动 被设置了clear属性的block元素会往下移动直至其border-box的上边缘与某* 方向上的浮动元素margin-box的下边缘在同一高度
此现象不是margin collapse
对于左浮动元素来说，为其清除右方浮动将是无效的
可以理解为clear：left；表示让元素的左方没有左浮动元素
可以理解为clear：right；表示让元素的右方没有右浮动元素
可以理解为clear：left；表示让元素的左方没有左浮动元素、右方没有 浮动元素
闭合浮动 enclosing float
使包含块变高以使其能够将其浮动后代包在其盒子范围内
思路：
1 让常规流里的元素把包含块撑大
给包含块一个处于最后的块级子结点
为此子结点清除浮动
此结点则会下移，同时撑高了包含块
放一个真实的结点
使用伪元素
2 触发包含块的BFC
使用任何一种方法触发包含块的BFC
overflow：visibale
BFC?
Block Format Context 块级格式化上下文
其内部的元素一定渲染在一个矩形区域内
此上下文的渲染不会影响其外部，反之亦然
子元素的margin不会超BFC元素的外面

定位
不同定位方式
fixed 相对于 视口

relative 同不定位
absolute 最近的定位祖先
定位盒子（ absolute ）
被定位元素的定位盒子是其marginbox
祖先的定位是其paddingbox

水平垂直居中
行内（影响行内垂直方向高度，即行内框，主要是vertical-align ）
父元素
line-height等于父元素的高度
text-align:center;
inline-block;
绝对定位
全部写死
绝对定位
里面的元素绝对定位上下左右都为0
margin为auto
指定宽高
要求比包含块小，否则只垂直居中，左边与父元素对齐
用table-cell，缺点，元素不能超出
父元素
display:table-cell
vertical-align:middle；单元格的垂直居中就是垂直居中
text-align:center;
子元素
inline-block
或
block，margin:auto
用transform，在任何比例下都能相对于父元素居中，但是到ie9才支持
transform 二维变化
scale(2) 放大2倍，像素级别的放大，zoom只是网页放大
scaleY(-1) 块级元素垂直方向倒置
rotate(90) 顺时针旋转90度
translate(x,y) 平移，y向下
具体如：

```
section{
  height:500px;
  background-color: pink;
  position:relative;
}
div{
  background-color: red;
  position:absolute;
  top:50%;
  left:50%;
  transform:translate(-50%,-50%);
}
```

用flex,父元素(已知宽高)display:flex
子元素margin:auto，此时margin对水平垂直都有效

图片水平垂直居中

```
.box{line-height:300px;text-align:center}
.box>img{vertical-align:middle;}
```

或

```
div{
  background-color: red;
  border:10px solid;
```

```
  display:table-cell;
  text-align:center;
  vertical-align:middle;
  width: 200px;
  height: 200px;
}
img{
  width:100px;
  display:inline-block;
  vertical-align:middle;
}
```

多行文本水平垂直居中

```
.box{line-height:250px;text-align:center}
.box>.text{display:inline-block;line-height:normal;text-align:left;vertical-align:middle}
```

--多行文字水平垂直居中实现的原理跟图片是一样的，区别在于要把多行文本所在的容器的display水平转换成和图片一样，也就是inline-block，以及重置外部继承的text-align 和 line-height属性值

RePaint 、 Reset.css

2017年4月18日 15:03

RePaint 重绘，重新把一个元素画一遍，如修改颜色，不需要调整布局
当元素不影响布局的属性发生变化的时候，浏览器会重新绘制该元素，而不去重新计算其他（包含发生变化的元素）的布局，速度很快

ReLayout 回流，需要重新调整布局
当某一个或多个元素的布局相关的css属性发生变化的时候，会影响其后面的元素的布局，浏览器需要【重新】计算所有受影响的元素的布局，此过程即为relayout
特点：浏览器的计算过程会比较久，回流一定会造成重绘
解决方案：尽量减少回流的次数：动画或其他交互尽量不要影响布局，使用绝对定位或2d变幻等 ，如只需要一个边框时，可以把其余边框改成透明色，这样既避免了回流

transition-property:all;造成的问题
所有的属性都会产生渐变，有可能会引起频繁的回流，造成页面卡顿
同时如果元素的display也发生变化的话，可能会让本来有的动画的属性变成没有
在chrome里，transition-property:all不会使z-index属性发生渐变

reset.css
重置 user agent 样式,
如list-style需要为none，可以在该样式表里写
把各个元素的样式设置为开发者想要的

normalize.css
调整浏览器之间差异的样式，让所有浏览器的默认效果尽量接近，保留有用的 user agent 样式，同时进行一些 bug 的修复
把个别浏览器里跟其他浏览器不同的细节调成相同

来自〈https://www.zhihu.com/question/20094066〉

reset.css

```
ol, ul {
list-style: none;
}

:link, :visited , ins {
text-decoration: none;
}
```

```
blockquote, q {
quotes: none;
}

blockquote:before, blockquote:after,
q:before, q:after {
content: ” ;
content: none;
}

table {
border-collapse: collapse;
border-spacing: 0;
}

:focus {
outline: 0;
}

body, div, dl, dt, dd, ul, ol, li,
h1, h2, h3, h4, h5, h6, pre, code,
form, fieldset, legend, input, button,
textarea, p, blockquote, th, td {
margin: 0;
padding: 0;
}
```

来自 <http://shawphy.com/2009/03/my-own-reset-css.html>

layout.css

闭合浮动

```
.clearfix:after {
content: “.” ;
display: block;
height: 0;
clear: both;
visibility: hidden;
}
.clearfix {display: inline-block;}
```

如

```
<section class="clearfix"><div ></div></section>
body *{
  background-color: rgba(0,0,0,0.1);
}
section{
  border:1px solid;
}
div{
    width: 10px;
  height: 10px;
  border:1px solid;
  float:left;
}
```

来自 <http://shawphy.com/2009/03/my-own-reset-css.html>

normalize.css

```
[hidden] {
        display: none;
        }

legend
{
        box-sizing: border-box; /* 1 */
        color: inherit; /* 2 */
        display: table; /* 1 */
        max-width: 100%; /* 1 */
        padding: 0; /* 3 */
        white-space: normal; /* 1 */
        }

/**
                * Remove the inner border and padding in Firefox.
                */
    button::-moz-focus-inner,
```

```
[type="button"]::-moz-focus-inner,
[type="reset"]::-moz-focus-inner,
[type="submit"]::-moz-focus-inner {
border-style: none;
padding: 0;
}

/**
 * Remove the margin in Firefox and Safari.
 */

button,
input,
optgroup,
select,
textarea {
margin: 0;
}
```

来自 <https://github.com/necolas/normalize.css/blob/master/normalize.css>

2017年4月20日　　18:46

list-style-type

只应用于display值为list-item的元素

disc 实心圆，默认值

decimal-leading-zero 如00 01 02...

cjk-ideographic　　中日韩数字

list-style-image url()

无法改变图片大小、无法跟文字对齐、无法调节图片位置

list-style-position

inside　影响行内布局

outside

content 生成内容无法被选中

string

url()

attr(href)

counter（计数器，list-style(可省略))

counters（计数器，"."）取到当前级的所有计数器

（计数器，".",list-style)

quotes 生成引号

quotes:'"' '"'； 成对写

如果引号的嵌套层次大于已经定义的引号对数，最后一对引号将重用与更深层次的嵌套

open-quote

close-quote

no-open-quote

no-close-quote　　闭合跟他对应的前括号，只是不显示了

相关：

<blockquote> 段落引用

<q> 行内引用

如：

```
h1{
   quotes:"《" "》"
}
h1:first-child:before{
  content:open-quote
```

```
}
h1:first-child:after{
  content:close-quote
}
```

计数器

counter-reset 计数器的起点

如 counter-reset:mycounter 1; 不写数值默认为0

counter-increment 递增一定的量，默认值为1

如counter-increment:mycounter;

同个元素上重置又递增则直接取递增值

计数器由同一个元素递增和使用时，递增也发生在计数器显示之前

作用域：只在他的上一级父元素内起效

多层嵌套的列表可以用counters()

如

```
ol{
        counter-reset:ordered;
}
ol li:before{
        counter-increment:ordered;
        content:counters(ordered,'.') "-";
}
```

2017年5月9日 19:49

1、simsun字体(宋体)下：
font-size + 行间距 = line-height

因此：
行间距= line-height - font-size

2、line-height:1.5;
可以根据当前元素的font-size大小计算
即line-height= 1.5 * font-size

3、line-height:150%
相对于设置了该line-height属性的元素的font-size大小计算

4、区别： line-height:1.5;与 line-height:150%（1.5em）
前者：所有可继承元素(子元素)根据font-size重新计算行高（全局使用比较好）
后者：当前元素根据font-size计算行高，继承给下面的元素、不重新计算

5、图片水平垂直居中
.box{line-height:300px;text-align:center}
.box>img{vertical-align:middle;}

6、多行文本水平垂直居中
.box{line-height:250px;text-align:center}
.box>.text{display:inline-block;line-height:normal;text-align:left;vertical-align:middle}

--多行文字水平垂直居中实现的原理跟图片是一样的，区别在于要把多行文本所在的容器的display水平转换成和图片一样，
也就是inline-block，以及重置外部继承的text-align 和 line-height属性值

7、doucment对象是window对象的一部分，window可省略
document.body==window.document.body //true
window.location.href==document.location.href==location.href //true

8.clientWidth、clientHeight 元素的可视部分宽度和高度!
clientWidth=style.width+style.padding*2-滚动条宽度

9、clientLeft、clentTop 元素周围边框的厚度
读取元素的border的宽度和高度
clientTop=border-top的border-width
clientLeft=border-left的border-width

10、offsetWidth offsetHeight
offsetWidth=style.width+style.padding*2+border*2
或
offsetWidth=clientWidth+滚动条宽度+border*2

11、offsetLeft、offsetTop
offsetParent定义：
如果当前元素的父级元素没有进行css定位（position），
offsetParent为body
如果当前元素的父级元素中有css定位，offsetParent取最近的那个父级元素

IE6/7中：

offsetLeft=(offsetParent的padding-left)
+（当前元素的margin-left）

IE8/9/10及Chrome:
offsetLeft=(offsetParent的margin-left)
+(offsetParent的border宽度)
+(offsetParent的padding-left)
+(当前元素的margin-left)

FireFox:
offsetLeft=(offsetParent的margin-left)
+(offsetParent的padding-left)
+(当前元素的margin-left)

12、scrollWidth、scrollHeight (body与div公式不一样)
body:
一、给定宽度小于浏览器窗口
scrollWidth通常是浏览器窗口的宽度
scrollHeight通常是浏览器窗口的高度

二、给定宽高大于浏览器窗口，且内容小于给定宽高
scrollWidth=给定的宽度+其所有的padding*2、margin*2、border*2
scorllHeight=给定的高度+其所有的padding*2、margin*2、border*2

三、给定宽高大于浏览器窗口，且内容大于给定宽高
scrollWidth=内容宽度+padding+margin+border
scrollHeight=内容高度+pading+margin+border

div:
无滚动轴：
scrollWidth=clientWidth=style.width+sytle.padding*2

有滚动轴：
scrollWidth=实际内容的宽度+padding*2
scrollHeight=实际内容的高度+padding*2

13、scrollLeft、scrollTop(可写，当元素其中的内容超出其宽高的时候，元素被卷起的高度和宽度（即被挡住部分））

14、clientX 事件属性返回当事件被触发时鼠标指针向对于浏览器页面（或客户区）的水平坐标
pageX 属性是鼠标指针的位置，相对于文档的左边缘。

15、display:inline-block;//应用此特性的元素呈现为内联对象，周围元素保持在同一行，但可以设置宽度和高度地块元素的属性,(可以让容器正好等于元素的宽和高)
display:inline; //就是将元素显示为行内元素,和其他元素都在一行上；高，行高及顶和底边距不可改变
display:block; //就是将元素显示为块级元素,总是在新行上开始；高度，行高以及顶和底边距都可控制；宽度缺省是它的容器的100%，除非设定一个宽度

16、绝对定位居中
position:absolute;
margin:auto;
left:0;
right:0;

17、普通元素的百分比margin都是相对于容器的宽度计算的！！
例如一个容器200*100
现在设置里面的图片元素img{margin:10%}
则左边、上边的实际距离都为20！！

绝对定位元素的百分比margin是相对于第一个定位祖先元素的宽度计算的
例如img{margin:10%；position:absolute}
若第一个定位祖先元素为1024*200，则左边、上边的实际距离都为102.4！！

18、overflow 默认值为visible
如果overflow-x与overflow-y设置的值相同，则等于与overflow,例如 overflow-x：auto,overflow-y:auto 相当于 overflow:auto
如果overflow-x与overflow-y值不同，且一个为visible,另外一个为auto/hidden/scroll等，则visible会被重置为auto!!

作用条件：
1、非display:inline;
2、对应方位的尺寸限制。width/height/max-width/max-height/absolute
3、对于单元格td等，需要table为table-layout:fixed状态才行！！

技巧：
IE7下，文字越多，按钮两侧padding留白就越大！解决方案 给按钮添加css样式overflow:visible