CSS

## CSS样式的引入方式

**内联（行内、行间）样式**：在HTML标签上添加**style属性**来实现

**内部样式：**在**<style>标签**内添加样式

注：内部样式的优点，可以复用代码

区别：内部样式的代码可以复用，符合W3C规范标准，进行让结构和样式分开处理。

**外部样式：**引入一个单独的css文件，name.css（空格的形式不进行限制）

引入形式：

<link>标签，rel属性指定资源和页面的关系，href属性指定资源的地址

@import：有很多问题，不建议使用

区别：<https://www.cnblogs.com/my--sunshine/p/6872224.html>

id前面是要加一个**#**

class前面要加一个**.**

## CSS颜色表示法

1、单词表示法：Blue，white等

2、十六进制表示法：

3、RGB三原色表示法：rgb(255,255,255)取值范围：0-255

## CSS背景样式

**1、Background-color**：背景颜色

**2、Background-image**：背景图片，默认会水平垂直都铺满背景图

**3、Background-repeat**背景图片的平铺方式。

Repeat默认x,y都平铺；repeat-x(y)代表x(y)轴平铺；no-repeat代表不平铺

**4、Background-position**：背景图片的位置。Number（数字或百分比）|英文

**正数：向下和右；负数：向上和左**

X轴：left，center，right

Y轴：top，center，bottom

**5、Background-attachment**：背景图随滚动条的移动方式

背景图片的附着点有三类，分别是浏览器的可视区域、背景容器本身区域和背景容器的内容区域。具体来说，background-attachment的值也就有三种，即：

**scroll：默认值，**背景图片附着在背景容器上，它是相对于背景容器（元素）固定，它会随着背景容器的滚动，而不是随着它的内容滚动（也可以看成是对元素边框固定）。

**fixed：** 背景图片附着在浏览器的可视区域。因为浏览器的可视区域不具备滚动的性格，所以它是不滚动的。（背景位置按照浏览器（若设置为50%，则代表是相对于浏览器的50%）进行偏移）

**local：**这是CSS3新增的属性值。背景图片附着在背景容器的内容区域。它会随着内容的滚动而滚动。

<https://blog.csdn.net/weixin_39256994/article/details/78671202>

<https://www.cnblogs.com/starof/p/4511367.html>

## CSS边框样式

Border-style:边框样式。solid实线，dashed虚线，dotted点线

Border-width:边框大小

Border-color:边框颜色 注：transparent是透明色

注：针对某一条边进行单独设置在border和style/width/color之间加上 -left/right

## CSS文字样式

### font-family

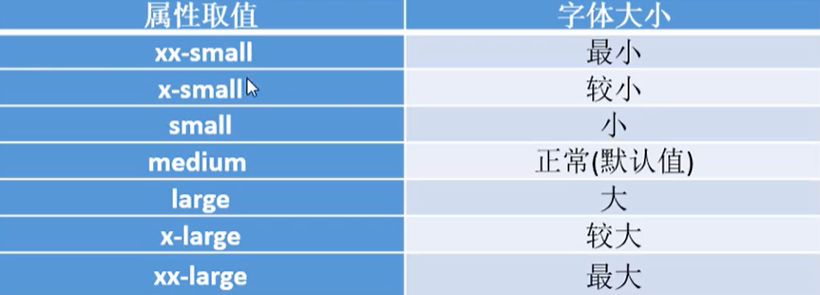
字体类型，字体中出现空格必须加引号

衬线体与非衬线体：<https://www.jianshu.com/p/414ea6c05276>

注意事项：1、设置多字体方式

### font-size

字体大小，默认大小为16px，不推荐设为奇数的单位



**不推荐用上图的表示方法**

### font-weight字体粗细

正常（normal）加粗（bold）|数值表示（100-500正常，600-900加粗）

### font-style字体样式

正常（normal）斜体（italic）注oblique也是斜体，但用得少

区别：italic所有带倾斜字体的可以设置，Oblique没有倾斜属性的字体也可以设为倾斜

Color对字体进行颜色设置

## CSS段落样式

### text-decoration：

文本修饰，下划线（underline），删除线（line-through），上划线（overline），无修饰（none）。可以添加多个修饰，空格隔开

### text-transform：

文本大小写，针对英文段落。小写（lowercase）大写（uppercase）只针对首字母大写（capitalize）

### text-indent：

文本缩进

针对首行缩进，单位px/em

em单位：相对单位（推荐）

### text-align：

文本对齐方式，默认左对齐，对齐方式：left，center，right，justify（两端点对齐）

### line-height：行高

默认行高不是固定值，根据当前字体大小变化

取值：1、number（px）| scale（比例值，和文字大小比较，无单位）

### 强制折行（针对英文）：

**letter-spacing**：字符之间的间距

**word-spacing：**词之间的间距（针对英文段落）

1. Word-break：break-all（非常强烈的折行）
2. Word-wrap：break-Word（不那么墙裂的折行，会产生留白）

## CSS复合样式

复合的写法是通过空格的方式实现的，有的不需要关心顺序，例如background，border；有的是需要顺序的，比如font（最少有两个值，size/line-height和family，顺序固定）

复合样式尽量不要混写，若非要混写，那么一定要先写复合样式再写单一样式，因为若先单一后复合，复合包含了单一样式，会覆盖单一的设置

## CSS选择器

### Id选择器

Css：#elem{}

Html:id=”elem”



注：

1. 在一个页面中是唯一值，只能出现一次
2. 命名规范，字母，\_,-,和数字，且第一个不能是数字
3. 命名方式：驼峰式，下划线式，短线式
4. div#名字+tab快捷创建<div id=”名字”></div>

### class选择器

CSS: .elem{}

Html:class=”elem1 elem2 …”

注：

1. class选择器可以复用
2. 可以添加多个class样式
3. 多个样式时，样式的**优先级根据CSS决定**，而不是class属性中的顺序
4. 标签+类的写法（.elem会对所有class为elem的标签进行css样式修饰，而**div.elem**会对所有class为elem的**div标签**进行css样式修饰）

### 标签（元素）选择器

用法：1、去掉标签的默认样式2、复杂的选择器中如层次选择器中

### 群组选择器

通过逗号的方式给多个不同的选择器添加统一CSS样式

css：如div,#text,.title{background:red}

### 通配选择器

\*{} -> div,p,ul,li…{}尽量避免使用通配选择器，因为会给所有的标签添加样式，慎用

使用场景：

1. 去掉所有标签的默认样式

### 层次选择器

后代：M N{}：只有M下的N才具有大括号中的样式

父子：M > N{}：只关心M的孩子，而M孩子的孩子并不关心

兄弟：M ~ N:当前M**下面**的所有兄弟**N标签**

相邻：M+N{}当前M下面的相邻的**N标签**

### 属性选择器



Div[class]：所有包含class属性的都匹配

Div[**class=box**]：（box不被引号引着）

Div[class=search]：只有完全等于search才匹配

Div[class\*=search]：只要出现了search就匹配，不用完全匹配

^=：起始匹配:起始值为search才匹配

$=：结束匹配，末尾值为search才匹配

多属性匹配：div[class][id][]

### 伪类选择器

CSS伪类用于向某些元素添加特殊效果。一般用于初始样式添加不上的时候，用伪类来添加。

写法： M:伪类{}

**:link** 访问前的样式（只能给a标签）

**:visited** 访问后的样式（只能给a标签）

**:hover** 鼠标移入时的样式（可以给所有标签）

**:active** 鼠标按下时的样式（可以给所有标签）

注：

1. 如果四个伪类都生效，要注意顺序：L V H A
2. 一般网站只这样设置：a{} a:hover{}

**:after :before**  通过伪类的方式给元素添加一个文本内容

例：div:after{content:”添加内容”;color:red;}

:after可以用来清除浮动

**:checked :disabled :focus**都是针对表单元素

用法：:checked/:disabled/:focus{}

结构性伪类选择器：

nth-of-type nth-child

用法举例：li nth-of-type(number | n值【可以为kn用来设置隔行换色】){}第几个（）li标签具有{}中属性

同样以上面的例子进行说明：

first-of-type：第一行具有某样式 nth-child

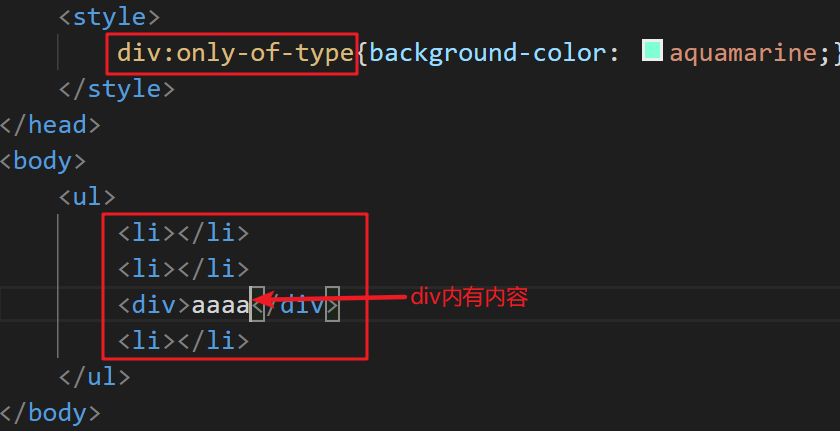
last-of-type：最后一行具有某样式 first-child

only-of-type(只有一个某标签的时候才具有某样式) only-child

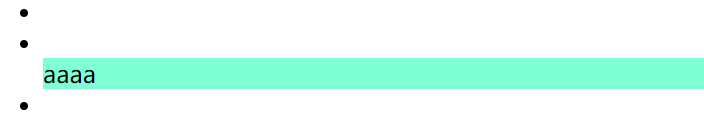
**nth-of-type和nth-child区别（其他同理）**

加li：li:nth-of-type(3)和li:nth-child(3)

前者是第三个li标签会有某属性；后者是第一个li开始后的第三个标签（可以不是li）会有某属性



浏览器效果



## CSS样式继承

文字相关的样式可以被继承，布局相关的样式不可被继承（默认不继承，但是可以设置继承）。示例：p{border:inherit;}

## CSS优先级

**相同样式优先级**：设置**相同样式时，后面的优先级较高**，但不建议出现重复设置样式的情况

**内部样式与外部样式：**优先级相同，若设置了相同样式，那么后写的（在文件后面覆盖了）引入方式优先级高

**单一样式优先级：style行间 > id > class > tag > \* > 继承**

注：

style行间 权重 1000

id 权重 100

class 权重 10

tag 权重 1

**!important：**提升样式优先级，非规范方式，不建议使用，不能针对继承的属性提升优先级

**标签+类与单类：**标签+类的优先级高于单类的优先级

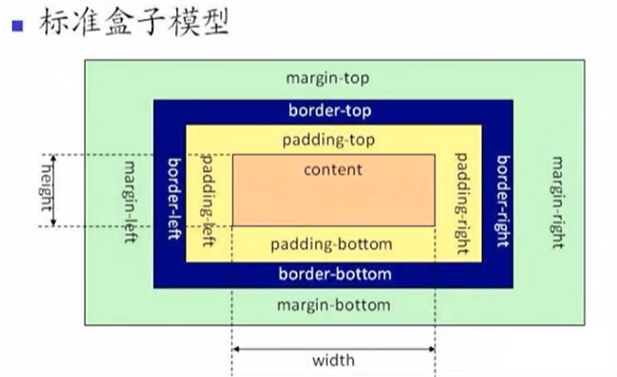
**群组优先级：**群组选择器与单一选择器优先级相同，谁后写谁优先级高

**层次优先级**（建议三层内）：

1. 权重比较：若含多个标签则把所有标签的权重相加进行比较优先级，但是10个tag的权重【1+1+…+1(10个)】< class(10)
2. 约分比较

把等级相同的约掉，比较剩下的权重

## CSS盒子模型



组成：content -> padding -> border -> margin

物品 填充物 包装盒 盒子之间的间距

**content部分：**内容区域，由宽（width）高（height）决定

**padding：**内边距（内填充），内容和边框之间的距离，图中淡黄色

padding-left/right/top/bottom

number：单位px;

写一个值：上下左右；两个值：上下，左右；四个值：上，右，下，左

**margin：**外边距，边框和其他盒子的距离

margin-left/right/top/bottom

number：单位px;

写一个值：上下左右；两个值：上下，左右；四个值：上，右，下，左

注：

1. 背景颜色会填充到margin以内的区域（不包括margin区域）
2. 文字会在content区域
3. padding不能出现负值，margin是可以出现负值

### box-sizing

box-sizing属性允许您以特定的方式定义匹配某个区域的特定元素，取值为content-box（默认值）| border-box。

理解：盒尺寸，可以改变盒子模型的展示形态

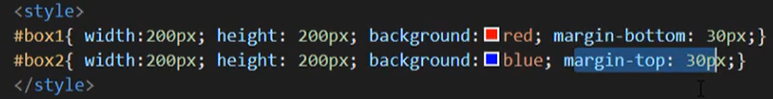
默认值：content-box，宽高对content起作用；而border-box的宽高对content和padding和border都起作用

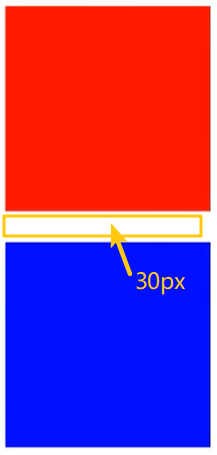
使用场景：

1. 不必计算很多值
2. 解决一些百分比问题（例添加100%的输入框但想加入30px的padding，若直接加会出现滚动条，可以使用border-box）

### 存在问题

**margin叠加：**当给两个盒子同时加上下外边距的时候，就会出现叠加的问题，且这个问题只在上下有，左右不存在这个叠加问题。二者之间的margin值会取较大的作为叠加的值





**Margin问题的解决方案：**

1. BFC规范
2. 想办法只给一个元素添加间距

**Margin传递问题：**

该问题只出现在**嵌套的结构**中，且只有margin-top有该问题，其他方向无。问题描述，对于嵌套的div，若设置里面的div的margin-top会使两个div一起向下移动

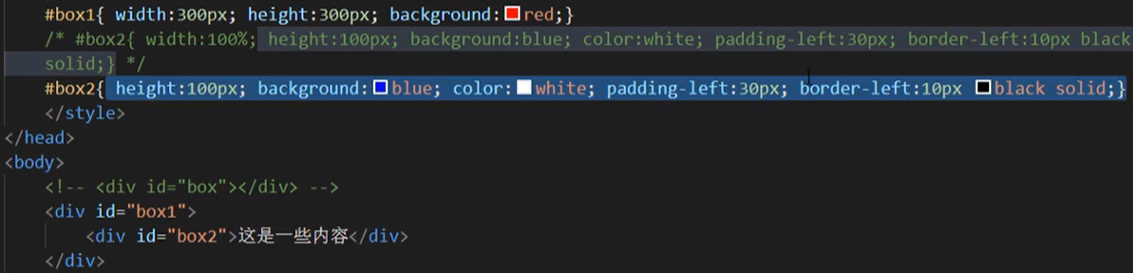
margin传递问题的解决方案：

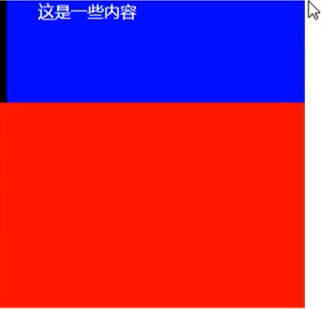
1. BFC规范
2. 给父容器加边框
3. margin换成padding（加给父元素）

扩展：

1、margin-left/right：auto 最左/最右

2、不设置宽高的现象：对盒子模型的影响是会自动计算容器的大小，节省代码





## 标签分类

### 按类型

**block：**块：div,p,ul,li,h1…

特性：

1、独占一行

2、支持所有样式

3、不写宽的时候和父元素的宽相同

4、所占区域是矩形

**lnline：**内联：span,a,em,strong,img…

特性：

1. 挨在一起，左右并排
2. 有些样式不支持，比如宽高，margin，padding（一些方向）
3. 不写宽的时候由内容决定宽度
4. 所占区域不一定是矩形
5. 内联标签之间有空隙，是由换行产生的，若两个内联标签相邻则不会有空隙；若设置body{ font-size:0}，设置内联标签的font-size为n px，也可以消除空隙

注：布局一般用块标签，修饰文本一般用内联 标签

**lnline-block：**内联块：input,select…

特性（具有内联和块的属性）：

1. 挨在一起但是支持宽高
2. 有空隙，所占区域为矩形

### 按内容

Flow：流内容

Metadata：元数据

Sectioning：分区

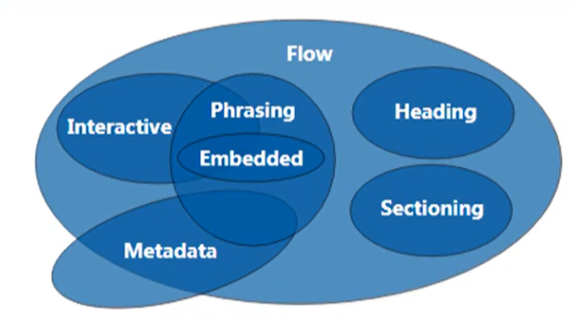
Heading：标题

Phrasing：措辞

Embedded：嵌入的

Interactive：互动的

参考：<https://html.spec.whatwg.org/multipage/dom.html#kinds-of-content>



### 按显示分

替换元素：浏览器根据元素的标签和属性，来决定元素的具体显示内容

示例：img，input…

<img>虽然属于inline但是也属于替换元素，因此支持宽高

非替换元素：将内容直接告诉浏览器，将其显示出来，只写标签没有属性，如h1

示例：div、h1、p…

## 显示框(display)类型

**display：**

display:none;表示此元素将不被显示。

display：block;将元素显示为块元素。（又叫块级元素）

display:inline;将元素显示为内联元素。（又叫行内元素）

display:inline-block;将元素显示为内联块元素。

display:inherit;规定应该从父元素继承 display 属性的值。

注：display:none 与 visibility:hidden 区别

display:none是不占用空间的隐藏，下方的可以占用原来空间

visibility:hidden是占空间的隐藏

## 标签嵌套规范

ul、li；dl、dt、dd；table、tr、td

块标签可以嵌套内联标签，内联不可以嵌套块

特殊：正确的写法

<a href=”#”>

<div></div>

</a>

块标签不一定可以嵌套块标签

错误写法：<p>

<div></div>

</p>

## 溢出隐藏(overflow)

Overflow：

Visible：默认

Hidden：隐藏

Scroll：出现滚轮，多就可滚动，少就不可以

Auto：内容多就出现滚动条，少就不出

x轴、y轴：

overflow-x、overflow-y针对不同轴设置滚轮

## 透明度与手势(opacity,cursor)

opacity：0（透明）-1（不透明）；

注：1、占用空间

2、所有子内容也透明

**3、若想要只对背景设置透明度，使用background:{rgba(num,num,num, [0-1] )}最后一个值为透明度**

**cursor：设置鼠标移入后的手势**

default：默认箭头

实现自定义手势：1、准备图片（.cur/.ico）

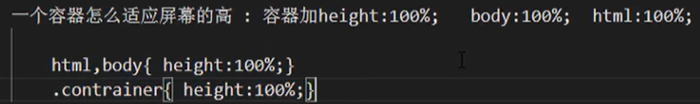
1. cursor:url(图片位置),auto;

## 最大、最小宽高

min-width max-width

min-height max-height div的宽高不定，在某一范围内

注：强化对百分比的理解， 百分比以**父容器**的大小进行换算



## CSS的默认样式

没有默认样式：div、span、…

有默认样式： body、h1…h6、p、ul、…

Body->margin:8px

H1->margin:上下 21.44px,font-weight:bold

p->margin：上下 16px

ul -> margin：上下 16px padding:左 40px

默认点：list-style：disc

a -> text-decoration:underline

## CSS reset

**简单的CSS reset：**

\*{ margin:0; padding:0; }

优点：不用考虑哪些标签会有默认的margin和padding

缺点：稍微的影响性能，改进：body,p,h1,ul{ margin:0; padding:0;}

ul{ list-style:none; }

a{ text-decoration:none; color:#666; }

img{ display:block; }

图片跟容器底部有一些空隙，原因：img是内联标签，内联元素的对其方式是按照文字基线对齐的。改变对齐方式方法1：vertical-align:bottom（底线对齐）方法2：display:block

**重置样式的参考：<https://blog.csdn.net/brain_bo/article/details/81560444>**

写具体页面的时候或者一个布局效果的时候：

1. 写结构
2. Css重置样式
3. 写具体样式

## float浮动

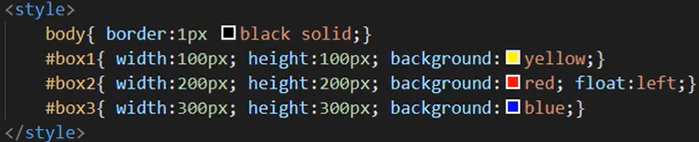
文档流：文档流是文档中可显示对象在排列时所占用的位置。

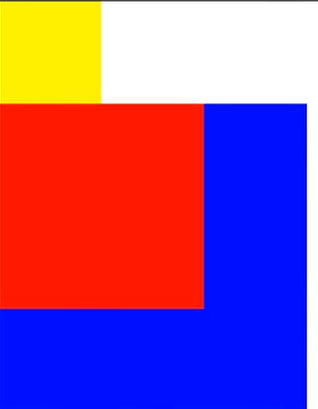
Float浮动特性：加浮动的元素会脱离文档流，会**沿着父容器**靠左或靠右排列，如果之前已经有浮动的元素，会挨着浮动的元素进行排列

取值：left，right，none

**float注意点**

1. 只会影响后面的元素。





2、内容默认提升半层，所以在蓝色div上加内容，内容不会被红色覆盖。

3、默认的width根据内容决定。

4、浮动元素放不下会换行排列。

5、主要给块元素添加，但也可以给内联元素添加。

### 清除浮动（重要）

**上下排列：**使用clear属性清除float浮动clear:left/right/both

**嵌套排列：**

固定宽高 : 不推荐 , 不能把高度固定死，不适合做自适应的效果。

父元素浮动 : 不推荐 , 因为父容器浮动也会影响到后面的元素。

overflow : hidden (BFC规范) , 如果有子元素想溢出，那么会受到影响。

display : inline-block (BFC规范)，不推荐，父容器会影响到后面的元素。

设置空标签 : 不推荐 , 会多添加一个标签。

after伪类 : 推荐，是空标签的加强版，目前各大公司的做法。



注：clear属性只会操作块标签，对内联标签不起作用

## Position定位

### position特性

css position属性用于指定一个元素在文档中的定位方式。top、right、bottom、left 属性则决定了该元素的最终位置。

### position取值

static（默认）

#### relative 相对定位：

1、如果没有定位偏移量，对元素本身没有任何影响

2、不使元素脱离文档流

3、不影响其他元素布局

4、left、top、right、bottom是相对于当前元素自身进行偏移的，正数：向下和右；负数：向上和左

#### absolute 绝对定位

1、使元素完全脱离文档流

2、使内联元素支持宽高 （让内联具备块特性）

3、使块元素默认宽根据内容决定（让块具备内联的特性）

4、如果有定位（**设置了position**）祖先元素，**相对于定位祖先元素发生偏移**，没有定位祖先元素，相对于整个文档发生偏移（绝对、相对、固定）

#### fixed 固定定位

1、使元素完全脱离文档流

2、使内联元素支持宽高 （让内联具备块特性）

3、使块元素默认宽根据内容决定（让块具备内联的特性）

4、**相对于整个浏览器窗口进行偏移**，不受浏览器滚动条的影响

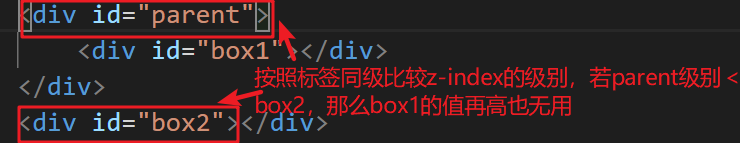
#### sticky 粘性定位

在指定位置进行粘性操作，比如对于设置了position:sticky,top:10px的标签，移动到距离浏览器≤10px的位置时，该标签会保持距离顶部10px而其他标签内容移动

### z-index定位层级

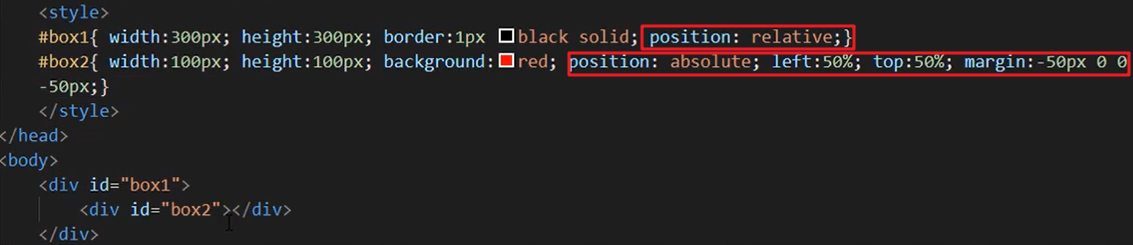
默认层级为0，是为了调整嵌套时候的层级问题（哪个在上面）

1. parent不写position，box1和box2写：后设置样式的优先级高
2. parent不写position，box1和box2写且都写z-index（默认为0），哪个层级高（数大）哪个在前面
3. parent 和box1和box2都写position，z-index，则按照图中说的，按照同级标签比较



### 举例

定位实现元素居中：



## CSS添加省略号

1、width：必须有一个固定的宽

2、white-space : nowrap

不让内容折行

3、overflow : hidden隐藏溢出的内容

4、text-overflow : ellipsis添加省略号

## CSS Sprite

**特性**

CSS雪碧也叫做CSS精灵，是一种网页图片应用处理方式。它允许你将一个页面涉及到的所有零星图片都包含到一张大图中去加载。

**好处**

可以减少图片的质量，网页的图片加载速度快

减少图片的请求的次数，加快网页的打开

## CSS圆角

border-radius：

两个值：左上和右下，右上和左下

四个值：左上，右上，右下，左下

正圆可以用border-radius:50%

border-radius:20(num1)px/40(num2)px 椭圆和矩形相切

## strong与b，em与i

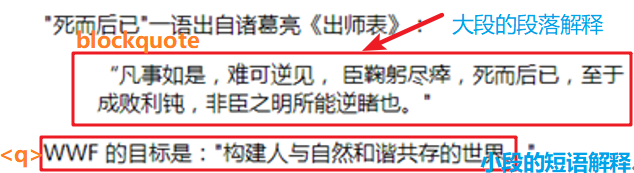
表现形态都是文本加粗和文本斜体，区别在于strong和em是具备语义化的，而b和i不具备语义化。

i，b都是内联标签，可以用来简化选择器操作

## 引用标签

blockquote : 引用大段的段落解释

q : 引用小段的短语解释



abbr : 缩写或首字母缩略词

用法：<abbr title=”全名”>缩写或首字母缩略词</addr>后面的话

address : 引用文档地址信息

cite : 引用著作的标题

语义化标签：便于浏览器找到，

## Iframe嵌套页面

iframe 元素会创建包含另外一个文档的内联框架（即行内框架）。可以引用其他的HTML到当前HTML中显示

|  |  |
| --- | --- |
| **属性** | **含义** |
| **frameborder** | **规定是否显示框架周围的边框** |
| **width** | **定义 iframe 的宽度** |
| **height** | **定义 iframe 的高度** |
| **scrolling** | **规定是否在 iframe 中显示滚动条** |
| **src** | **规定在 iframe 中引入的 URL** |
| **srcdoc(和src同时存在，优先显示)** | **规定在 iframe 中显示的页面内容** |

应用场景：数据传输、共享代码，局部刷新、第三方介入等。

## br与wbr

br 标签表示换行操作，而 wbr 标签表示软换行操作。

提示：如果单词太长，或者您担心浏览器会在错误的位置换行，那么您可以使用 wbr 元素来添加 Word Break Opportunity（单词换行时机）

## pre和code

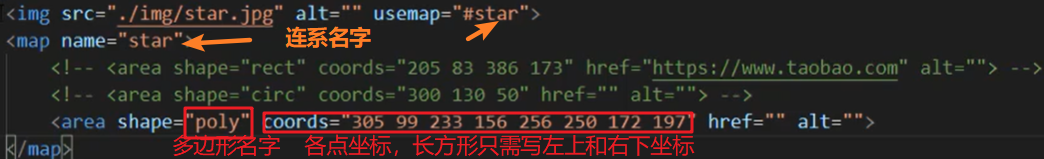
pre 元素可定义预格式化的文本。被包围在 pre 元素中的文本通常会保留空格和换行符。

只应该在表示计算机程序源代码或者其他机器可以阅读的文本内容上使用 code 标签。虽然 code 标签通常只是把文本变成等宽字体，但它暗示着这段文本是源程序代码。

## map与area

定义一个客户端图像映射。图像映射（image-map）指带有可点击区域的一幅图像。area 元素永远嵌套在 map 元素内部。area 元素可定义图像映射中的区域。（给特殊图形添加链接，area能添加的热区的形状，矩形、圆形、多边形）

area 元素的href属性定义区域的URL，shape属性来定义区域的形状，coords属性定义热区的坐标。



## embed与object

embed和object都表示能够嵌入一些多媒体，如flash动画、插件等。基本使用没有太多区别，主要是为了兼容不同的浏览器而已。

object 元素需要配合param 元素一起完成。

用得少，移动端不支持flash

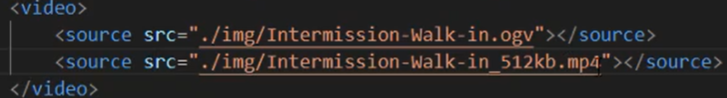
## audio和video

audio标签表示嵌入音频文件，video标签表示嵌入视频文件。默认控件是不显示的，可通过controls属性来显示控件。

用法：<audio src=”音频地址” controls></audio>

loop:循环播放 autplay自动播放

为了能够支持多个备选文件的兼容支持，可以配合source标签做备选功能。



## 文字注解和文字方向

ruby 标签定义 ruby 注释（中文注音或字符），rt 标签定义字符（中文注音或字符）的解释或发音。

bdo 标签可覆盖默认的文本方向。



## 扩展link标签

<link rel="stylesheet" type="text/css" href="theme.css">

<link rel="icon" type="/image/x-icon" href="http://www.mobiletrain.org/favicon.ico">

<link rel="dns-prefetch" href="//static.360buyimg.com">

## meta标签扩展

辅助信息

<https://www.jianshu.com/p/850d2a209ba8>

## HTML5新语义化标签

header : 页眉

footer : 页脚

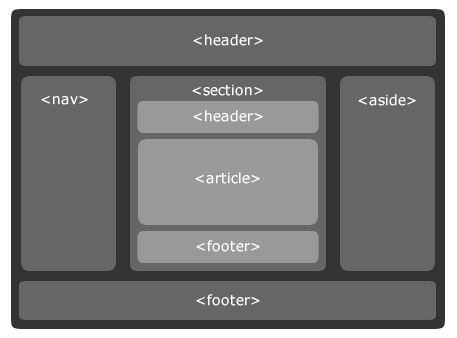
main : 主体

hgroup : 标题组合

nav : 导航

article : 独立的内容

aside : 辅助信息的内容

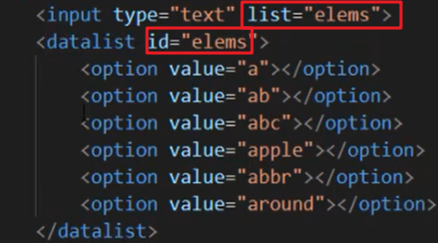


section（在article中嵌套） : 区域

figure : 描述图像或视频

figcaption : 描述图像或视频的标题部分

datalist :选项列表，自动提示（通过id连接input）



details / summary : 文档细节 / 文档标题

progress / meter : 定义进度条 / 定义度量范围





注：meta设置了low和high属性后会根据当前值与两个属性的比较变化颜色

time : 定义日期或时间

mark : 带有记号的文本

参考：<https://blog.csdn.net/qq_41229582/article/details/80637518>

## 表格扩展

添加单线 : border-collapse : collapse表格默认是双线且有空隙，添加此属性后变成单线

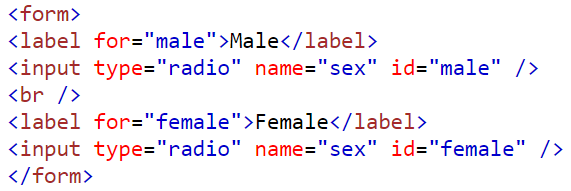
隐藏空单元(style中写): empty-cells : hide

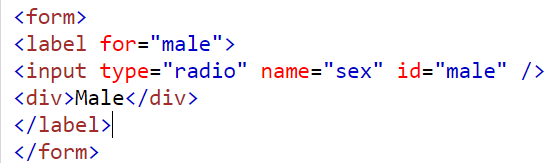
斜线分类 : border 或 rotate

列分组（同时对几列操作） : colgroup 和 col

## 表单扩展

### 美化表单





点击按钮或文本之一，就可以触发相关控件，因此可以利用此原理，将美化效果放在div里

1. label+:checked
2. position+opacity

### 新的input控件

email : 电子邮件地址输入框

url : 网址输入框

number : 数值输入框

range : 滑动条

date / month / week : 日期控件

search : 搜索框

color : 颜色控件

tel : 电话号码输入框（在移动端默认调起数字键盘）

time : 时间控件

### 新的表单属性

autocomplete : 自动完成（默认为on，会保留输入的历史）

autofocus : 获取焦点（打开界面光标自动在含该属性的栏）

required : 不能为空

pattern : 正则验证

method : 数据传输方式（GET/POST POST更安全）

enctype : 数据传输类型

name / value : 数据的键值对

### 扩展标签

fieldset : 表单内元素分组 （和legend组合使用）

legend : 为fieldset元素定义标题

optgroup : 定义选项组

## BFC规范

Formatting context(格式化上下文) 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。

BFC 即 Block Formatting Contexts (块级格式化上下文) ，它属于上述中的其中一种规范。具有 BFC 特性的元素可以看作是隔离了的独立容器，容器里面的元素不会在布局上影响到外面的元素，并且 BFC 具有普通容器所没有的一些特性

触发BFC规范的元素可以形成一个独立的容器，**不受外界的影响**，从而解决一些布局问题

### 触发BFC

浮动元素：float 除 none 以外的值

绝对定位元素：position (absolute、fixed)

display 为 inline-block、table-cells、flex

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

### BFC的应用

1. **解决margin叠加问题：**

上下两个div都设置距离对方固定距离的margin值会导致二者的距离取margin的最大值而不是和

解决：分别在两个div外再加一个div，使后加的div触动BFC规范，如对其设置overflow或定位等

1. **解决margin传递问题**

举例：两个div嵌套，对里面的盒子设置margin会导致两个盒子一起移动

解决：使里面的盒子触发BFC规范，使里面的盒子成为一个独立的容器，不受外面影响

1. 解决浮动问题

让收到浮动影响的容器触发BFC规范，成为一个独立的容器

1. 解决覆盖问题

## 浏览器前缀（背）

浏览器厂商以前就一直在实施CSS3，但它还未成为真正的标准。为此，当有一些CSS3样式语法还在波动的时候，他们提出了针对浏览器的前缀

|  |  |  |
| --- | --- | --- |
| **浏览器** | **内核** | **前缀** |
| **IE** | Trident | -ms- |
| **Firefox** | Gecko | -moz- |
| **Opera** | Presto | -o- |
| **Chrome** | Webkit | -webkit- |
| **Safari** | Webkit | -webkit- |
| **Opera、Chrome** | Blink |  |

## transtion过渡

transition-property : 规定设置过渡效果的CSS属性的名称。（width/height/all）

transition-duration : 规定完成过渡效果需要多少秒或毫秒。

transition-delay : 定义过渡效果何时开始。（延迟时间可正可负）

transition-timing-function : 规定速度效果的速度曲线。

transition-timing-function属性值

linear

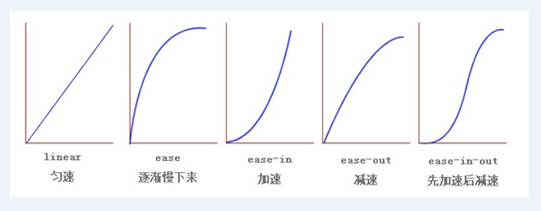
ease（默认值）

ease-in

ease-out

ease-in-out

cubic-bezier：贝赛尔曲线，各数值可以在下面的链接中获取



（ [http://cubic-bezier.com](%20http:/cubic-bezier.com%20) ）

复合写法

Transition：property（默认all） 共需多少时间duration 延迟多少时间delay 曲线类型

如：transition:1s 2s linear

**注：不要加到hover上**

## transform变形

translate : 位移（translateX translateY translateZ）

用法：transform:translate(10s,10s)

scale : 缩放，是一个比例值，若正常大小就是1，也有X Y Z的方向。会以当前元素中心点进行缩放。scale(值)只有一个元素时，代表X Y轴都是该值

用法同上

rotate : 旋转，**单位是deg**

rotateX（3d） rotateY（3d） rotateZ（平面，和rotate等值时效果同）

skew : 斜切

skewX：单位也是角度，正值向左倾斜，负值向右倾斜

skewY

注：设置多个值时候的顺序；设置transform-origin的基点位置。

## transform注意事项

1. 变形操作不会影响到其他元素
2. 变形操作只能添加给块元素，但不能添加给内联元素
3. 复合写法，可以同时添加多个变形操作，执行有顺序，先执行后面操作后执行前面操作

translate会受到rotate，scale，skew的影响，**如**transform：rotate(45deg) translate(100px,0);translate并不是水平方向沿x轴移动100px，而是受rotate角度影响所以是先向45度方向移动100px再旋转45度。

1. transform-origin：基点，变形操作的定点，默认为中心点（center,center）有三个方向，xyz

## animation动画

**animation-name** : 设置动画的名字 (自定义的)

**animation-duration** : 动画的持续时间

**animation-delay** : 动画的延迟时间

**animation-iteration-count** : 动画的重复次数 ，默认值就是1，infinite无限次数

**animation-timing-function** : 动画的运动形式

注：

From=0%；to=100%

**复合写法**：animation:名字 持续时间 延迟时间 循环次数 曲线

**animation-fill-mode** : 规定动画播放之前或之后，其动画效果是否可见。

none (默认值) : 在运动结束之后回到初始位置，在延迟的情况下，让0%在延迟后生效（延迟后变）

**backwards** : 在延迟的情况下，让0%在延迟前生效（延迟前变）

**forwards**  : 在运动结束的之后，停到结束位置

**both**  : backwards和forwards同时生效

**animation-direction** : 属性定义是否应该轮流反向播放动画。

**alternate** : 一次正向(0%~100%)，一次反向(100%~0%)

**reverse** : 永远都是反向 , 从100%~0%

**normal** (默认值) : 永远都是正向 , 从0%~100%

## animate.css

一款强大的预设css3动画库。

官网地址：https://daneden.github.io/animate.css/

基本使用：

animated : 基类(基础的样式，每个动画效果都需要加)

infinite : 动画的无限次数

## transform3D相关属性

rotateX() : 正值向上翻转

rotateY() : 正值向右翻转（记忆：正值是向第一象限翻转）

translateZ() : 正值向前，负值向后

scaleZ() : 立体元素的厚度

### 3d写法

scale3d() : 三个值 x y z

translate3d() : 三个值 x y z

**rotate3d()** : 四个值 0|1(x轴是否添加旋转角度) 0|1(y轴是否添加旋转角度) 0|1(z轴是否添加旋转角度) deg

**perspective** : 离屏幕多远的距离去观察元素，值越大幅度越小，越不明显。

**perspective-origin** : 景深-基点位置，观察元素的角度。

**transform-origin** : x y z（z轴只能写数值不能写单词）

**transform-style** : 3D空间

flat (默认值2d)、preserve-3d (3d，产生一个三维空间)

backface-visibility : 背面隐藏

hidden、visible (默认值)

## 背景扩展

**background-size :** 背景图的尺寸大小

cover : 覆盖（等比放大，会有空出来的情况）

contain : 包含 （等比放大，会有多出去的情况多出去的部分被隐藏）

**background-origin** : 背景图的填充位置

padding-box (默认)

border-box（在边框位置开始填充）

content-box（在content部分开始填充）

**background-clip** : 背景图的裁切方式

padding-box

border-box (默认) 在border就开始裁切图片

content-box 在内容区域裁切图片

写成复合样式时，第一个写的代表**background-origin**第二个写的代表**background-clip**

## CSS3渐变内容

**linear-gradient** : 线性渐变，是值，加到背景图（background-image）上,如：background-image:linear-gradient(to left/top/right/right bottom[改变渐变方向],起始颜色,…,终点颜色)

point（目标点） || angle（角度，含单位，0deg在页面下方，正值顺时针选择，负则逆时针）

color

percentage：设置渐变开始位置和结束位置

background-image:linear-gradient(red 25%,blue 75%)

若要在容器中加入两种颜色，则开始都设置为50%

**radial-gradient** : 径向渐变（了解）中心点向四周渐变

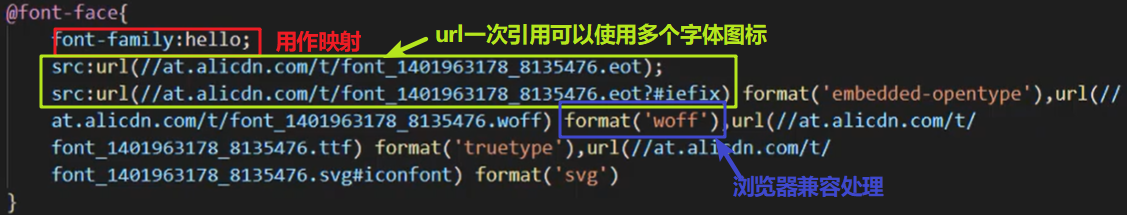
point

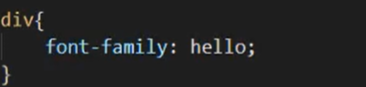
color

percentage

## 字体图标

font-face是CSS3中的一个模块，他主要是把自己定义的Web字体嵌入到你的网页中。





简化：将代表图标的字符通过伪类表示，再在body中调用样式

<style>

.shopping:after{content:”代表图标的字符”}

</style>

<body>

<div class=”shopping”></div>

</body>

好处：

1.可以非常方便的改变大小和颜色

2.放大后不会失真

3.减少请求次数和提高加载速度

4.简化网页布局

5.减少设计师和前端工程师的工作量

6.可使用计算机没有提供的字体

## Iconfont矢量图标库

**阿里巴巴矢量图标库**：

<https://www.iconfont.cn> : 提供了大量免费的字体图标。

**自定义字体图标**

[https://icomoon.io/app](%20https:/icomoon.io/app) : 在线生成字体图标。

使用方法：<https://www.bilibili.com/video/BV1R4411J7tQ?p=157>

## 文字阴影text-shadow

text-shadow:可选值：

x

y

blur：模糊值，值越大越模糊

color

多阴影：可以通过逗号的方式进行添加阴影样式

**注：阴影的默认颜色和文字颜色相同**

## 盒子阴影box-shadow

Box-shadow可选值

X

Y

Blur：模糊值

Spread：范围，扩大阴影的范围

Color

Inset:设置阴影为内阴影，默认值为外阴影

多阴影

注：默认颜色为黑色，和盒子颜色无关；默认就是外阴影，若设置outset则不起作用，关于内外阴影，有效的可选值只有inset内阴影

## 遮罩mask

**Mask可选值**

url：图片地址

repeat：设置重复方式

x，y：坐标值（写数值时，若hover加放大效果会以左上角为中心，为避免可以将坐标值写为center，center）

w：坐标值之间用斜线/分开，改变大小

h

多遮罩：-webkit-mask:url(‘图片地址’) no-repeat center center / width值 height值,url(‘’) no-repeat right center /width值 height值;

注：mask目前还没有标准化，所以需要加浏览器前缀，默认是x轴y轴都平铺

## 倒影box-reflect

box-reflect可选值

above：上

below：下

left：左

right：在右侧出现倒影

距离：单位px

遮罩 | 渐变：

遮罩：-webkit-box-reflect：below 20px url(‘图片地址’)

渐变：-webkit-box-reflect：below 20px linear-gradient(rgba(255,255,255,20%), rgba(255,255,255,1))

注：无标准写法，需要加浏览器前缀；此处的渐变只是针对透明度的渐变，不能支持颜色渐变

利用scale为负也可以实现翻转：

transform:scale(-1);实现倒影=below的效果，常用此方法实现倒影=below

transform:scaleX(-1);左右对称

transform:scaleY(-1);上下对称

## 模糊与计算

### blur模糊

filter:blur(n px);

### calc计算

width/height:calc(百分比和px进行运算)

用法：height:calc(100% - 80px);

**注，calc的括号里进行运算时必须有空格**

## 分栏布局

column-count : 分栏的个数

column-width : 分栏的宽度

column-gap : 分栏的间距

column-rule : 分栏的边线

column-span : 合并分栏

注：分栏的个数和宽度同时设置会有冲突，故不要一起设置

## 伪类和伪元素

CSS3中规定伪类由一个冒号开始，然后为伪类的名称；伪元素由两个冒泡开始，然后为伪元素的名称

伪类本质上是为了弥补常规CSS选择器的不足，以便获取到更多信息。通常表示获取不存在与DOM树中的信息，或获取不能被常规CSS选择器获取的信息。

伪元素本质上是创建了一个有内容的虚拟容器。这个容器不包含任何DOM元素，但是可以包含内容。另外，开发者还可以为伪元素定制样式。样式作用到这个虚拟容器上

如： :: selection

:: first-line / first-letter

:: before / after

## CSS hack

CSS Hack用来解决浏览器的兼容性问题，为不同版本的浏览器定制编写不同的CSS效果，使用每个浏览器单独识别的样式代码，控制浏览器的显示样式。

Hack分类

1. CSS属性前缀法

2. 选择器前缀法

3. IE条件注释法

### CSS属性前缀法

属性前缀法是在CSS样式属性名前加上一些只有特定浏览器才能识别的hack前缀，以达到预期的页面展现效果。

|  |  |
| --- | --- |
| **前缀标识** | **兼容浏览器** |
| **\_** | IE6 |
| **+、\*** | IE6、IE7 |
| **\9** | IE6、IE7、IE8、IE9 |
| **\0** | IE8、IE9、IE10、IE11 |

### 选择器前缀法

选择器前缀法是针对一些页面表现不一致或者需要特殊对待的浏览器，在CSS选择器前加上一些只有某些特定浏览器才能识别的前缀进行hack。

写法：IE6：\*html .box{修饰}

IE7：\*+html{修饰语句}

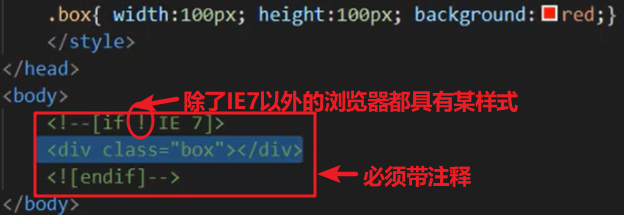
|  |  |
| --- | --- |
| **前缀标识** | **兼容浏览器** |
| **\*html** | IE6 |
| **\*+html** | IE7 |
| **:root** | IE9以上及现代浏览器 |

### IE条件注释法

这种方式是IE浏览器专有的Hack方式，微软官方推荐使用的hack方式。

**注：IE10以上已经不支持注释法**

|  |  |
| --- | --- |
| **前缀标识** | **兼容浏览器** |
| **<!--[if IE]>…<![endif]-->** | IE |
| **<!--[if IE 7]>…<![endif]-->** | IE7 |
| **<!--[if lte IE 7]>…<![endif]-->** | IE7以下 |
| **<!--[if ! IE 7]>…<![endif]-->** | 非IE7 |



其中”!”部分可以用lte(小于等于)，gte（大于等于）代替

### IE低版本BUG（了解）

由于旧浏览器在设计上有很多缺陷，导致一些BUG的产生，所以针对这些问题需要做出兼容处理。

常见兼容问题

1. 透明度（IE8及以下版本不识别）兼容性写法：filter:alpha(opacity=n);其中n取值为1-100

2. 双边距：IE6下含margin-left的div在设置浮动时会在左面出现2倍边距。解决办法：加上：\_display:inline;

3. 最小高度：解决办法：加overflow:hidden;

4. 图片边框

给图片加链接时会在图片外加边框，解决方法：img{border:none}

## 渐进增强与优雅降级

渐进增强 ：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，以达到更好的用户体验。

优雅降级 ：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。

## 网页布局扩展

### 等高布局（margin-bottom和padding-bottom配合实现）

等高布局：设置成当任何一面变长，对面都跟着变到相同高度

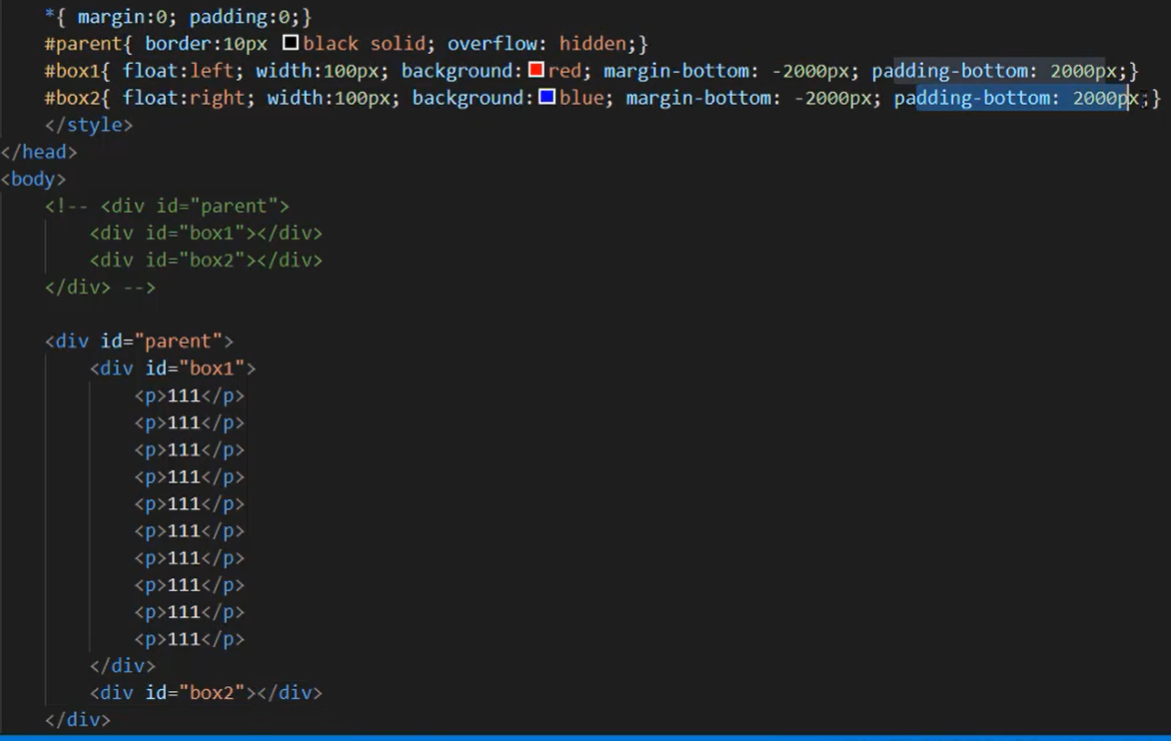


空间撑开的高度由元素本身的高度和margin间距共同决定（图中两个div高100px，margin-bottom值为-150px，若margin-bottom值过大会导致添加的内容无法显示）

解决此问题：设置padding-bottom:150px进行抵消再添加内容，空间高度会和内容高度一样



实现等高布局：





### 双飞翼布局

双飞翼布局：左侧和右侧都固定，中间自适应，适合做3D布局

做三列布局，左右固定，中间自适应布局的方式

1. BFC方式
2. 定位
3. 浮动（双飞翼布局和圣杯布局）
4. **flex弹性（现在常用）**

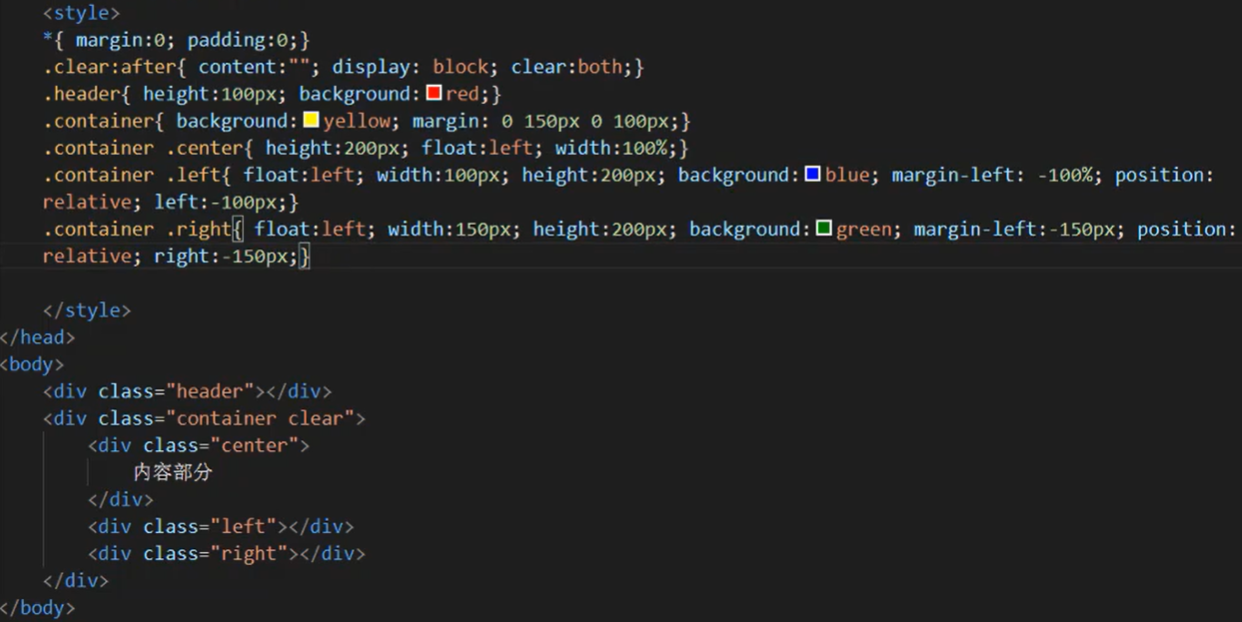


特点：利用margin的负值和浮动实现

1. 先写中间部分
2. 三部分都要设置浮动
3. 左侧和右侧都要通过margin负值移动到指定位置
4. 让中间的内容部分利用margin固定在中间

### 圣杯布局

和双飞翼的区别：圣杯布局更加简洁



## Flex弹性盒模型

随着移动互联网的发展，对于网页布局来说要求越来越高，而传统的布局方案对于实现特殊布局非常不方便，比如垂直居中。

2009年，W3C 提出了一种新的方案----Flex 布局，可以简便、完整、响应式地实现各种页面布局。目前，它已经得到了所有浏览器的支持，这意味着，现在就能很安全地使用这项功能。

|  |  |
| --- | --- |
| **作用在flex容器上** | **作用在flex子项上** |
| flex-direction | order |
| flex-wrap | flex-grow |
| flex-flow | flex-shrink |
| justify-content | flex-basis |
| align-items | flex |
| align-content | align-self |

### flex-direction

flex-direction用来控制子项整体布局方向，是从左往右还是从右往左，是从上往下还是从下往上。

|  |  |
| --- | --- |
| **取值** | **含义** |
| **row** | 默认值，显示为行。方向为当前文档水平流方向，默认情况下是从左往右。 |
| **row-reverse** | 显示为行。但方向和row属性值是反的 |
| **column** | 显示为列 |
| **column-reverse** | 显示为列。但方向和column属性值是反的 |

### flex-wrap

默认情况下，flex弹性盒模型会自动调节加入的内容的宽度，但如果内容过多超出盒子本身的宽度还是会溢出

flex-wrap用来控制子项整体单行显示还是换行显示。

|  |  |
| --- | --- |
| **取值** | **含义** |
| **nowrap** | 默认值，表示单行显示，不换行。 |
| **wrap** | 宽度不足换行显示 |
| **wrap-reverse** | 宽度不足换行显示，但是是从下往上开始，也就是原本换行在下面的子项现在跑到上面。（换行后行间有空隙） |

### flex-flow

flex-flow属性是flex-direction和flex-wrap的缩写，表示flex布局的flow流动特性。**第一个值表示方向，第二个值表示换行**，中间用空格隔开。简单讲就是上面两个属性的**复合写法**

### justify-content

justify-content属性决定了主轴方向(和flex-direction设置的方向一致的方向)上子项的对齐和分布方式。

|  |  |
| --- | --- |
| **取值** | **含义** |
| **flex-start** | 默认值，表现为起始位置对齐。 |
| **flex-end** | 表现为结束位置对齐。 |
| **center** | 表现为居中对齐。 |
| **space-between** | 表现为两端对齐。between是中间的意思，意思是多余的空白间距只在元素中间区域分配。 |
| **space-around** | around是环绕的意思，意思是每个flex子项两侧都环绕互不干扰的等宽的空白间距，最终视觉上边缘两侧的空白只有中间空白宽度一半。 |
| **space-evenly** | evenly是匀称、平等的意思。也就是视觉上，每个flex子项两侧空白间距完全相等。 |

### align-items（一行中每个子项的排列）

align-items中的items指的就是flex子项们，因此align-items指的就是flex子项们相对于flex容器在侧轴方向上的对齐方式。

|  |  |
| --- | --- |
| **取值** | **含义** |
| **stretch** | 默认值，flex子项拉伸。 |
| **flex-start** | 表现为容器顶部对齐。 |
| **flex-end** | 表现为容器底部对齐。 |
| **center** | 表现为垂直居中对齐。 |

### align-content（多行之间的对齐方式，是行与行）

align-content可以看成和justify-content是相似且对立的属性，如果所有flex子项只有一行，则align-content属性是没有任何效果的。

|  |  |
| --- | --- |
| **取值** | **含义** |
| **stretch** | 默认值。每一行flex子元素都等比例拉伸。例如，如果共两行flex子元素，则每一行拉伸高度是50%。 |
| **flex-start** | 表现为起始位置对齐。 |
| **flex-end** | 表现为结束位置对齐。 |
| **center** | 表现为居中对齐。 |
| **space-between** | 表现为两端对齐。 |
| **space-around** | 每一行元素上下都享有独立不重叠的空白空间。 |
| **space-evenly** | 每一行元素都完全上下等分。 |

### 作用在flex上的css属性

|  |  |
| --- | --- |
| **取值** | **含义** |
| **order** | 可以通过设置order改变某一个flex子项的排序位置。所有flex子项的默认order属性值是0。 |
| **flex-grow** | 属性中的grow是扩展的意思，扩展的就是flex子项所占据的宽度，扩展所侵占的空间就是除去元素外的剩余的空白间隙。默认值为0。所有值≥0 |
| **flex-shrink** | 属性中的shrink是“收缩”的意思，flex-shrink主要处理当flex容器空间不足时候，单个元素的收缩比例。默认值是1。所有值≥0 |
| **flex-basis** | flex-basis定义了在分配剩余空间之前元素的默认大小。同时存在时，flex-basis的优先级高于width |
| **flex** | flex属性是flex-grow，flex-shrink和flex-basis的缩写。默认为flex:0 1 auto;只写一个时代表grow值，flex的优先级高于单写flex-grow等。（flex>width>flex-grow） |
| **align-self** | align-self指控制单独某一个flex子项的垂直对齐方式。Align-items针对一行中所有的，而align-self针对一行里的某一个 |

## grid网格布局

Grid布局是一个二维的布局方法，纵横两个方向总是同时存在。

|  |  |
| --- | --- |
| **作用在grid容器上** | **作用在grid子项上** |
| grid-template-columns | grid-column-start |
| grid-template-rows | grid-column-end |
| grid-template-areas | grid-row-start |
| grid-template | grid-row-end |
| grid-column-gap | grid-column |
| grid-row-gap | grid-row |
| grid-gap | grid-area |
| justify-items | justify-self |
| align-items | align-self |
| place-items | place-self |
| justify-content |  |
| align-content |  |
| place-content |  |

### grid-template-columns和grid-template-rows

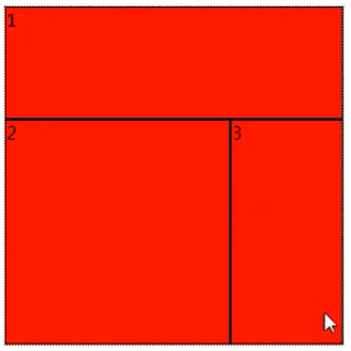
对网格进行横纵划分，形成二维布局。单位可以是像素，百分比，自适应以及**fr单位**（网格剩余空间比例单位）。

有时候，我们网格的划分是很规律的，如果需要添加多个横纵网格时，可以利用repeat()语法进行简化操作。repeat(重复次数，每次重复的值[含单位])

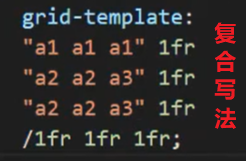
### grid-template-areas和grid-template

area是区域的意思，grid-template-areas就是给我们的网格划分区域的。此时grid子项只要使用grid-area属性指定其隶属于那个区。





grid-template是grid-template-rows，grid-template-columns和grid-template-areas属性的缩写。



### grid-column-gap和grid-row-gap

grid-column-gap和grid-row-gap属性用来定义网格中网格间隙（间距）的尺寸。

CSS grid-gap属性是grid-row-gap和grid-column-gap属性的缩写，先横后纵。

### justify-items和align-items

**justify-items**指定了网格元素的水平呈现方式，是水平拉伸显示，还是左中右对齐，**是针对当前的网格内元素的**。**Justify-content：是针对所有网格的排列**；align-items指定了网格元素的垂直呈现方式，是垂直拉伸显示，还是上中下对齐。

**place-items可以让align-items和justify-items属性写在单个声明中。（第一个值代表纵向第二个代表横向）**

|  |  |
| --- | --- |
| **取值** | **含义** |
| **stretch** | 默认值，拉伸。表现为水平或垂直填充。 |
| **Start** | 表现为容器左侧或顶部对齐。 |
| **end** | 表现为容器右侧或底部对齐。 |
| **center** | 表现为水平或垂直居中对齐。 |

### justify-content和align-content

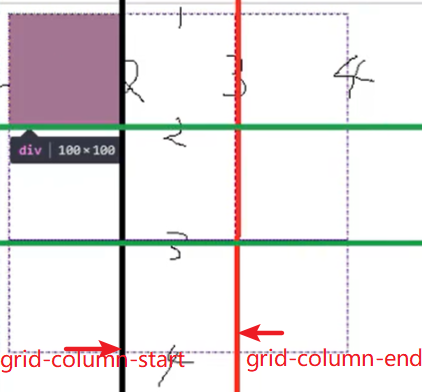
justify-content指定了网格元素的水平分布方式。align-content指定了网格元素的垂直分布方式。**place-content**可以让align-content和justify-content属性写在一个CSS声明中。第一个值是纵向的，第二个是横向的

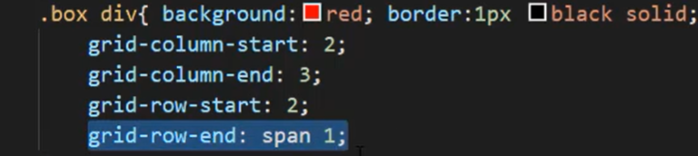
|  |  |
| --- | --- |
| **取值** | **含义** |
| **stretch** | 默认值，拉伸。表现为水平或垂直填充。 |
| **start** | 表现为容器左侧或顶部对齐。 |
| **end** | 表现为容器右侧或底部对齐。 |
| **center** | 表现为水平或垂直居中对齐。 |
| **space-between** | 表现为两端对齐。 |
| **space-around** | 享有独立不重叠的空白空间。 |
| **space-evenly** | 平均分配空白空间 |

### 作用在grid子项上的css属性

|  |  |
| --- | --- |
| **取值** | **含义** |
| **grid-column-start** | 水平方向上占据的起始位置。 |
| **grid-column-end** | 水平方向上占据的结束位置。（span属性） |
| **grid-row-start** | 垂直方向上占据的起始位置。 |
| **grid-row-end** | 垂直方向上占据的结束位置。（span属性） |
| **grid-column** | grid-column-start + grid-column-end的缩写。 |
| **grid-row** | grid-row-start + grid-row-end的缩写。 |
| **grid-area** | 表示当前网格所占用的区域，名字和位置两种表示方法。 |
| **justify-self** | 单个网格元素的水平对齐方式。 |
| **align-self** | 单个网格元素的垂直对齐方式。 |
| **place-self** | align-self和justify-self的缩写。 |

1、





只有grid-column-end和grid-row-end含有span属性，加上span属性后含义变为从开始位置向后的n个格子都具有某个样式

2、grid-column和grid-row

grid-column是grid-column-start + grid-column-end的缩写，用法是：grid-column:开始行 / 结束行，grid-row同理

3、**grid-area:横向起始 / 纵向起始 / 横向结束 / 横向结束**

## Viewport视口

在**移动端**viewport视口就是浏览器显示页面内容的屏幕区域。在viewport中有两种视口，分别表示为， visual viewport（可视视口）和layout viewport（布局视口）。

visual viewport 固定大小跟屏幕大小相同，在上面，而layout viewport 可改变大小，在下面。Layout viewport默认大小为980像素，可通过document.documentElement.clientWidth获取。

现代网页需要将layout viewport设置成跟visual viewport等同大小，方便进行网页制作。

### Viewport设置

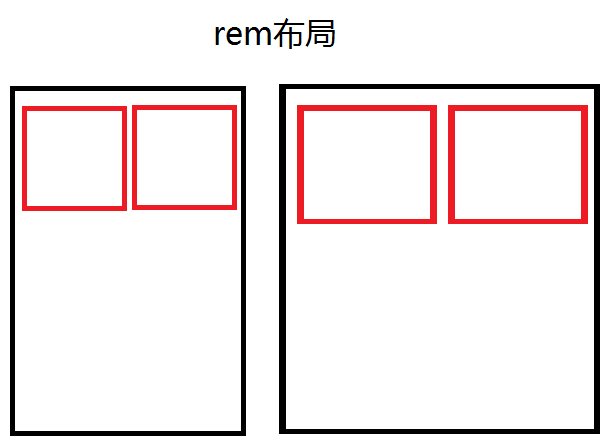
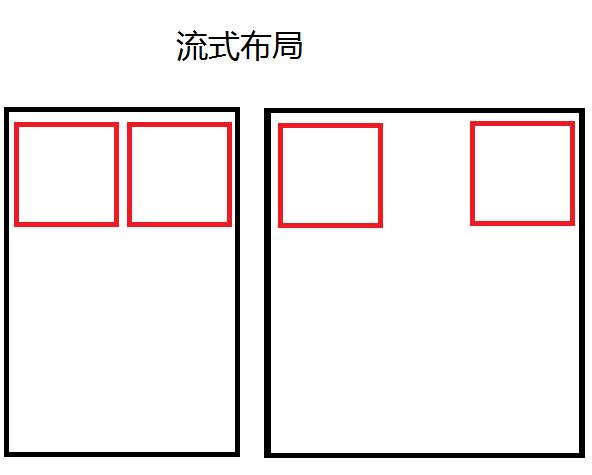
通过<meta>标签进行设置，name属性指定viewport值，content属性进行视口配置。

|  |  |
| --- | --- |
| **取值** | **含义** |
| **width** | 设置layout viewport的宽度特定值，device-width表示设备宽。 |
| **height** | 设置layout viewport的高度特定值，一般不进行设置。 |
| **initial-scale** | 设置页面的初始缩放。 |
| **minimum-scale** | 设置页面的最小缩放。 |
| **maximum-scale** | 设置页面的最大缩放。 |
| **user-scalable** | 设置页面能否进行缩放。 |

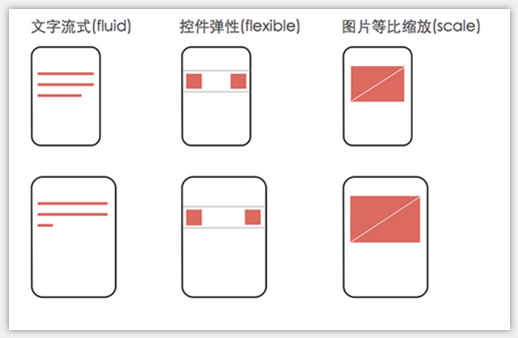
## 移动端适配方案

1. 百分比布局，也叫流式布局。代表网站：优酷、百度、天猫、腾讯。

2. 等比缩放布局，也叫rem布局。代表网站：网易、爱奇艺、淘宝、美团。



### 流式布局原则



好处： 大屏幕下显示更多的内容。 坏处：宽屏下比例会有一些不协调。

### Rem布局

单位：

em : 是一个相对单位，1em等于当前元素或父元素（当前元素没有设置font-size时）的font-size值。

rem : 是一个相对单位，1rem等于根元素(html)的font-size值。动态设置HTML的font-size，且通过rem单位设置不同div的长宽，可以使用一套代码在不同的设备下等比显示

vw / vh : 把屏幕分为100份，1vw等于屏幕宽的1%，同理1vh是屏幕高的1%。

### 动态设置font-size

**1、通过JS**

**2、通过vw**

通常修改HTML的fontsize为26.66667vm

注：要给body重置一下font-size : 16px。因为HTML里的font-size默认是16px，但是需要改变其默认值方便rem换算，所以在比HTML优先级高的body里对font-size进行重置设置，使默认值仍为16px

## 响应式布局

利用媒体查询，即media queries，可以针对不同的媒体类型定义不同的样式，从而实现响应式布局 。



### 媒体类型

|  |  |
| --- | --- |
| **取值** | **含义** |
| all | 用于所有设备。 |
| print | 用于打印机和打印预览。 |
| screen | 用于电脑屏幕，平板电脑，智能手机等。 |
| speech | 应用于屏幕阅读器等发声设备。 |

### 常见选项：

媒体类型

and、not（在all前加not表示取反）

min-width、max-width



orientation:portrait（方向为纵屏）、orientation:landscape（方向为横屏）



<link>



### 常见修改样式

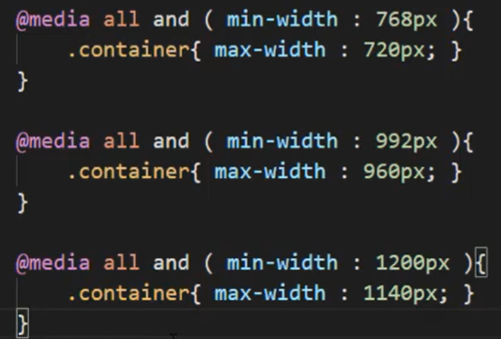
display

float

width

注：响应式代码写到要适配的CSS后面。

### 响应式的两种操作模式





## bootstrap

Bootstrap 是最受欢迎的 HTML、CSS 和 JS 框架，用于开发响应式布局、移动设备优先的 WEB 项目。

特色：

1. 响应式布局

2. 基于flex的栅格系统

3. 丰富的组件和工具方法

4. 常见交互使用

官网：<https://getbootstrap.com/>

### containers

container

container-fluid

Responsive breakpoints



### Grid system

Grid options

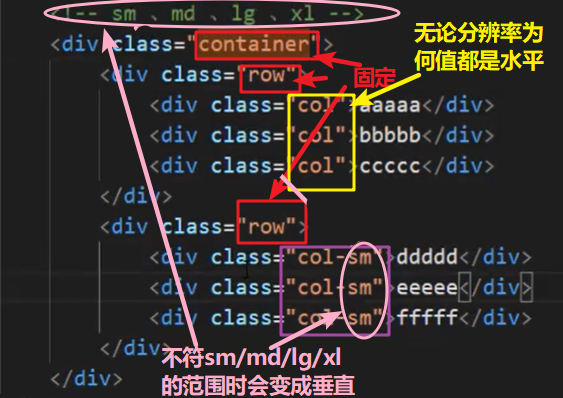
Responsive classes响应式的class

Gutters间距

Alignment对齐方式

Reordering排序

Offsetting间距



## C

### R