使用CSS样式

HTML是内容控制，CSS是格式控制。网页格式的设置先找到元素（使用选择器），然后应用样式。

# HTML中添加样式

## 内嵌样式

|  |
| --- |
| <table style="color:red;margin-right:220px">  ...  </table> |

## 内部样式表(head中)

|  |
| --- |
| <style type="text/css">  body{  /\*注释\*/  }  </style> |

## 链接外部样式表

|  |
| --- |
| <head>  <link rel="stylesheet" type="text/css" href="../....css"/>  ...  </head> |

# CSS中常用的选择器

语法结构：选择器{属性1:取值1；属性2:取值2;...}

选择器的权重规则：标签的权值为1，类选择符的权值为10，ID选择符的权

值最高为100。还有一个权值比较特殊--继承也有权值但很低，有的文献提出

它只有0.1，所以可以理解为继承的权值最低。

css层叠特性：就是在html文件中对于同一个元素可以有多个css样式存在，当有相同权重的样式存在时，会根据这些css样式的前后顺序来决定，处于最后面的css样式会被应用。

设置最高权限：p{color:red!important;}

## 基本选择器

|  |
| --- |
| #id选择器  标签{... } 例如：p{color:red}  #类选择器  .class{… } 例如：.item{color:red}  #标签选择器  标签{… } 例如：p{color:red}  #群组选择器  p，h1{... }  #通配符选择器  \*{…} |

## 层级选择器

|  |
| --- |
| #后代选择器  div p { color : red }  #子代选择器(ie6不支持)  div>p { color : red }  #兄弟选择器  div + p { color : red } //div下一个兄弟p（找的是下一元素）  div ~ p {color : red } //div后面所有兄弟p |

## 属性选择器

|  |
| --- |
| [属性名] : 选取含有指定属性的元素  [属性名="属性值"] : 选取属性值等于指定值的元素  [属性名^="属性值"] : 选取属性值以指定内容开头的元素  [属性名$="属性值"] : 选取属性值以指定内容结尾的元素  [属性名\*="属性值"] : 选取属性值中包含指定内容的元素 |

## 伪类选择器

伪类和伪元素用来表示元素所处的一个特殊的状态，或者是一个特殊的位置。

|  |
| --- |
| :link a:link：选择所有未访问链接  :visited a:visited：选择所有访问过的链接(只能设置字体颜色)  :active a:active：选择正在活动链接  :hover a:hover：把鼠标放在链接上的状态  注意：hover和active可以给其他元素设置（IE6不支持），link和visited只能给a设置。，a:hover 必须被置于 a:link 和 a:visited 之后，才是有效的；a:active 必须被置于 a:hover 之后，才是有效的；伪类的名称不区分大小写。  :focus input:focus：选择元素输入后具有焦点  ::selection 表示内容被选中的状态(在火狐中使用::-moz-selection来代替)  :first-letter p:first-letter：选择每个<p> 元素的第一个字母  :first-line p:first-line：选择每个<p> 元素的第一行  :before p:before：在每个<p>元素之前插入内容, 一般该伪类都会结合content一起使用，通过content可以向指定位置添加内容  :after p:after：在每个<p>元素之后插入内容  #子元素伪类  :first-child : 寻找父元素的第一个子元素，在所有的子元素中排序  :last-child: 寻找父元素的最后一个子元素，在所有的子元素中排序  : :nth-child : 寻找父元素中的指定位置子元素，在所有的子元素中排序。p:nth-child(3) ，可以使用even/odd，来找到偶/奇数的子元素  :first-of-type : 寻找指定类型中的第一个子元素  :last-of-type : 寻找指定类型中的最后一个子元素  :nth-of-type : 寻找指定类型中的指定子元素  注意：XXX-child和XXX-of-type的区别在于，后面实在所有指定类型的子元素进行排序；而XXX-child在所有的子元素中排序。  #否定伪类  :not(选择器)：从一组元素中将符合要求的元素剔除出去.abc:not(div) |

## 选择器的特性

|  |
| --- |
| #样式的继承  为祖先元素设置样式，会同时应用到它的后代元素上，这一特性称为样式的继承。通过样式的继承可以将一些样式统一设置个祖先元素，这样所有的后代都会应用到相同的样式。但是并不是所有的样式都会继承，比如：背景相关的，边框相关的，定位相关的。具体参考文档  #选择器的优先级  样式表执行顺序：先应用默认样式->使用继承样式->使用通配符选择器样式->使用元素选择器样式->使用类和伪类选择器样式->使用id选择器样式->使用内联样式-> !important样式执行（后面的会覆盖前面的样式，交集选择器是效果相加，优先级一样的会按代码顺序执行） |

# 字体属性

## 字体font-family

font-family: "字体1（宋体）","字体2",...

如果不支持字体1就会采用字体2。这里注意不要设置不常用的字体，因为如

果用户本地电脑上如果没有安装你设置的字体，就会显示浏览器默认的字体。

## 字号font-size

font-size:字体大小 //em 值不是固定,根据父级元素的大小来进行计算

font-size:12px

字体大小：xx-small,x-small,small,medium(正常）,large,x-large,...

注意：如果你不指定一个字体的大小，默认大小和普通文本段落一样，是16像素（16px=1em）。

## 字体风格font-style

font-style:样式取值（italic/normal/obliqe(中间态，偏斜体））

## 加粗字体font-weight

font-weight:字体粗细值

字体粗细值：normal正常,bold粗体,bolder特粗.lighter特细

number（100~900）

## 小写字母转换为大写font-variant

font-variant：取值（normal，small-caps(字体变小））

## 字体符合属性font

## font:字体风格 字体大小 字体加粗 字体类型（font-size和font-family必须写这两个，必须是倒数两个样式）

# 颜色和背景属性

## 颜色属性

color：颜色取值

颜色取值：英文名（red,green,blue).或者使用16进制数字来表示颜色

#xxxxxx(可以表示1677万种颜色）

## 背景颜色

background-color:#xxxxxx

## 背景图片

background-image:url(图像地址）

## 背景重复

background-repeat:no-repeat(不平铺）/repeat-x水平铺/repeat-y垂

直铺‘

## 背景附件

background-attachment:scroll(默认）/fixed(背景固定不动）

## 背景位置

background-position:left/top/right/bottom/..%/

## 背景符合属性

background:背景颜色 背景图片 背景重复 背景附件 背景位置

background:url(images/bg.jpg) no-repeat left top

注意：背景样式会应用于盒子模型的 内容区+padding区

# 段落属性

## 单词间隔

word-spacing:取值

word-spacing:3px，

## 字符间隔

letter-spacing:取值

letter-spacing:3px

注意：这个样式使用在英文单词时，是设置字母与字母之间的间距。

## 文字修饰

text-decoration:underline/overline/line-through/blink(闪烁)

## 文本转换

text-transform:none/capitalize(首字母大写)/uppercase/lowercase

## 水平对齐

text-align:left/right/center/justify(两端对齐）

## 垂直对齐

vertical-align:sub/super/top/text-top/middle/text-bottom

## 文本缩进

text-indent:缩进值

text-indent:25px（2em）

## 文本行高(行之间的间距,也叫行间距)

line-height:行高值

line-height:25px

## 处理空白

white-space:nomal/pre/nowrap(同一行显示）

## 文本反排

unicode-bidi:direction(按direction属性值重排）

direction:ltr(左到右）/rtl(右到左)

word-wrap: break-word 在长单词或 URL 地址内部进行换行。

# 盒子模型

CSS盒模型本质上是一个盒子，封装周围的HTML元素，它包括：边距，边框，填充，和实际内容。盒模型允许我们在其它元素和周围元素边框之间的空间放置元素。

Margin - 清除边框区域。Margin没有背景颜色，它是完全透明

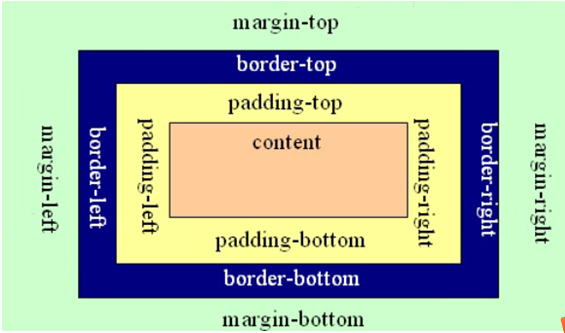
Border - 边框周围的填充和内容。边框是受到盒子的背景颜色影响

Padding - 清除内容周围的区域。会受到框中填充的背景颜色影响

Content - 盒子的内容，显示文本和图像

元素的总宽度计算公式：总元素的宽度=宽度+左填充+右填充+左边框+右边框+左边距+右边距（width来设置）

元素的总高度计算公式：总元素的高度=高度+顶部填充+底部填充+上边框+下边框+上边距+下边距（height来设置）



子元素默认在父元素的内容区。

## 元素分类

在CSS中，html中的标签元素大体被分为三种不同的类型：块状元素、内联元素(又叫行内元素)和内联块状元素。

常用的块状元素有：<div>、<p>、<h1>...<h6>,<ol>,<ul>,<dl>,<table>,<address>,<blockquote> ,<form>

常用的内联元素有：<a>,<span>,<br>,<i>,<em>,<strong>, <label>,<q>,<var>,<cite>,<code>

常用的内联块状元素有： <img>、<input>

### 块级元素

在html中<div>、 <p>、<h1>、<form>、<ul> 和 <li>就是块级元素。设 置display:block就是将元素显示为块级元素。如下代码就是将内联元素a转换为块状元素，从而使a元素具有块状元素特点。a{display:block;}

块级元素特点：每个块级元素都从新的一行开始，并且其后的元素也另起一行。（一个块级元素独占一行）；元素的高度、宽度、行高以及顶和底边距都可设置。元素宽度在不设置的情况下，是它本身父容器的100%（和父元素的宽度一致），除非设定一个宽度。

### 内联元素

在html中，<span>、<a>、<label>、 <strong> 和<em>就是典型的内联元素（行内元素）（inline）元素。当然块状元素也可以通过代码display:inline将元素设置为内联元素。如下代码就是将块状元素div转换为内联元素，从而使 div 元素具有内联元素特点。div{ display:inline; }

内联元素特点：和其他元素都在一行上；内联元素设置宽度和高度没有效果，可以设置左右内边距也可以设置垂直方向内边距，但是不会影响整体布局。支持四个方向边框，但是垂直的边框不会影响布局 。支持水平方向外边距，不支持垂直方向的。

### 内联块状元素

内联块状元素（inline-block）就是同时具备内联元素、块状元素的特点，代码display:inline-block就是将元素设置为内联块状元素。(css2.1新增)， <img>、<input>标签就是这种内联块状标签。

inline-block 元素特点：和其他元素都在一行上； 元素的高度、宽度、行高以及顶和底边距都可设置

## 外边距属性

margin清除周围的元素（外边框）的区域。margin没有背景颜色，是完全透明的。margin可以单独改变元素的上，下，左，右边距。也可以一次改变所有的属性。

上边距：margin-top:20px //像素、百分比

下边距：margin-bottom:20px //像素、百分比

左边距：margin-left:20px //像素、百分比

右边距：margin-right:20px //像素、百分比

边距综合属性：margin:如果只设一个值应用四个边界，2个值的时候应用上下和左右，3个值则应用上右下；设置四个值按照上右下左的顺时针。

注意：外边距不会影响盒子模型的大小。表示的当前盒子和其他盒子的距离。主要用来设置盒子位置的。上边距和左边距影响自己的位置，右下边距影响其他盒子和当前盒子的距离。设置边距值为负值会导致盒子向相反的方向移动。

相邻的垂直外边距会发生重叠现象（相邻元素的外边距会取最大值，子元素的外边距会传递给父元素）。水平外边距不会重叠，而是取和。

## 内边距属性（填充），

CSS Padding（填充）属性定义元素边框与元素内容之间的空间。当元素的 Padding（填充）（内边距）被清除时，所"释放"的区域将会受到元素背景颜色的填充。单独使用填充属性可以改变上下左右的填充。缩写填充属性也可以使用，一旦改变一切都改变。设置同margin。

## 边框属性

CSS边框属性允许你指定一个元素边框的样式和颜色。

### 边框样式(边框旁边的线）

border-style:样式

border-top-style:样式

border-right-style:样式

border-bottom-style:样式

border-left-style:样式

样式：none/dotted/dashed/solid/double/groove/ridge/inset/outset

无/点/虚线/实线/双实线/立体/脊状/...

### 边框宽度

border-width:取值

border-top-width:取值

border-right-width:取值

border-bottom-width:取值

border-left-width:取值

取值：像数/medium(默认宽度)/thin(小于默认宽度)/thick(大于默认宽度)

### 边框颜色

border-color:颜色值

border-top-color:颜色值

border-right-color:颜色值

border-bottom-color:颜色值

border-left-color:颜色值

### 边框属性border

border:border-style border-width border-color(没有顺序要求)

border-top:2px solid #00ccff

border-right:2px solid #00ccff

border-bottom:2px solid #00ccff

border-left:2px solid #00ccff

注意：大部分浏览器中边框的颜色和宽度都有默认值。边框样式的默认值为none

# 定位属性

静态定位，position:static;是元素的默认的定位方式，简单的地按照文档的普通流动，top和left属性无效。

相对定位，position:relative;元素还是会按照文档的普通流动，但是，top和left属性的值会使得它相对它本来的位置作偏移。注意，在使用相对定位时，无论是否进行移动，元素仍然占据原来的空间。因此，移动元素会导致它覆盖其它框。

绝对定位，position:absolute;元素完全跳出页面布局的普通流动，它会相对于它的第一个非静态定位的祖先而展示。如果没有这样的祖先，则相对于整个文档。绝对定位与文档流无关，所以它们可以覆盖页面上其他的元素，可以通过z-index属性来控制这些层的对方顺序。

固定定位，position:fixed;元素相浏览器窗口而定位。它会忽略浏览器的滚动条，一直可见的！

注意：position:absolute，float会隐式地改变display的类型（display：none除外）。即是当元素设置position:absolute、float:left、float:right中任意一个时，都会让元素以display:inline-block的方式显示（特点是：可以设置长宽，默认宽度不占满父元素）。这时，即使故意设置display:inline; display:block都是无效的。但是float在IE6下的双边距bug就是用display:inline; 来解决的。

position:relative不会隐式改变display的类型。

## 定位方式

position:static/absolute/fixed/relative

static：HTML元素的默认值，即没有定位，元素出现在正常的流中。静态定位的元素不会受到top, bottom, left, right影响。

fixed：元素的位置相对于浏览器窗口是固定位置。即使窗口是滚动的它也不会移动。注意： Fixed 定位在 IE7 和 IE8 下需要描述 !DOCTYPE 才能支持；Fixed定位使元素的位置与文档流无关，因此不占据空间；Fixed定位的元素和其他元素重叠。

relative：相对其原来的位置。

absolute：相对于最近的已定位父元素，如果元素没有已定位的父元素，那么它的位置相对于<html>。Absolutely定位使元素的位置与文档流无关，因此不占据空间。Absolutely定位的元素和其他元素重叠。

## 元素位置：

top/right/bottom/left:auto/长度值25px、百分比

## 层叠顺序：

z-index:auto|数字 //默认情况为1，数字越大越在上面（）

## 浮动属性：

float:none(默认值对象浮动)/left(浮动在元素左边)/right(浮动在元素右边)

元素的水平方向浮动，意味着元素只能左右移动而不能上下移动。一个浮动元素会尽量向左或向右移动，直到它的外边缘碰到包含框或另一个浮动框的边框为止。浮动元素之后的元素将围绕它。浮动元素之前的元素将不会受到影响。

如果图像是右浮动，下面的文本流将环绕在它左边。

## 清除属性：

clear:none(可以浮动)/left(不许左浮动)/right(不许右浮动)/both(不许浮动)

## 可视区域：

clip:auto | rect(数字按照上右下左,auto)

## 层的宽高：

width:auto/长度值

height:auto/长度值

## 超出范围：

overflow:visible| auto| hedden|scroll超出部分显示方式）

## 可见属性：

visibility:inherit(继承父类的可见性)/visible/hidden

#display:none和visibility:hidden的区别

display属性none ：此元素不被显示；block ：此元素按块级元素显示：前后带换行符，自己占一行。inline ：此元素按内联元素显示：1个挨着1个。

相同点：visibility和display两个属性都有隐藏元素的功能

display:none和visibility:hidden的区别:visibility:hidden隐藏，但在浏览时保留位置，会影响布局；而display:none视为不存在，且不加载！

# 列表属性

list-style:list-style-image | list-style-position | list-style-type

列表符号：list-style-type:disc/circle/square/decimal/lower- roman/upper-roman/lower-alpha/upper-alpha/none

图片符号：list-style-image:none | url(图像地址)

列表缩进：list-style-position:outside/inside(列表标记放在文本内，对齐)

# 光标属性

cursor:auto(默认) | 形状取值 | url

形状取值：default/hand/crosshair(十字光标)/test(I)/help/wait/e-resize/ne-resize/n-resize/nw-resize/w-sesize...(eswn东南西北...箭头）

# 布局模型

## 流动模型（Flow）

流动（Flow）是默认的网页布局模式。也就是说网页在默认状态下的 HTML 网页元素都是根据流动模型来分布网页内容的。流动布局模型具有2个比较典型的特征：

第一点，块状元素都会在所处的包含元素内自上而下按顺序垂直延伸分布，因为在默认状态下，块状元素的宽度都为100%。实际上，块状元素都会以行的形式占据位置。如右侧代码编辑器中三个块状元素标签(div，h1，p)宽度显示为100%。高度默认是子元素高度。

第二点，在流动模型下，内联元素都会在所处的包含元素内从左到右水平分布显示。（内联元素可不像块状元素这么霸道独占一行），默认高度和宽度都是被内容撑开。

## 浮动模型 (Float)

任何元素在默认情况下是不能浮动的，但可以用 CSS 定义为浮动，如 div、p、table、img 等元素都可以被定义为浮动。如下代码可以实现两个 div 元素一行显示。

|  |
| --- |
| <div id="div1"></div>  <div id="div2"></div>  div{  width:200px;  height:200px;  border:2px red solid;  float:left;  } |

浮动特点：元素浮动以后会完全脱离文档流；元素浮动的位置是上一个块元素的左上或者右上浮动，浮动元素会盖住文档流中元素；多个浮动元素的宽度大于页面宽度，后面浮动的元素会换行展示；浮动元素不会覆盖文字，文字会自动环绕在浮动元素的周围，可以通过浮动来实现文字环绕的效果

脱离文档流的特点：块元素的宽高默认被元素撑开，内联元素也可以设置宽高。

## 层模型（Layer）

布局模型就像是图像软件PhotoShop中非常流行的图层编辑功能一样，每个图层能够精确定位操作，但在网页设计领域，由于网页大小的活动性，层布局没能受到热捧。但是在网页上局部使用层布局还是有其方便之处的。如何让html元素在网页中精确定位，就像图像软件PhotoShop中的图层一样可以对每个图层能够精确定位操作。CSS定义了一组定位（positioning）属性来支持层布局模型。层模型有三种形式：

* 绝对定位(position: absolute)
* 相对定位(position: relative)
* 固定定位(position: fixed)

### 层模型--绝对定位

要设置position:absolute(表示绝对定位)，这条语句的作用将元素从文档流中拖出来，然后使用left、right、top、bottom属性相对于其最接近的一个具有定位属性的父包含块进行绝对定位。如果不存在这样的包含块，则相对于body元素，即相对于浏览器窗口。

### 层模型--相对定位

需要设置position:relative（表示相对定位），它通过left、right、top、bottom属性确定元素在正常文档流中的偏移位置。相对定位完成的过程是首先按static(float)方式生成一个元素(并且元素像层一样浮动了起来)，然后相对于以前的位置移动，移动的方向和幅度由left、right、top、bottom属性确定，偏移前的位置保留不动(next元素的起始位置在偏移前后面开始)。

### 层模型--固定定位

fixed：表示固定定位，与absolute定位类型类似，但它的相对移动的坐标是视图（屏幕内的网页窗口）本身。由于视图本身是固定的，它不会随浏览器窗口的滚动条滚动而变化，除非你在屏幕中移动浏览器窗口的屏幕位置，或改变浏览器窗口的显示大小，因此固定定位的元素会始终位于浏览器窗口内视图的某个位置，不会受文档流动影响，这与background-attachment:fixed;属性功能相同。以下代码可以实现相对于浏览器视图向右移动100px，向下移动50px。并且拖动滚动条时位置固定不变。

### Relative与Absolute组合使用

相对于其它元素进行定位使用position:relative.必须遵守下面规范：

第一：参照定位的元素必须是相对定位元素的前辈元素：

|  |
| --- |
| <div id="box1"><!--参照定位的元素-->  <div id="box2">相对参照元素进行定位</div><!--相对定位元素-->  </div> |

第二：参照定位的元素必须加入position:relative;

|  |
| --- |
| #box1{  width:200px;  height:200px;  position:relative;  } |

第三：定位元素加入position:absolute，便可以使用top、bottom、left、right来进行偏移定位了。

|  |
| --- |
| #box2{  position:absolute;  top:20px;  left:30px;  } |
| #效果图 |

# 单位和值

## 颜色

英文命令颜色：前面几个小节中经常用到的就是这种设置方法：p{color:red;}

#RGB颜色,这个与 photoshop 中的 RGB 颜色是一致的，由 R(red)、G(green)、B(blue)

三种颜色的比例来配色。p{color:rgb(133,45,200);}每一项的值可以是 0~255 之间的整数，也可以是 0%~100% 的百分数。如： p{color:rgb(20%,33%,25%);}

#十六进制颜色，这种颜色设置方法是现在比较普遍使用的方法，其原理其实也是 RGB 设置，但是其每一项的值由 0-255 变成了十六进制 00-ff。

## 长度值

长度单位总结一下，目前比较常用到px（像素）、em、% 百分比，要注意其实这三种单位都是相对单位。

像素：像素为什么是相对单位呢？因为像素指的是显示器上的小点（CSS规范中假设“90像素=1英寸”）。实际情况是浏览器会使用显示器的实际像素值有关，在目前大多数的设计者都倾向于使用像素（px）作为单位。

em：就是本元素给定字体的 font-size 值，如果元素的 font-size 为 14px ，那么1em = 14px；如果 font-size 为 18px，那么 1em = 18px。如下代码：

p{font-size:12px;text-indent:2em;}

上面代码就是可以实现段落首行缩进24px（也就是两个字体大小的距离）。

下面注意一个特殊情况：但当给 font-size 设置单位为 em 时，此时计算的标准以 p 的父元素的 font-size 为基础。

百分比： 相对父元素。 p{font-size:12px;line-height:130%}

设置行高（行间距）为字体的130%（12 \* 1.3 = 15.6px）。

# CSS使用小技巧

|  |
| --- |
| #水平居中设置-行内元素：水平居中是通过给父元素设置 text-align:center  #水平居中设置-定宽块状元素：  margin-left:auto;  margin-right:auto;  #水平居中总结-不定宽块状元素方法：  加入 table 标签：用table标签的长度自适应性---即不定义其长度也不默认父元素body的长度（table其长度根据其内文本长度决定），因此可以看做一个定宽度块元素，然后再利用定宽度块状居中的margin的方法，使其水平居中。  设置 display: inline 方法：与第一种类似，显示类型设为 行内元素，然后使用 text-align:center 来实现居中效果。进行不定宽元素的属性设置；  <style>  .container{text-align:center;}  .container ul{list-style:none;margin:0;padding:0;display:inline;}  .container li{margin-right:8px;display:inline;}  </style>  </head>  <body>  <div class="container">  <ul>  <li><a href="#">1</a></li>  <li><a href="#">2</a></li>  <li><a href="#">3</a></li>  </ul>  </div>  </body>  设置 position:relative 和 left:50%：利用 相对定位 的方式，将元素向左偏移 50% ，即达到居中的目的。  #垂直居中-父元素高度确定的单行文本  置父元素的 height 和 line-height 高度一致来实现的：  height:100px; line-height:100px;  #垂直居中-父元素高度确定的多行文本  方法：使用插入 table (包括tbody、tr、td)标签，同时设置 vertical-align：middle。css 中有一个用于竖直居中的属性 vertical-align，在父元素设置此样式时，会对inline-block类型的子元素都有用。下面看一下例子：  #隐性改变display类型  当为元素（不论之前是什么类型元素，display:none 除外）设置以下 2 个句之一：  position : absolute  float : left 或 float:right  元素的display显示类型就会自动变为以 display:inline-block（块状元素）的方式显示，当然就可以设置元素的 width 和 height 了，且默认宽度不占满父元素。 |