# CSS样式

## 1、什么是CSS？

CSS 指层叠样式表 (Cascading Style Sheets)

CSS就是控制页面布局和样式

### 1.1、HTML和CSS的关系：

HTML 结构层 负责从 语义的角度搭建页面结构 （页面结构）

CSS 样式层 负责从 审美的角度美化页面（页面样式的表现）

JavaScript 行为层 负责从 交互的角度提升用户体验（交互行为）

## 2 如何编写CSS

### 2.1 内联样式style属性（优先级最高，不推荐使用）

<span style="color: red;">我的名字是alice</span>

### 2.2 内嵌样式（第2优先级，推荐使用）

<style>

p{

color: red;

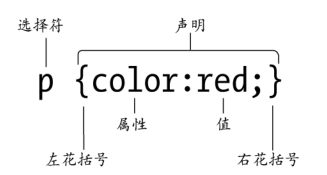
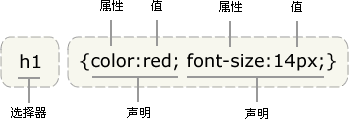
}

</style>

### 2.3外部样式（第3优先级，强烈推荐使用）

<link rel="stylesheet" href="./a.css" />

## 3、CSS语法

## 4、CSS简单属性

* width: 设置宽度，单位px像素
* height:高度
* color:设置字体颜色
* background-color: 背景色
* font-size:设置字体大小px像素

## 5、CSS注释

/\*\*/

## \*\*6、CSS选择器

### 6.1所有标签选择器（universal selector）

ie:\*{}

### \*6.2标签选择器(type selector)

i.e: p{}

### \*6.3 ID选择器(ID selector)

ie: #username

* 严格区分大小写
* 只允许字母开头

### \*6.4类选择器(class selector)

ie: .username{}

### \*6.5层级选择器（Descendant combinator）

ie: div span{}

### \*6.6并集选择器（comma combinator）

ie: .class,h3

### \*6.7子元素选择器（Child combinator）

ie: A>B

### 6.8兄弟选择器

### 6.8.1 相邻兄弟选择器（Adjacent sibling combinator）

注：只选一个

ie: A+B

### 6.8.2 兄弟选择器（General Sibling combinator）

注：选择多个

ie: A~B

### 6.9属性选择器（attribute selector）

**Syntax:**

 [attr] [attr=value] [attr~=value] [attr^=value][attr$=value] [attr\*=value]

* 简单属性选择：h1[attr]
  + ie: h1[class]{color:red;}
  + 说明：选择所有拥有class属性的h1标签
* 根据属性值选择：p[attr=value]
* 全包含（完整）：p[attr~=value]
  + 适用于多个值
  + ie:

<p aa="title asasdf">我的名字是bob</p>

<p aa="title">我的名字是bob</p>

* 属性名模糊匹配：p[attr\*=value]
* 属性模糊选择:
  + p[attr^=value] 开头
  + p[attr$=value] 结尾

### 6.10伪类选择器（Pseudo classes）

:hover a:hover{color:green}

:link a:link{color:green}

### 6.11优先级

优先级：当通过不同的选择器组合同时定位到一个地方的时候，这个时候就存在使用哪个css的问题

**默认样式（浏览器自带）<标签选择<类|属性选择器<ID选择器<行内样式<Important**

**0           1           10          100           1000         1000以上**

## 7、CSS文本样式

### 7.1块级元素

div, p, h1-h6, ul, li, ol

特点：独占一行、可以设置宽高、如果没有设置宽高则和父类一致

### 7.2行内元素

span a em strong del ins

特点：

* 在一行显示、不能设置宽高、元素的宽高就是内容支撑起来的
* 行内元素不能嵌套块级元素，只能嵌套其他行内标签

### 7.3行内块

image input

特点：在一行显示，可以设置宽和高

### 7.4行内元素和块级元素转换(display)

1、块元素转行内元素：display:inline;

2、行内元素转块元素：display:block; 转换完成以后就可以设置宽和高，可以使用text-align:center;了

3、块元素和行内元素转行内块元素：display:inline-block;

特点：同一行显示，并且可以设置宽和高

4、display:none：此元素不被显示，在文档中被移除。

面试题：visibility:hidden 和display:none区别？

visibility:hidden，还占用原来的位置，还存在文档中

### 7.5字体（font）

font-size: 字体大小

font-weight:字体粗细

font-style:字体风格

font设置字体属性连写：

font: font-style font-weight font-size/line-height font-family;

### 7.6 背景属性（background）

background-color: 背景颜色

background-image背景图片

background-repeat: repeat / no-repeat/ repeat-x/ repeat-y 背景平铺

background-position：left right center top bottom

背景属性连写：

background: red url(“1.png”) no-repeat 30px 40px;

### 7.7行高（line-height）

行高定义：设置行与行的举例。

特点：设置行高字体会居中显示。

### 7.8文本样式

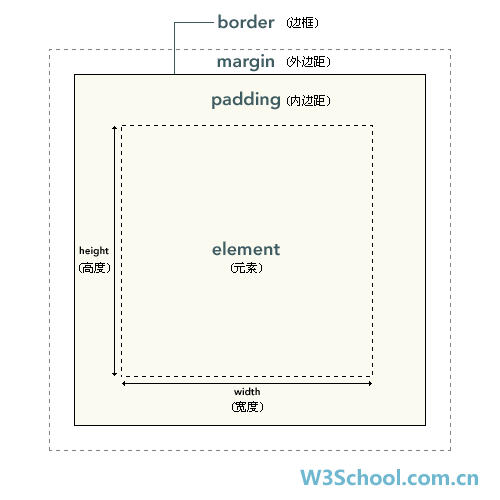
* color:设置文本颜色
  + RGBA
  + red
  + #eee
* text-decoration：文本装饰（a标签）
* text-align：文本对齐方式
* letter-spacing:字间距
* word-spacing: 单词间距

## 8、盒模型（Box Model）

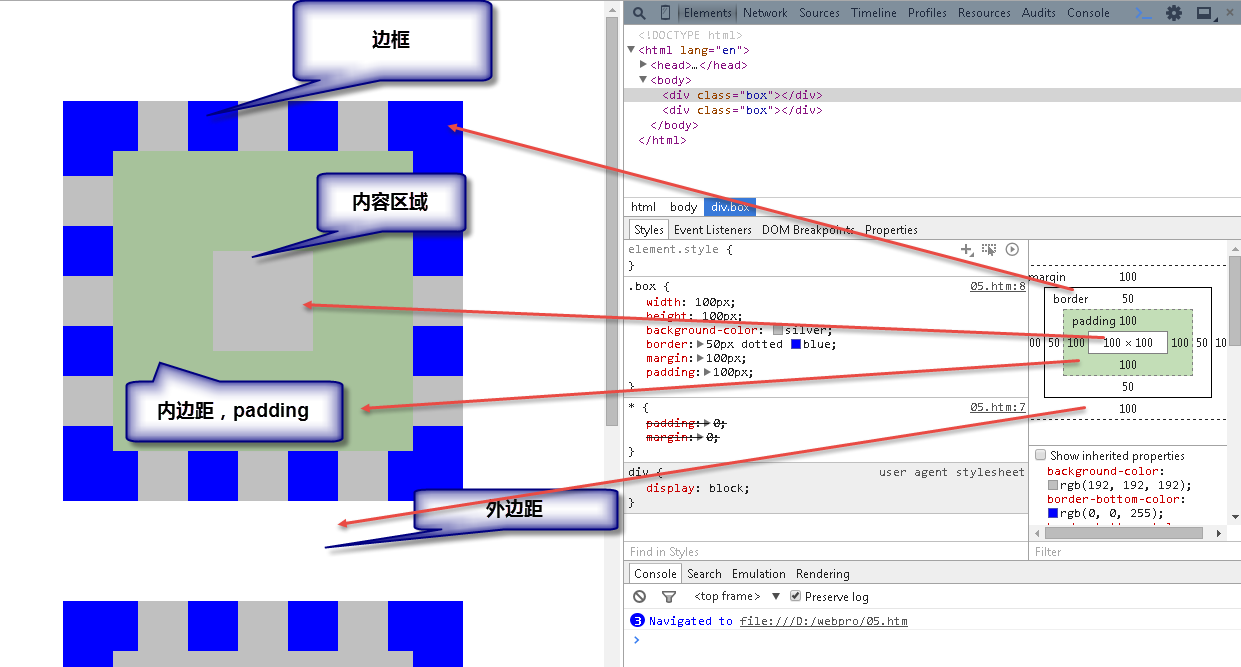
### 8.1、什么是盒模型？

所谓盒子模型就是把HTML页面中的元素看作是一个矩形的盒子，也就是一个盛装内容的容器。每个矩形都由元素的内容、内边距（padding）、边框（border）和外边距（margin）组成。

也就是说：margin, border, padding组成了盒模型



分析：



### 8.2 边框（border）

border-style: solid

border-width: 10px;

border-color: red

* 边框相同：

border: 100px solid red;

* + 100px->边框的粗细
  + solid-> 边框的类型dashed, dotted, solid
  + red->边框颜色
* 边框不同：

border-top/right/bottom/left: 1px solid red

border-top/right/bottom/left-color: red;

border-top/right/bottom/left-style: solid

#### \*连写规则：

border-style: solid

全部实线

border-style: solid dashed

上下是实线，右左是虚线

border-style: solid dotted dashed

上实线，右左圆点，下虚线

border-style: solid dotted dashed solid

上右下左

### 8.3内边距（padding）

padding: 20px;

padding:20px 30px;

padding:20px 15px 30px;

padding: 20px 10px 5px 30px;

### 8.4外边距（margin）

margin: 20px;

margin:20px 30px;

margin:20px 15px 30px;

margin: 20px 10px 5px 30px;

### \*8.5、margin, padding, border注意事项：

#### 8.5.1 外边距（margin）距属性

* + 垂直方向外边框合并

两个盒子垂直一个设置上外边距，一个设置下外边距，取的设置较大的值。

* 嵌套块元素的外边距合并（高度塌陷）

俩个盒子都设置上边距，会取最大的值。

**解决方法：后面再讲**

* 行内元素不要给上下的margin 和padding
  + 上下margin和padding会被忽略。
  + 左右margin和padding会起作用。

### 8.6、盒模型相关样式

#### 8.6.1、display：设置元素显示样式

可选值：

* none: 元素不会在页面中显示，并且不会占据页面的位置
* block: 元素会作为块元素显示
* inline; 元素会作为内联元素显示
* inline-block:
  + 元素会作为行内块元素显示
  + 既具有内联元素的特点也具有块元素的特点:
    - 不独占一行
    - 可以设置宽高

#### 8.6.2、visibility：设置元素是否在页面显示

可选值：

* visible:可见
* hidden:隐藏

还占据原来的位置

#### 8.6.3、overflow：设置元素如何处理溢出内容

可选值：

* visible: 默认值，不会处理溢出的内容，在父元素以外的地方显示
* hidden: 溢出的内容会被隐藏不会显示
* scroll: 在父元素中同时添加水平和垂直方向的滚动条
* auto: 根据需要自动生成滚动条

## 9、文档流/标准流（normal flow）

文档流指的是网页中的一个位置，文档流是网页的基础，是网页的最底层，所有的元素默认都是在文档流中排列。

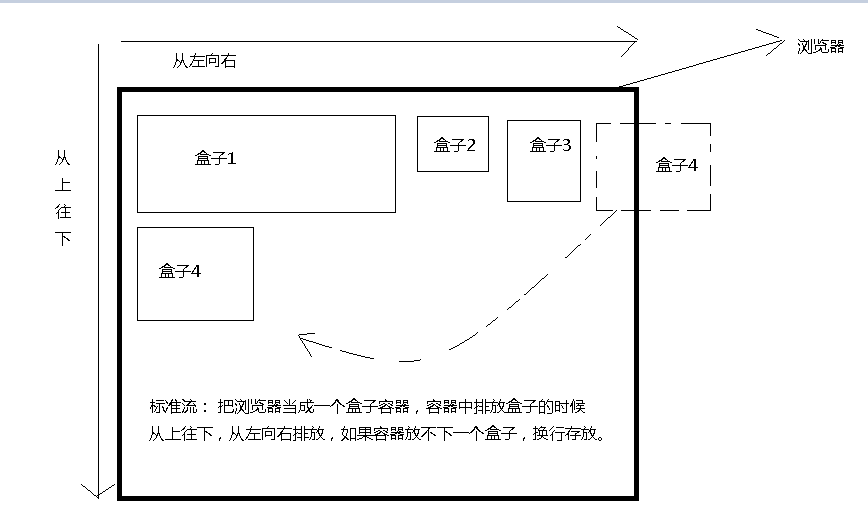
元素在文档流中默认自左向右，自上向下排列（和我们的书写习惯一致）

* 块级元素：

1. 块元素在文档流中自上向下排列
2. 块元素在文档流中宽度默认是父元素的100%（独占一行）
3. 块元素在文档流中高度默认被内容撑开

* 内联元素：

1. 内联元素在文档流中自左向右排列，如果一行中不足以容下所有的内联元素，则换到下一行，继续自左至右排列
2. 内联元素在文档流中宽度和高度默认都被内容撑开



### \*\*9.1 浮动（float）

什么是浮动？

元素的浮动是指设置了浮动属性的元素会脱离标准文档流的控制，移动到其父元素中指定位置的过程。

定义浮动：float:left、right

#### \*\*浮动特性：

1.浮动脱离标准流，不占位置，但会影响标准流。浮动只有左右浮动。

2. 浮动的元素A排列位置，跟上一个元素（块级）有关系。如果上一个元素有浮动，则A元素顶部会和上一个元素的顶部对齐；如果上一个元素是标准流，则A元素的顶部会和上一个元素的底部对齐。

3. 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

4.不会独占1行

5.宽度和高度都被内容撑开

6.内联元素会变成块级元素

**7.浮动元素具有包裹性**

### 9.2 父盒子高度塌陷

高度塌陷情景：

box1包含box2，box1下面有Box3

box1（不设宽高）包含box2，box2设置宽高并且浮动，box1会高度塌陷。box3会向上移动。

**解决方案1：**

给父级box1设置浮动（触发BFC），给box3清楚浮动。

BFC概念：

根据W3C的标准，在页面中元素都一个隐含的属性叫做Block Formatting Context。

简称BFC，该属性可以设置打开或者关闭，默认是关闭的。

当开启元素的BFC以后，元素将会具有如下的特性：

**\*\*BFC内部和BFC外部布局不互相影响\*\***

**解决方案2：**

给父级box1设置绝对定位position:absolute;

缺点：会导致Box3上移，布局紊乱。

**解决方案3：**

给父级Box1设置display: inline-block;

**解决方案4：**

给父级box1设置overflow:hidden;

**\*\*\*终极解决方案：**

给父级box1添加clearfix类

**.clearfix::after {**

**content: "";**

**clear: both;**

**display: block;**

**visibility: hidden;**

**height：0；**

**}**

### 9.3定位（position）

|  |  |
| --- | --- |
| 值 | 描述 |
| static | 自动定位（默认定位方式） |
| relative | 相对定位，相对于其原文档流的位置进行定位 |
| absolute | 绝对定位，相对于其上一个已经定位的父元素进行定位 |
| fixed | 固定定位，相对于浏览器窗口进行定位 |

偏移属性：top, right, bottom, left

#### static（默认定位）

特点：

在静态定位状态下，无法通过边偏移属性（top、bottom、left或right）来改变元素的位置。

#### relative（相对定位）

特点：

* + 相对定位是将元素相对于它在**标准流**中的位置进行定位
  + 可以通过边偏移属性改变元素的位置，但是它在文档流中的位置仍然保留
  + 可以设置偏移量



#### absolute（绝对定位）

* 绝对定位是将元素依据最近的已经定位（绝对、固定或相对定位）的父元素进行定位。
* 若所有父元素都没有定位，则依据body根元素进行定位。当position属性的取值为absolute时，可以将元素的定位模式设置为绝对定位。
* ！！！脱标，不占据原来位置。
* 可以设置偏移量



绝对定位特性：

* 包裹性
  + 根据实际内容来撑开绝对定位元素的大小。
* 破坏性，导致父容器塌陷
  + 由于脱离了文档流，导致父容器塌陷。
* 悬浮性
  + 会漂浮起来，在原始流的元素之上。

#### fixed（固定定位）

* 以浏览器窗口作为参照物来定义网页元素
* 脱离标准文档流的控制，始终依据浏览器窗口来定义自己的显示位置。不管浏览器滚动条如何滚动，也不管浏览器窗口的大小如何变化，该元素都会始终显示在浏览器窗口的固定位置
* 可以设置偏移量。

#### z-index 层叠等级属性

要想调整重叠**定位元素的堆叠顺序**，可以对定位元素应用z-index层叠等级属性，其取值可为正整数、负整数和0。

**z-index的默认属性值是0**，取值越大，定位元素在层叠元素中越居上。

#### 技巧：父相子绝

## 10、CSS常用技巧

### 10.1、logo优化：text-indent：-2000em

### 10.2、CSS重要技巧

#### 10.2.1、父相子绝

#### 10.2.2、ul>li>a>img，a标签不能嵌套a和input标签

#### 10.2.3、背景图片和文字对齐

* 第一种方式：

<li>新闻列表</li> 通过设置li的背景图片，调整背景图片的位置。再通过padding来设置文字的位置，进行对齐

* 第二种方式：

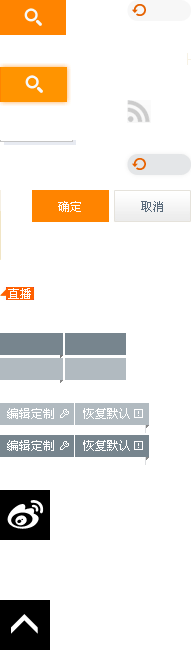
将背景图片，设置在i标签上，i标签是行内元素，设置i标签为Inline-block（目的可以设置宽高）并且verticle-align middle来进行对齐。使用margin-top进行微调。

### 10.2、精灵技术

#### 为什么要用精灵技术？

目的：优化网站，减少像服务器发送请求。

CSS 精灵其实是将网页中的一些背景图像整合到一张大图中（精灵图）



#### 精灵图使用方式

通过设置background image来调整显示的位置。

### 10.3、CSS滑动门



图片：

ul,li{

padding: 0;

margin: 0;

}

ul{

height: 40px;

line-height: 40px;

}

li{

border: 1px solid black;

float: left;

list-style: none;

margin:0 10px;

background: url('./images/l.png') 0 3px no-repeat;

padding-left: 5px;

}

a{

float: left;

background: url('./images/r.png') right no-repeat;

padding-right: 20px;

}

<ul>

<li><a>菜单1</a></li>

<li><a>菜单2</a></li>

<li><a>菜单dddd3</a></li>

</ul>

margin 负值的应用