CSS学习（10.19）



CSS格式

p {color:red;text-align:center;}

CSS注释以 **/\*** 开始, 以 **\*/** 结束, 实例如下:

/\*这是个注释\*/

HTML元素以id属性来设置id选择器,CSS 中 id 选择器以 "#" 来定义

class 选择器在 HTML 中以 class 属性表示, 在 CSS 中，类选择器以一个点 **.** 号显示：

在以下的例子中，所有拥有 center 类的 HTML 元素均为居中。

.center {text-align:center;}

外部样式表

当样式需要应用于很多页面时，外部样式表将是理想的选择。在使用外部样式表的情况下，你可以通过改变一个文件来改变整个站点的外观。每个页面使用 <link> 标签链接到样式表。 <link> 标签在（文档的）头部：

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

浏览器会从文件 mystyle.css 中读到样式声明，并根据它来格式文档。

外部样式表可以在任何文本编辑器中进行编辑。文件不能包含任何的 html 标签。样式表应该以 .css 扩展名进行保存。下面是一个样式表文件的例子：

hr {color:sienna;} p {margin-left:20px;} body {background-image:url("/images/back40.gif");}

*IMG_256 不要在属性值与单位之间留有空格（如："margin-left: 20 px" ），正确的写法是 "margin-left: 20px" 。*

## 内部样式表

当单个文档需要特殊的样式时，就应该使用内部样式表。你可以使用 <style> 标签在文档头部定义内部样式表，就像这样:

<head> <style> hr {color:sienna;} p {margin-left:20px;} body {background-image:url("images/back40.gif");} </style> </head>

**多重样式优先级**

**（内联样式）Inline style > （内部样式）Internal style sheet >（外部样式）External style sheet > 浏览器默认样式**

**注意：**如果外部样式放在内部样式的后面，则外部样式将覆盖内部样式

CSSbackgrounds

background-image 属性描述了元素的背景图像.

默认情况下，背景图像进行平铺重复显示，以覆盖整个元素实体

让背景图片不平铺；XY平铺

background-repeat:no-repeat;

background-repeat:repeat-x;

利用 background-position 属性改变图像在背景中的位置:

background-position:right top;

为了简化这些属性的代码，我们可以将这些属性合并在同一个属性中.

背景颜色的简写属性为 "background":

* background-color
* background-image
* background-repeat
* background-attachment
* background-position

body {background:#ffffff url('img\_tree.png') no-repeat right top;}

****CSS的****background-attachment****属性****定义了背景图像是固定的还是随着页面滚动而滚动。这个属性接受几个关键值：scroll、fixed和local。

如果您想要背景图像在页面滚动时保持固定，您可以设置background-attachment: fixed;

如果您希望背景图像随着元素的内容滚动，而不是整个页面，您可以使用background-attachment: local;

## 文本的对齐方式

文本排列属性是用来设置文本的水平对齐方式。

文本可居中或对齐到左或右,两端对齐.

当text-align设置为"justify"，每一行被展开为宽度相等，左，右外边距是对齐（如杂志和报纸）。

h1 {text-align:center;} p.date {text-align:right;} p.main {text-align:justify;}

设置行间距

<p class="small">

<p class="big">

设置字间距（在设置英文时是定义字母间距）

h2 {letter-spacing:-3px;}

增加单词之间的空白空间（设置单词之间间距）

**word-spacing:30px;**

禁用文字换行

white-space:nowrap;

设置文字阴影

h1 {text-shadow:2px 2px #FF0000;}

字体

如果字体系列的名称超过一个字，它必须用引号，如Font Family："宋体"。

多个字体系列是用一个逗号分隔指明：

p{font-family:"Times New Roman", Times, serif;}

## Serif 字体（结尾带拖尾）: 衬线体,Times 字体，衬线字体

## sans - serif字体(不带拖尾):草书，无衬线体,日内瓦

## Monospace 字体（等宽）:摩纳哥，等宽，信使

正常与斜体：

p.normal {font-style:normal;}  
p.italic {font-style:italic;}

设置文字大小

## 设置字体大小像素

设置文字的大小与像素，让您完全控制文字大小（不指定时文字大小16px）：

h1 {font-size:40px;}

em设置字体大小（可以在所有浏览器都可以设置文字大小）

1em的默认大小是16px。可以通过下面这个公式将像素转换为em：**px/16=em**

h1 {font-size:2.5em;} /\* 40px/16=2.5em \*/  
h2 {font-size:1.875em;} /\* 30px/16=1.875em \*/

在所有浏览器的解决方案中，设置 <body>元素的默认字体大小的是百分比：

body {font-size:100%;}

设置文字粗细：

p.normal {font-weight:normal;}(正常)

p.light {font-weight:lighter;}（更细）

p.thick {font-weight:bold;}（加粗）

p.thicker {font-weight:900;}（加粗pro）

字体设置的简化

p.ex1

{

font:15px arial,sans-serif;

}

CSS链接

显示交互（不可更改顺序）：

a:link {color:#000000;} /\* 未访问链接\*/

a:visited {color:#00FF00;} /\* 已访问链接 \*/

a:hover {color:#FF00FF;} /\* 鼠标移动到链接上 \*/

a:active {color:#0000FF;} /\* 鼠标点击时 \*/

顺序口诀

L(link)OV(visited)E and H(hover)A(active)TE

创建链接框

CSS列表

无序列表：

ul.a {list-style-type:circle;}

ul.b {list-style-type:square;}

有序列表

ol.c {list-style-type:upper-roman;} /\*罗马数字\*/

ol.d {list-style-type:lower-alpha;}/\*英文小写\*/

图符列表

ul

{list-style-image:url('sqpurple.gif');}

列表简写顺序

可以按顺序设置如下属性：

* list-style-type
* list-style-position (有关说明，请参见下面的CSS属性表)
* list-style-image

CSS表格边框

table { border-collapse:collapse; }/折叠th.td为一个边框/

table,th, td { border: 1px solid black; }/设置1像素黑色边框/

边框长宽：

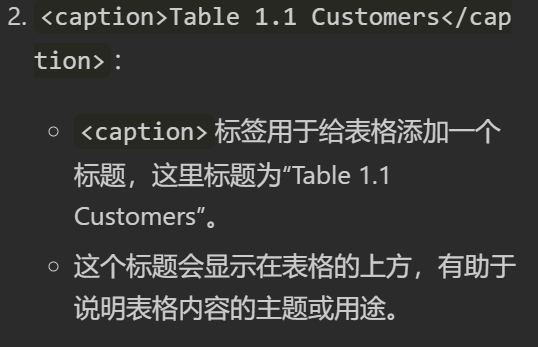
table { width:100%; }/可用百分比/

th { height:50px; }/也可以用像素/

控制边框和表格内容之间的距离

td { padding:15px; }

<caption>Table 1.1 Customers</caption>



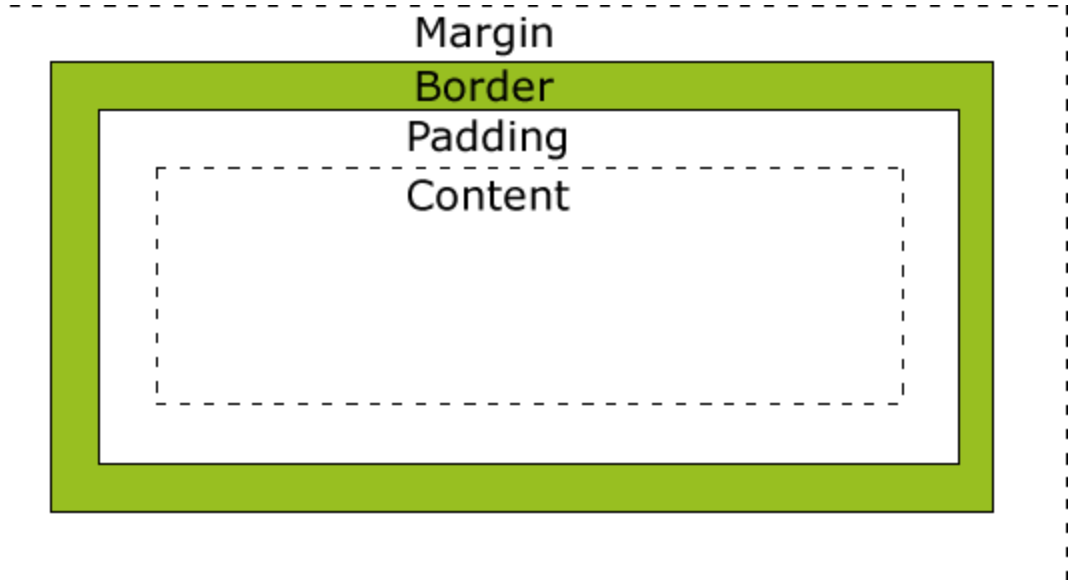
CSS盒子模型

元素空间计算

试想一下，你只有 250 像素的空间。让我们设置总宽度为 250 像素的元素:

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

div { width: 220px; padding: 10px; border: 5px solid gray; margin: 0; }



CSS边框

p.none {border-style:none;}/无边框/

p.dotted {border-style:dotted;}/点线边框/

p.dashed {border-style:dashed;}/虚线边框/

p.solid {border-style:solid;}/实线边框/

p.double {border-style:double;}/3D/

p.groove {border-style:groove;}/3D/

p.ridge {border-style:ridge;}/3D/

p.inset {border-style:inset;}/3D/

p.outset {border-style:outset;}/3D/

p.hidden {border-style:hidden;}

p.mix {border-style: dotted dashed solid double;}

CSS轮廓（outline不占据空间）

outline-style:dotted;/种类/

outline-width:3px;/宽度/

outline-color:#00ff00;/颜色/

CSS外边距margin(定义元素周围的空间)

Margin可以使用负值，重叠的内容

在CSS中，它可以指定不同的侧面不同的边距：

margin-top:100px; margin-bottom:100px; margin-right:50px; margin-left:50px;

## Margin - 简写属性

margin属性可以有一到四个值。

* **margin:25px 50px 75px 100px;**
  + 上边距为25px
  + 右边距为50px
  + 下边距为75px
  + 左边距为100px

CSS填充

## 填充 - 简写属性

Padding属性，可以有一到四个值。

**padding:25px 50px 75px 100px;**

* 上填充为25px
* 右填充为50px
* 下填充为75px
* 左填充为100px

CSS分组和嵌套

为了尽量减少代码，你可以使用分组选择器。

## 实例

h1,h2,p { color:green;}

嵌套选择器

## 嵌套选择器

它可能适用于选择器内部的选择器的样式。

在下面的例子设置了四个样式：

* **p{ }**: 为所有 **p** 元素指定一个样式。
* **.marked{ }**: 为所有 **class="marked"** 的元素指定一个样式。
* **.marked p{ }**: 为所有 **class="marked"** 元素内的 **p** 元素指定一个样式。
* **p.marked{ }**: 为所有 **class="marked"** 的 **p** 元素指定一个样式。

CSS尺寸

使用百分比值设置元素的高度；使用px设置元素的高度

{height:120px;}

设置元素的最大宽度

max-width:20%;

CSS Display

元素的隐藏

h1.hidden {visibility:hidden;}/占用空间隐藏/

h1.hidden {display:none;} {visibility: collapse;}

/不占用空间隐藏/

## CSS Display - 块和内联元素

块元素是一个元素，占用了全部宽度，在前后都是换行符。

块元素的例子：

<h1><p><div>

内联元素只需要必要的宽度，不强制换行。

内联元素的例子：<span><a>

更改块元素为内联元素：

display:inline;**属性将元素设置为内联元素，这意味着元素前后没有换行符，它会与其他内联元素一起显示在同一行上[1](https://blog.csdn.net/weixin_45389633/article/details/103743017" \t "https://cn.bing.com/_blank)**

下面的示例把span元素作为块元素（内联变块元素）

## 实例

span {display:block;}

CSS定位

## static 定位

HTML 元素的默认值，没有定位

## fixed 定位

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：

p.pos\_fixed { position:fixed; top:30px; right:5px; }

## relative 定位

相对定位元素的定位是相对其正常位置。移动并不改变它所占空间。

## absolute 定位

绝对定位的元素的位置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>:

absolute 定位使元素的位置与文档流无关，因此不占据空间。

## sticky 定位

sticky 英文字面意思是粘，粘贴，所以可以把它称之为粘性定位。

**position: sticky;** 基于用户的滚动位置来定位。

粘性定位的元素是依赖于用户的滚动，在 **position:relative** 与 **position:fixed** 定位之间切换。

它的行为就像 **position:relative;** 而当页面滚动超出目标区域时，它的表现就像 **position:fixed;**，它会固定在目标位置。

元素定位表现为在跨越特定阈值前为相对定位，之后为固定定位。

div.sticky { position: -webkit-sticky; /\* Safari \*/ position: sticky; top: 0; background-color: green; border: 2px solid #4CAF50; }

**Overflow**

属性创建一个滚动条，当一个元素的内容在指定的区域过大时如何设置以适应

overflow: auto;

|  |  |
| --- | --- |
| visible | 默认值。内容不会被修剪，会呈现在元素框之外。 |
| hidden | 内容会被修剪，并且其余内容是不可见的。 |
| scroll | 内容会被修剪，但是浏览器会显示滚动条以便查看其余的内容。 |
| auto | 如果内容被修剪，则浏览器会显示滚动条以便查看其余的内容。 |
| inherit | 规定应该从父元素继承 overflow 属性的值。 |

更改光标

<span style="cursor:auto">auto</span><br>

# CSS Float(浮动)

图像是右浮动，下面的文本流将环绕在它左边

img

{float:right;}

## 彼此相邻的浮动元素

如果你把几个浮动的元素放到一起，如果有空间的话，它们将彼此相邻

使用 clear 属性往文本中添加图片廊：

.text\_line { clear:both; }

**CSS水平垂直对齐**

## **元素居中对齐**

要水平居中对齐一个元素(如 <div>), 可以使用 **margin: auto;**

## **文本居中对齐**

如果仅仅是为了文本在元素内居中对齐，可以使用 **text-align: center;**

## 图片居中对齐

要让图片居中对齐, 可以使用 **margin: auto;** 并将它放到 **块** 元素中:

## 左右对齐 - 使用定位方式

我们也可以使用 **float** 属性来对齐元素:

我们可以使用 **position: absolute;** 属性来对齐元素:

当使用 **position** 来对齐元素时, 通常 **<body>** 元素会设置 **margin** 和 **padding** 。 这样可以避免在不同的浏览器中出现可见的差异。

body { margin: 0; padding: 0; }

***如果子元素的高度大于父元素，且子元素设置了浮动，那么子元素将溢出，这时候你可以使用 "clearfix(清除浮动)" 来解决该问题。overflow***

如果要水平和垂直都居中，可以使用 **padding** 和 **text-align: center**:

**CSS组合选择器**

## 后代选择器

后代选择器用于选取某元素的后代元素。

以下实例选取所有 <p> 元素插入到 <div> 元素中: div p { background-color:yellow; }

## 子元素选择器

与后代选择器相比，子元素选择器（Child selectors）只能选择作为某元素直接/一级子元素的元素。

以下实例选择了<div>元素中所有直接子元素 <p> ：div>p

## 相邻兄弟选择器

相邻兄弟选择器（Adjacent sibling selector）可选择紧接在另一元素后的元素，且二者有相同父元素。div+p

## 后续兄弟选择器

后续兄弟选择器选取所有指定元素之后的相邻兄弟元素。

以下实例选取了所有 <div> 元素之后的所有相邻兄弟元素 <p> :

div~p

# CSS 伪类(Pseudo-classes)

CSS伪类是用来添加一些选择器的特殊效果。

将链接或文字中部分内容给予特殊效果

## :first-line 伪元素

"first-line" 伪元素用于向文本的首行设置特殊样式

## :first-letter 伪元素

"first-letter" 伪元素用于向文本的首字母设置特殊样式

## CSS - :before 伪元素

":before" 伪元素可以在元素的内容前面插入新内容。

下面的例子在每个 <h1>元素前面插入一幅图片：

h1:before { content:url(smiley.gif); }

## CSS - :after 伪元素

":after" 伪元素可以在元素的内容之后插入新内容。

下面的例子在每个 <h1> 元素后面插入一幅图片：

**CSS导航条**

display:block - 显示块元素的链接，让整体变为可点击链接区域（不只是文本），它允许我们指定宽度

## 垂直导航条实例

创建一个简单的垂直导航条实例，在鼠标移动到选项时，修改背景颜色：

## 激活/当前导航条实例

在点击了选项后，我们可以添加 "active" 类来标注哪个选项被选中：

## 全屏高度的固定导航条

接下来我们创建一个左边是全屏高度的固定导航条，右边是可滚动的内容。

## 水平导航栏

有两种方法创建横向导航栏。使用**内联(inline)**或**浮动(float)**的列表项。li { display:inline; }li { float:left; }

这两种方法都很好，但如果你想链接到具有相同的大小，你必须使用浮动的方法。

### **添加分割线**<li> 通过 **border-right** 样式来添加分割线:

### **固定导航条**

### 可以设置页面的导航条固定在头部或者底部：

ul { position: fixed; top: 0; width: 100%; }

块元素换行符

display: block; /\* 使链接出现在彼此下方而不是并排 \*/

# CSS 下拉菜单

**CSS 部分：**

.dropdown 类使用 position:relative, 这将设置下拉菜单的内容放置在下拉按钮 (使用 position:absolute) 的右下角位置。

.dropdown-content 类中是实际的下拉菜单。默认是隐藏的，在鼠标移动到指定元素后会显示。 注意 min-width 的值设置为 160px。你可以随意修改它。 **注意:** 如果你想设置下拉内容与下拉按钮的宽度一致，可设置 width 为 100% ( overflow:auto 设置可以在小尺寸屏幕上滚动)。

### **实例**

<style>  
/\* 下拉按钮样式 \*/  
.dropbtn {  
    background-color: #4CAF50;  
    color: white;  
    padding: 16px;  
    font-size: 16px;  
    border: none;  
    cursor: pointer;  
}  
/\* 容器 <div> - 需要定位下拉内容 \*/  
.dropdown {  
    position: relative;  
    display: inline-block;  
}  
/\* 下拉内容 (默认隐藏) \*/  
.dropdown-content {  
    display: none;  
    position: absolute;  
    background-color: #f9f9f9;  
    min-width: 160px;  
    box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);  
}  
/\* 下拉菜单的链接 \*/  
.dropdown-content a {  
    color: black;  
    padding: 12px 16px;  
    text-decoration: none;  
    display: block;  
}  
/\* 鼠标移上去后修改下拉菜单链接颜色 \*/  
.dropdown-content a:hover {background-color: #f1f1f1}  
  
/\* 在鼠标移上去后显示下拉菜单 \*/  
.dropdown:hover .dropdown-content {  display: block;}  
/\* 当下拉内容显示后修改下拉按钮的背景颜色 \*/  
.dropdown:hover .dropbtn {background-color: #3e8e41;}  
</style>  
<div class="dropdown">  
  <button class="dropbtn">下拉菜单</button>  
  <div class="dropdown-content">  
    <a href="#">菜鸟教程 1</a>  
    <a href="#">菜鸟教程 2</a>  
    <a href="#">菜鸟教程 3</a>  
  </div>  
</div>

一般left和right在一个样式中只能使用一次，同时使用会出现兼容问题，left优先级高于right

# CSS 提示工具(Tooltip)

基础提示框(Tooltip)

提示框在鼠标移动到指定元素上显示：

**tooltip** 类使用 **position:relative**, 提示文本需要设置定位值 **position:absolute**

<style>

/\* Tooltip 容器 \*/ .tooltip {

position: relative;

display: inline-block;

border-bottom: 1px dotted black;/\* 悬停元素上显示点线 \*/ }

/\* Tooltip 文本 \*/

.tooltip .tooltiptext { visibility: hidden;

width: 120px;

background-color: black;

color: #fff;

text-align: center;

padding: 5px 0;

border-radius: 6px; /\*设置圆角为6像素，使toltip的角更柔和，外观更现代。/\*

/\* 定位 \*/ position: absolute;

z-index: 1;/\*设置Tooltip的堆叠顺序，使其在重叠情况下高于其他元素（默认为0）这样Tooltip能够正常显示在其他内容上方\*/ }

/\* 鼠标移动上去后显示提示框 \*/

.tooltip:hover .tooltiptext

{ visibility: visible; }

</style>

<div class="tooltip">鼠标移动到这

<span class="tooltiptext">提示文本</span>

</div>

## 定位提示工具

以下实例中，提示工具显示在指定元素的右侧(**left:105%**) 。

注意 **top:-5px** 同于定位在容器元素的中间。使用数字 **5** 因为提示文本的顶部和底部的内边距（padding）是 5px。

如果你修改 padding 的值，top 值也要对应修改，这样才可以确保它是居中对齐的。

在提示框显示在左边的情况也是这个原理。

## 显示在右侧：

.tooltip .tooltiptext { top: -5px; left: 105%; }

如果你想要提示工具显示在头部和底部。我们需要使用 **margin-left** 属性，并设置为 -60px。

这个数字计算来源是使用宽度的一半来居中对齐，即： width/2 (120/2 = 60)。

## 显示在头部：

.tooltip .tooltiptext

{ width: 120px; bottom: 100%; left: 50%; margin-left: -60px;

/\* 使用一半宽度 (120/2 = 60) 来居中提示工具 \*/ }

## 添加箭头

.tooltip .tooltiptext::after {

    content: "";/\*申明伪元素内容，由于是空字符串，意味着这个元素本身不会显示任何内容，但它仍会存在于文档流中，用于样式和定位\*/

    position: absolute;

    top: 100%;/\*这里将伪元素的顶部边缘放置在其包含块的底部边缘（100%表示完整高度）这通常用于定位工具的箭头，使其出现在工具提示文本的下方\*/

    left: 50%;

    margin-left: -5px;/\*居中效果\*/

    border-width: 5px;/\*定义伪元素边框宽度为5px\*/

    border-style: solid;/\*实线\*/

    border-color: black transparent transparent transparent;/\*设置四个边框颜色，只有上边框是黑色（箭头颜色）其他三条边透明\*/

}

## 淡入效果

我们可以使用 CSS3 transition 属性及 opacity 属性来实现提示工具的淡入效果:

## 淡入效果：

.tooltip .tooltiptext

{ opacity: 0; /\*设置提示文本的初始透明度为0，意味着不可见\*/

transition: opacity 1s; } /\*定义一个过渡效果，当透明度变化时，会在一秒内平滑过度\*/

.tooltip:hover .tooltiptext

{ opacity: 1; }/\*当鼠标悬停在.Tooltip的元素上时，提示文本透明度变为1

# CSS 图片廊

。。。。。[CSS 图片廊 | 菜鸟教程](https://www.runoob.com/css/css-image-gallery.html)

# CSS 图像透明/不透明

## 图像的透明度 - 悬停效果

CSS3中属性的透明度是 **opacity**。

img

{ opacity:0.4;

filter:alpha(opacity=40); /\* IE8 及其更早版本 \*/ }

img:hover

{

opacity:1.0;

filter:alpha(opacity=100); /\* 适用 IE8 及其更早版本 \*/

}

## 实例3 - 透明的盒子中的文字

# CSS 图像拼合技术

## 图像拼合 - 简单实例

img.home  
{  
width:46px;  
height:44px;  
background:url(img\_navsprites.gif) 0 0;  
}

**实例解析：**

* <img class="home" src="img\_trans.gif" /> -因为不能为空,src属性只定义了一个小的透明图像。显示的图像将是我们在CSS中指定的背景图像
* 宽度：46px;高度：44px; - 定义我们使用的那部分图像
* background:url(img\_navsprites.gif) 0 0; - 定义背景图像和它的位置（左0px，顶部0px）

## 图像拼合 - 创建一个导航列表

## 图像拼合s - 悬停效果

#home a:hover{background: url('img\_navsprites\_hover.gif') 0 -45px;}  
#prev a:hover{background: url('img\_navsprites\_hover.gif') -47px -45px;}  
#next a:hover{background: url('img\_navsprites\_hover.gif') -91px -45px;}

* 由于该列表项包含一个链接，我们可以使用：hover伪类
* #home a:hover{background: transparent url(img\_navsprites\_hover.gif) 0 -45px;} - 对于所有三个悬停图像，我们指定相同的背景位置，只是每个再向下45px

# CSS 媒体类型

## @media 规则

@media 规则允许在相同样式表为不同媒体设置不同的样式。

@media screen

|  |  |
| --- | --- |
| screen | 用于电脑显示器。 |

{ p.test {font-family:verdana,sans-serif;font-size:14px;}

} @media print

|  |  |
| --- | --- |
| print | 用于打印机。 |

{ p.test {font-family:times,serif;font-size:10px;} }

@media screen,print

{ p.test {font-weight:bold;} }

**voice-family** 属性是专为听觉用户代理。其他一些属性可用于不同的媒体类型。例如， **font-size** 属性可用于屏幕和印刷媒体，但有不同的值。屏幕和纸上的文件不同，通常需要一个更大的字体，**sans-serif** 字体比较适合在屏幕上阅读，而 **serif** 字体更容易在纸上阅读。

# CSS 属性 选择器

CSS 属性选择器用于根据元素的属性或属性值来选择 HTML 元素。

属性选择器可以帮助你在不需要为元素添加类或 ID 的情况下对其进行样式化。

/\* 选择所有具有 `type` 属性的元素 \*/[type] {

border: 1px solid red;}

# CSS 表单

我们使用 CSS 来渲染 HTML 的表单元素：

<form>/\*用于定义一个表单，表单是用户与用户进行交互的方式，可以收集输入的数据并发送到服务器处理\*/

  <label for="fname">/label用于为表单输入元素提供描述性文本////for=fname属性将标签与输入框关联，点击标签输入框会获得焦点/First Name</label>

  <input type="text" id="fname" name="fname">/input用于创建输入元素,这里用于创建输入框/

</form>

# CSS 计数器

## 使用计数器自动编号

body { counter-reset: section; }

h2::before { counter-increment: section; content: "Section " counter(section) ": "; }

* counter-reset - 创建或者重置计数器
* counter-increment - 递增变量
* content - 插入生成的内容
* counter() 或 counters() 函数 - 将计数器的值添加到元素

## CSS 计数器属性

|  |  |
| --- | --- |
| **属性** | **描述** |
| [content](https://www.runoob.com/cssref/pr-gen-content.html" \t "https://www.runoob.com/css/_blank) | 使用 ::before 和 ::after 伪元素来插入自动生成的内容 |
| [counter-increment](https://www.runoob.com/cssref/pr-gen-counter-increment.html" \t "https://www.runoob.com/css/_blank) | 递增一个或多个值 |
| [counter-reset](https://www.runoob.com/cssref/pr-gen-counter-reset.html" \t "https://www.runoob.com/css/_blank) | 创建或重置一个或多个计数器 |

# CSS 网页布局

## 网页布局

网页布局有很多种方式，一般分为以下几个部分：**头部区域、菜单导航区域、内容区域、底部区域**。

# CSS !important 规则

### **什么是 !important**

CSS 中的 !important 规则用于增加样式的权重。

**!important** 与优先级无关，但它与最终的结果直接相关，使用一个 !important 规则时，此声明将覆盖任何其他声明。

**使用建议：**

* **一定**要优先考虑使用样式规则的优先级来解决问题而不是 !important
* **只有**在需要覆盖全站或外部 CSS 的特定页面中使用 !important
* **永远不要**在你的插件中使用 !important
* **永远不要**在全站范围的 CSS 代码中使用 !important

#myid { background-color: blue; }

.myclass { background-color: gray; }

p { background-color: red !important; }